Study Setup User Guide

Study Setup User Guide

Release 5.5.0

Publication date Aug 30, 2022

Abstract

This guide describes the general process of defining a study database in DFdiscover and specifically the use of **DFsetup** to assist in study definition.

Table of Contents

List of Figures

List of Tables

List of Examples

Preface

Images in this document are of **DFsetup** in a Windows 10 environment. Functionality is identical on all operating systems, and images are identical, except for window decorations.

Instructions to *click* an object require either pointing to and clicking the object using the mouse, or by keyboarding to it and pressing **space** or **Return**.

Instructions to select **Menu** > **Option** refer to the selection of menu items from the application menubar. For example, to open the Plates window, select **View** > **Plates**.

The instructions contained in this guide describe how to setup a new study database, define the data fields, and enter other relevant study configuration details. The focus in the main chapters is on how to use **DFsetup** after you have determined the study specifications you want to enter. For additional information on CRF Design guidelines and visit map design see Chapter 9, *CRF Design Guidelines* and Chapter 8, *Subject Visit Scheduling*.

Chapter 1. Introduction

DFdiscover includes 3 primary tools:

1. **DFexplore** for study data management,
2. **DFsetup** for creating new study databases, and
3. **DFadmin** for user and study administration.

This guide explains how to use **DFsetup**. This introduction covers DFdiscover study configuration components, describes the steps required to create a new study database, and provides suggestions for testing your study setup. The next chapter, Using DFsetup - An Overview, provides an overview with screenshots of the main dialogs. This is followed by a chapter, Defining Data Fields, that explains how to create modules and define fields within them, optionally get study CRFs into **DFsetup** and assign the data fields on each unique page. The remainder of the guide explains each of the study setup options, in a separate chapter for each of the main menus: File Menu, Field Menu, Study Menu and View Menu.

1.1. Study Setup - Components

We start with a list of the components that make up a DFdiscover study database. Use this as a guide to identify the information needed before using **DFsetup** to create a new study.

CRFs

Study case report forms (CRFs) must be designed in a word processing or graphics program, saved as a PDF file, and then imported into **DFsetup** where modules and data fields have been defined.

The same CRFs used in a paper-based study to record data and fax or scan it to a DFdiscover server, also appear in the screens used to enter data and send it to the server over the internet. This dual purpose means that data entry screens do not need to be programmed, and that sites can easily switch between submitting data via paper CRFs and electronically via **DFexplore**. It also means that CRF designers need to realize that they are simultaneously designing the paper CRFs and data entry screens for the study. For example, if a hidden data field is to appear (for specified users only) in **DFexplore**, white space must appear on the paper CRF where this field will be located.

For a description of DFdiscover CRF design guidelines see CRF Design Guidelines.

Plates

In most studies some CRF pages are repeated at follow-up visits or on each occurrence of some event. In DFdiscover the term plate is used to refer to a unique CRF page consisting of a set of data fields and their layout on a CRF page. Each plate corresponds to a data entry screen in **DFexplore** and to a data table in the study database.

When CRFs are imported into **DFsetup** (see Import CRFs), any title and instruction pages are typically discarded and one copy of each unique plate is preserved. The copy of each plate that is imported is used to define the data fields on that plate, and also as the background for that plate in **DFexplore**.

If CRFs have different background types, such as translations, add the categories to the CRF Type Map before importing the CRFs.

Plates can also be created without importing CRF backgrounds, by designing an eCRF directly in **DFsetup**. See Add eCRF Plate for details.

Key Fields

Each CRF page must contain 3 key fields which together uniquely identify each data record in a DFdiscover study database.

1. **Plate**

   Each page or plate is assigned a unique number in the range 1-500. This key field always appears in the barcode at the top of the page.

2. **Visit**

   Each visit is assigned a unique number in the range `0-65535`. This key field may appear in the barcode at the top of the page, or as the first data entry field on the plate.

3. **Subject**

   Each study subject is assigned a unique identification number in the range `0-281474976710655`. This key field must always appear as a data entry field, immediately after the visit key field at the top of the page.

Styles

A style is one of the properties that must be specified for each data field. Styles include default specifications for all data field properties: type, name, description, legal values, coding, etc. Each of these properties may be locked in the style or left open so it can be changed at the field level.

DFdiscover includes simple styles for each of the supported data types: check, choice, number, string, date, time and visual analog scale (VAS), and allow users to define study specific styles. Creating study specific styles with default and fixed property values is highly recommended because: it makes entering field definitions easier, ensures that fields which should have the same properties do, and makes it much easier to modify properties when the need arises.

Styles are defined using the Styles dialog. See Styles for a complete description of styles and all data field properties.

Modules

Modules are a layer between plates and fields that allow grouping of fields. Each module has a unique name. Fields are defined within a module before mapping them to the plate layout. This work can be done before CRFs are imported. You can have as many modules as you like, but modules cannot contain other modules and fields cannot be part of more than one module. Modules can be used more than one time, even on different plates. Each instance of a module is assigned a module number that can be used to refer to a specific field on a plate with repeating modules. See Modules for a complete description.

Data Fields

DFdiscover supports the following data types:

- number - an integer or fixed decimal value
- date - any combination of year, month and day
- time - either hh:mm or hh:mm:ss format
- check - a single box which may be checked or unchecked
- choice - a list of 2 or more mutually exclusive options
- string - a text field
- VAS - visual analog scale

For a description of DFdiscover data types and limits see Data Types, and for CRF design guidelines for each field type see CRF Design Guidelines.

The properties of each data field are specified in the **DFsetup** Modules view. The location of each data field is specified in the **DFsetup** CRF window where the imported plates are displayed. A field is mapped to a plate by clicking its data entry boxes or dragging out a data entry widget to identify its location on the plate, and then selecting an unassigned field from a module.

DFdiscover supports copy and paste of individual fields and field groups both within and across CRF plates, and allows style, module, and field definitions to be imported from other studies.

See Defining Data Fields for instructions on defining modules and data fields.

Visits

Each study has different subject visit scheduling requirements. Some are simple with one series of visits from screening to a final follow-up for all subjects, while others have multiple visit cycles and conditions that determine whether a particular CRF page, visit, or cycle is required, that signal the end of one cycle and the beginning of another, or that mark the end of subject follow-up.

A schedule comprised of all required and optional study visits and repeating forms along with all required and optional plates that may be collected at each visit must be specified for each study. This defines the subject binders used in **DFexplore** and also allows DFdiscover to create subject schedules and generate queries for overdue visits and missing pages.

This user guide shows how to enter visit map specifications and explains all of the rules and describes all of the available options. To learn how to design a DFdiscover visit map see Subject Visit Scheduling. To learn how to enter your specifications see Visit Map.

Sites

Each participating clinical site responsible for subject recruitment and follow-up must be registered before data from that site can be entered into the study database.

Each entry identifies the site ID and name, the contact person who will receive Query Reports, the subject ID numbers belonging to the site, and other optional specifications all of which are fully described in this guide at Sites.

Subject Aliases

As described earlier in this section, each subject is assigned a unique numeric identifier. This identifier is an intrinsic, required database property of each data element related to a subject. In many studies, this numeric identifier is all that is needed for unique subject identification. In some studies however, a non-numeric, string, or formatted identifier is required to meet external requirements. For example, rather than having subject identifier `33002`, the subject may be known as `C01-3002-A`. In DFdiscover this is accomplished by defining `C01-3002-A` as the subject alias for the subject identifier `33002`. All internal, database management of the subject data is performed with the subject identifier. However for presentation purposes the subject alias may be preferred.

Definition of subject alias is a study option. Specifications for definition are described in this guide at Subject Alias Map.

Translations

Language translations for field Prompt, Description and Coding Labels properties can be entered in the Translations view.

Definition of translations is a study option. Specifications for definition are described in this guide at Translations.

Early Release

In this release, translations are available to API clients only and are not implemented elsewhere in DFdiscover, other than **DFsetup**.

Edit checks

Edit checks can be programmed to perform data consistency checks, use lookup tables, calculate total scores and unit conversions, generate queries, send someone an email message, log an event, etc. Edit checks are entered, checked and published using the Edit checks dialog described in Edit checks.

A description of the DFdiscover edit check programming language is beyond the scope of this guide, but is fully described in *Programmer Guide, Edit checks*.

Conditional Terminations

Follow-up for some subjects may terminate early within one visit cycle or for all visits in all cycles, being signaled by some event recorded in a database field. Termination triggers are specified in the conditional termination map as described in Conditional Terminations.

For a detailed explanation of conditional terminations see Subject Visit Scheduling.

Conditional Cycles

Some visit maps have more than one cycle of visits, each cycle being required, optional or conditional, and each starting with a new baseline visit. The conditions which make a cycle required, optional or unexpected are entered using the Conditional Cycles dialog as described in Conditional Cycles.

For a detailed explanation of conditional cycles see Subject Visit Scheduling.

Conditional Visits

Each visit is defined as required or optional in the visit map, but this may be changed by rules specified in the conditional visit map using the Conditional Visits dialog as described in Conditional Visits.

For a detailed explanation of conditional visits see Subject Visit Scheduling.

Conditional Plates

Each visit in the visit map may include a list of required and optional plates, but this may be changed by rules specified in the conditional plate map using the Conditional Plates dialog as described in Conditional Plates.

For a detailed explanation of conditional plates see Subject Visit Scheduling.

Page Map

Each data record in a study database is uniquely identified by 3 keys: the subject ID, visit number, and plate number. Queries are further identified by 4th and 5th keys, the data field to which they are attached and the query category. When Query Reports are sent to the clinical sites queries are identified by these keys. The subject ID is meaningful and the field number is automatically replaced in the Query Reports by the field description so it too is meaningful. The visit and plate numbers may not be very meaningful to the sites and thus need to be replaced with descriptive labels. This can be done using the Page Map dialog as described in Page Map.

In an EDC study, where the sites use **DFexplore** to enter data, a Page Map may not be required because **DFexplore** provides links between the queries and the relevant data fields which makes it easy for the site to review outstanding queries and jump to the relevant data fields to make corrections or enter an explanation. Page map labels also appear in **DFexplore** Data View where they replace the plate labels in the subject binder navigation list. In an EDC or paper-based study, where queries are sent to the sites in Query Reports, page map labels should be defined.

Missing Map A list of missing value codes and labels can be specified using the Missing Map dialog as described in Missing Value Codes. Missing value codes may be applied to fields for which the 'need' property has been defined as optional or required, but can not be applied to fields defined as essential.

Query Category Map DFdiscover includes a generic set of category codes for identifying queries and the problems that they identify. In most studies these are sufficient. However if more categories are needed, additional categories can be added with the Query Category Map dialog as described in Query Category Map.

Sort Map In Query Reports, queries are always sorted in ascending order by subject ID. Within subject the default sort order is ascending by visit number, and within visit, ascending by plate number. But this within subject sort order can be changed using the Sort Map dialog as described in Sort Map.

Lookup Tables Lookup tables are used in edit checks to retrieve information based on a specified string match. The name used to refer to each lookup table in edit checks, and the name of the lookup table file (stored on the DFdiscover server in the study lut directory) must be specified using the Lookup Tables Map dialog as described in Lookup Tables Map.

DFsetup also includes a lookup table editor which can be used to add, modify and delete entries in each of the study lookup tables, as described in Lookup Tables.

For a explanation of how lookup tables are used in edit checks see *Programmer Guide, Edit checks*.

Query Titles The titles used in the standard DFdiscover Query Reports created by **DF_QCreports** can be customized using the Query Titles dialog as described in Query Titles.

Query Covers A cover sheet can be included with each Query Report created by **DF_QCreports**. Different cover sheets can be specified for different clinical sites. Query Report cover sheets are entered using the Query Covers dialog as described in Query Covers.

Query Messages Messages can be included at the bottom of Query Report cover sheets. Different messages can be specified for different clinical sites. Query Report messages are entered using the Query Messages dialog as described in Query Messages.

Users interested in a level of detail that includes the underlying file formats for storing study setup specifications can consult *Programmer Guide, Study FilesDFdiscover Study Files*.

1.2. Study Setup - Step by Step

This section describes the steps needed to create a new study database, identifies dependencies between setup components, offers tips to help you avoid problems and make the process as efficient as possible, and describes what can and cannot be changed once a study is in production.

Procedure 1.1. Study Setup Steps

1. **Study Registration**

   Before anything can be done in **DFsetup** a DFdiscover administrator using **DFadmin** must:

   - register the study on the DFdiscover server specifying: a unique study number, a study name and a study home directory.

- create at least one study role with permission to use **DFsetup**
- grant this role to at least one user

These steps are described in *System Administrator Guide, StudyAddingAdding a New Study*.

2. **Study CRFs**

   This step is optional. Study CRFs are not required for studies where there is no analogous paper CRF. In such studies, eCRFs are defined directly with **DFsetup**.

   Study CRFs must be created in a word processing or graphics package, barcoded (if DFdiscover will receive faxed or scanned CRFs), and saved in PDF format for import into **DFsetup**. The CRFs are used to create the data entry screens used in **DFexplore** For CRF design guidelines see CRF Design Guidelines.

3. **Subject Scheduling**

   A visit map must be created for each study. The sequence of subject visits and repeating forms and the plates used at each visit define the subject CRF binders that appear in **DFexplore** and thus data entry cannot be performed until a visit map has been specified. The visit map refers to the visit and plate key fields that appear on each CRF page, thus CRF design and Visit Map design should go hand in hand. For a complete description of visit map components and rules see Subject Visit Scheduling.

4. **DFsetup Permissions & Locking**

   The **DFsetup** program is available for Windows 10, macOS 10.14 (Mojave) or later, and Linux, and can be used by anyone with permission to connect to the DFdiscover server where all study setup information is stored. What a user can do in **DFsetup** is determined by the permissions granted to their study role by a study administrator, using **DFadmin**. Some users may have view only permission, others may be able to define and modify specific setup components, and others may have full permission for all components.

   More than one user can access a study in **DFsetup** at a time but only one user at a time can edit each of the setup components. While one user has a configuration file open and locked, other users can open it in view only mode.

   Users can choose to login in view only mode, or to edit configuration files only - without being able to change data fields. By selecting the minimum access mode necessary to accomplish the task at hand users can leave locks free for other users who may need them. Thus, an understanding of how **DFsetup** permissions and locking work will help you cooperate with other users who share study setup responsibilities (see Access Modes).

5. **Global Settings**

   Global study specifications can be changed at any time, but should be set as one of the first things you do in **DFsetup** when defining a new study, because they include default values that have an impact on the definition of styles and data fields.

   See Global Settings.

6. **Define Modules**

   Modules are groups of fields. Before fields can be mapped to a CRF they must be defined within a module. Modules should contain variables that logically go together. For example, when recording adverse events all variables that are used to describe an event should be included in a module. If there is more than one event on the page, each event is an instance of a module.

   See Modules.

7. **Import CRFs**

   Once modules are defined, plates are created. The definition and layout of a plate can be based upon a paper CRF, or an eCRF.

   - For a paper CRF, a blank copy of each CRF plate must be imported. Plates can be imported individually or many plates can be imported at one time. Title and instruction pages can be discarded during import and users can select which plates to import and which to discard. It is possible to import a revised version of a CRF plate to replace an old one.

     The file to be imported must be a PDF and reside on the user's local computer. You can also import CRFs from the DFdiscover server if you have access to the file system where the CRF file is stored.

     See Import CRFs.

   - For an eCRF, there is no blank copy to import. Simply choose **Study** > **Add eCRF Plate** and assign a plate number.

8. **Import Definitions**

   If a plate already exists in another study all of the style and module definitions can be imported from that study. The import is just a copy that can then be modified in the new study. Before defining new data modules you should always consider whether it is possible to reuse work that has already been performed in another study.

   You might also want to consider creating a special DFdiscover database to store predefined versions of frequently used modules or plates. Such a CRF Library database can provide one standard location where such modules or plates can be registered and tested, and where any necessary modifications can be maintained over time.

   See Import Definitions.

9. **Plates**

   Use this dialog to enter a descriptive label and other plate properties for each unique plate in the study. If plate definitions have been imported from another study the plate descriptive label and plate properties will be imported as well, and can then be modified in the Plates dialog if necessary.

   See Plates.

10. **Data Styles**

    Every data field definition begins with the selection of a style that specifies the data type (number, string, date, check, choice and VAS) and other properties. These properties can be locked in the style so they cannot be modified at the field level, or left as default values which can be over-ridden when a field is defined.

    DFdiscover includes a number of Simple styles that should be reviewed and modified as needed for the study at hand. For example the SimpleDate style's format and legal range should always be modified as appropriate for the study.

    You should also review the CRFs and decide what study specific styles are needed. Common examples include:

    - a YesNo style used to lock the coding for data fields consisting of Yes / No choice options.
    - a StudyDate style used to lock the format and legal range for events occurring after the study start date.
    - a SubjectID style used to lock all properties of the subject identification field that appears on each CRF page.

    Time spent creating a complete set of styles will both simplify data field definitions and ensure that properties that should be consistent across a set of data fields cannot be entered incorrectly when defining individual fields.

    See Styles.

11. **Annotated CRFs**

    It may be helpful to create annotated CRFs before beginning to define data fields. One version of the CRFs might be annotated with field numbers and style names, while another might be used for field level properties like field names.

    If you do this remember to number the visit key field as field 6 if it is not in the barcode, and that subject ID must be field 7 on every CRF plate. DFdiscover includes 3 header fields you do not need to define: status, workflow level, and primary image, and 2 which must be in the barcode: study number and plate number, so the first 5 fields are always predefined.

12. **Data Fields**

    The definition of a data field requires:

    - first locating the field on the CRF page,
    - then selecting the appropriate module and field.

    This process must be repeated for each data field on each plate, but copy and paste can be used to reduce some of the effort. For example, the subject ID field must appear at the top of every CRF plate, and in some studies this is always followed by subject initials and the visit date. Such common fields can be defined on the first plate and then copied and pasted on all other plates.

    Some CRF plates may be comprised of a repeating blocks of fields, e.g. allowing data on several medications or medical history items to be recorded on the same page. Here too copy and past can reduce the effort.

    See Defining Data Fields.

13. **Sites**

    Each clinical site that contributes subject data must be registered using the Sites dialog. A site will not appear in **DFexplore** and thus data entry cannot be performed for that site until it has been registered.

    The required fields for each site include:

    - a unique site ID in the range 0-21460
    - name of the site coordinator or chief contact person
    - name of the clinic, hospital or institution
    - fax number(s) and/or email address(es) to which Query Reports are sent
    - list of subject ID numbers and ranges used by the site

    All other site properties are optional but a "Replyto" email address is highly recommended if Query Reports will be emailed to the clinical sites.

    The site properties, including the subject IDs that belong to a site, can be changed at any time during the study. If a subject ID is moved from one site to another (e.g. because the subject moved) all subject data and metadata records will appear under the new site in **DFexplore** and any outstanding queries will be included in the next Query Report sent to the new site.

    See Sites.

14. **Subject Aliases**

    This is an optional step. If subjects are identified externally by a key that is not a numeric identifier, those keys will be defined in DFdiscover as aliases for the unique numeric subject IDs. Internally, numeric values are always required for subject IDs. In site definition, for example, numeric subject IDs are always used.

    Once created for a subject, their numeric identifier will not change for the duration of the study. It is possible however for the alias to change, so long as the change is from a unique alias to another unique alias. Sites and subjects may find this confusing so it is best practice to define subject aliases, if they are being used, at the outset of the study and thereafter not change them.

    See Subject Alias Map.

15. **Translations**

    This is an optional step. Multiple language translations for field Prompt, Description and Coding Labels properties can be entered in Translations view.

    See Translations.

16. **Visit Scheduling**

    The visit map determines how subject binders are organized and displayed in **DFexplore** and thus data entry cannot begin for any subject until a visit map has been specified.

    The visit map includes all scheduled and unscheduled subject visits and repeating forms, when they are due and overdue, and which CRF pages are required and optional for each visit. In addition, conditional terminations, cycles, visits and plates may be specified in the corresponding conditional maps. Visit scheduling can be very simple or quite complicated and DFdiscover supports many options to handle the needs of different studies.

    CRF and visit map design should be completed together and particular attention should be paid to the selection of visit and plate numbering. While a visit map can always be modified during a study to add visits or plates, changing the numbering of the plate and visit key fields is not possible as this would conflict with data and audit trail records that have already been created.

    DF_QCupdate uses the visit and conditional maps to identify overdue, missing and unexpected visits and pages, thus this function also depends on these specifications.

    It is not uncommon for changes to be made in the visit and conditional map specifications after a study has started. The new rules will be applied the next time **DF_QCupdate** is run, and queries for overdue visits and missing pages that are no longer relevant will be removed.

    See Subject Visit Scheduling for information on designing a visit map and any required conditional maps, and the following sections for information on how to enter your specifications:

    - Visit Map
    - Conditional Terminations
    - Conditional Cycles
    - Conditional Visits
    - Conditional Plates

    Documentation on **DF_QCupdate** can be found in *Standard Reports Guide, DF_QCupdate*.

17. **Page Map**

    The page map is an optional configuration file used to enter labels for the plate and visit numbers associated with each query on the Query Reports sent to the clinical sites. The page map can also be used to configure plate labels based on the visit number as displayed in the subject binder in **DFexplore**.

    In an EDC study where the sites use **DFexplore** to enter data and respond to queries and Query Reports are not used, a page map is not necessary, but in a paper based study where Query Reports will be delivered by fax or email, the page map should be completed before the first Query Report is sent.

    See Page Map.

18. **Missing Value Codes**

    The missing value map is an optional configuration file, used to register study sanctioned missing value codes and labels. In **DFexplore** these codes can be assigned to any optional or required data field, and can be thought of as pre-approved reasons for missing data. Data fields defined as 'essential' do not accept missing value codes.

    If the missing values configuration file does not exist for a study **DFexplore** uses '*' as the one and only default missing value code. If the missing values configuration file exists but is empty then no missing value codes are allowed for any data field in the study.

    New missing value codes can be added during a study but you should not remove or reassign missing value codes that are in use, otherwise you could render some data fields uninterpretable, and invalid. There is no harm in updating the label on existing missing value codes to correct mistakes or poor wording, as long as you do not change the meaning. It is only the code that is stored in the study database. Thus any change in the label will apply to both past and future use of that missing value code.

    See Missing Value Codes.

19. **Edit checks**

    Edit checks are optional but extremely useful. They can be programmed and tested as soon as the data fields have been defined on the relevant plates. They can be added and modified as needed during a study.

    See Edit checks for a description of the **DFsetup** edit check editor.

20. **Lookup Tables**

    Lookup tables are optional. They can be defined for standard queries and reasons, and for use in edit checks. They can be modified as needed during a study.

    See Lookup Tables for information on how to create lookup tables, and Lookup Tables Map to register lookup tables for use in edit checks.

21. **Query Category Map**

    DFdiscover provides a basic set of query categories that can be assigned to queries. The optional query category map can be defined to extend the available categories, providing different criteria, perhaps study specific, to identify and group queries.

22. **Query Sort Map**

    The sort map is an optional file used by **DF_QCreports** to determine the order in which queries are displayed on Query Reports. It is not used by **DFexplore**. It can be changed at any time and will only affect subsequent Query Reports.

    See Sort Map.

23. **Customizing Query Reports**

    The titles used on the standard Query Reports created by **DF_QCreports** can be customized and cover sheets and messages can be added and customized for different clinical sites. This is of course optional and can be added or changed at any time during the study.

    See

    - Query Titles
    - Query Covers
    - Query Messages

1.3. Study Setup - Testing

Each study setup should be tested before it is put into production for data collection. This section provides some suggestions on what to test and when.

Procedure 1.2. Plate Testing

Plate testing can begin as soon as the first plate is defined. Testing each plate after it is defined may detect problems to be corrected and avoided in the remaining study plates.

The first three tests below are performed in **DFsetup**, while the remaining tests are done in **DFexplore**. Before plate testing can begin in **DFexplore**, the visit map must be defined with at least one visit with the plate to be tested (see Visit Map), and the Sites definition must include at least one test site (see Sites).

1. **Field Order**

   Examine the field number of each data entry widget to verify that they are in the desired order. If the visit number is a defined field on the plate (rather than included in the barcode or predefined in the visit map), it will be field number 6 and the subject ID will be field number 7. If the subject ID is field number 6, the plate property Sequence is not set correctly. Correct it using the Plates dialog as described in Plates.

2. **Field List**

   In **DFsetup** switch from CRF View to Field List View using the button at the bottom of the screen. Scan down the columns to review the same property quickly across all fields on the plate. In addition to spotting field level problems, this may also suggest changes needed to the styles or module fields.

3. **Verify All**

   Select **File** > **Verify All** in **DFsetup** to run all of the DFdiscover study setup integrity-check reports. These reports can also be run individually in the **DFexplore** Reports view. For more information select **View** > **Reports** in **DFexplore**, select each report that begins with 'DF_IC' and click Explain. See *Standard Reports Guide, Alphabetical Listing, Legacy Reports* for further details on these reports.

4. **Field Colors**

   Open the plate in **DFexplore** and examine the color-coding of all blank fields. Required and essential fields are red. This step might identify fields that have been misclassified.

5. **Field Traversal**

   Starting with the first field, tab through each field on the plate to confirm that the cursor moves through the fields in the expected order and that any skip specifications work as expected.

6. **Essential Fields**

   Essential fields are required for a data record to be saved; neither a blank value nor a missing value code is acceptable. The visit and subject ID key fields fall in this category. Confirm Missing Value in the bottom left corner of the screen is inactive for all essential fields.

7. **Missing Values**

   Verify that all missing value codes needed for the study appear in the list.

8. **Data Values**

   Test each field by entering both legal and illegal values. Fields turn white on entering a legal value and red on entering an illegal value. This might identify changes needed to the legal value property of styles or fields, and will also allow verification that each field has the correct format and length.

9. **Edit checks**

   If edit checks have been defined, enter test values to verify edit check behavior. A separate test plan is recommended for each edit check, comprised of as many test cases as required to test each combination of data values that is expected to produce different edit check behavior.

10. **ICR**

    If data will be collected on paper CRFs, submit a few example pages to test intelligent character recognition (ICR). Numbers, dates, choice fields, check boxes, and visual analog scales should be read correctly, provided the fields are legible. ICR errors may be due to unusual or unclear handwriting, but consistent ICR errors when the values look readable may indicate a problem with CRF design, the legal range specifications, or the location of data entry widgets in **DFsetup**.

Procedure 1.3. Visit Schedule Testing

1. **Is It Complete**

   The visit map defines the subject binders used by **DFexplore**. All visits and plates must be included in the visit map in order to appear in **DFexplore**.

2. **DF_ICvisitmap**

   This report looks for problems in the visit map, conditional maps, and VisitDate fields. It can be run from the **DFexplore** Reports View or by selecting Verify on the Visit Map dialog.

3. **Conditional Maps**

   Enter one or more test cases in **DFexplore** for each condition specified in each of the conditional maps. Then run **DF_QCupdate** to apply the conditional rules. Check for overdue visit and missing plate queries using the **DFexplore** Query View, and check for unexpected visits and plates by running the **DF_PTunexpected** report. The **DF_PTvisits** report lists the status of all cycles and visits for each subject.

Procedure 1.4. Query Report Testing

1. **Query Titles, Covers, and Messages**

   Test these specifications by running **DF_QCreports** to create a Query Report for each site and then using **DF_QCview** to review each report. If different sites have different cover sheets specified, include test cases for each site. Note that a Query Report is only created for sites that have enrolled study subjects.

2. **Subject Scheduling**

   The standard subject schedule section of Query Reports includes the date of enrollment, last follow-up and next scheduled follow-up. Enter test cases, then run **DF_QCupdate** to update `DFX_schedule`. Then create and review Query Reports to see if the expected scheduling is reported.

   Run **DF_PTvisits** to examine the entire visit schedule for each subject, including the dates of all completed and scheduled visits. This report also provides information on conditional events and terminations and is a good way to test different subject follow-up scenarios.

3. **Page Map**

   Test the labels specified for visit/plate combinations by reviewing the subject binder labels in **DFexplore** or by adding an unresolved query to the data record, running **DF_QCreports** and then **DF_QCview** to examine the query labels.

4. **Sort Map**

   If the sort map is defined to reorder queries within a subject, examine the query order on Query Reports to confirm they are sorted correctly. Without a sort map the default order is by plate number within visit number (both ascending).

5. **Query Report Format**

   Query Reports come in 2 different formats depending on whether field descriptions are specified as short (25 character) or long (40 character) in the Study Global settings dialog. Short descriptions result in a more compact report.

   Changing the global setting and rerunning **DF_QCreports** will show the difference. See Global Settings for further information on this setting.

1.3.1. User Permissions

Before a study goes into production, test each of the roles defined for the study to verify the expected user permissions for:

1. Tools
2. Data Records
3. Reports
4. Image Router
5. Study Administration
6. Database Permissions

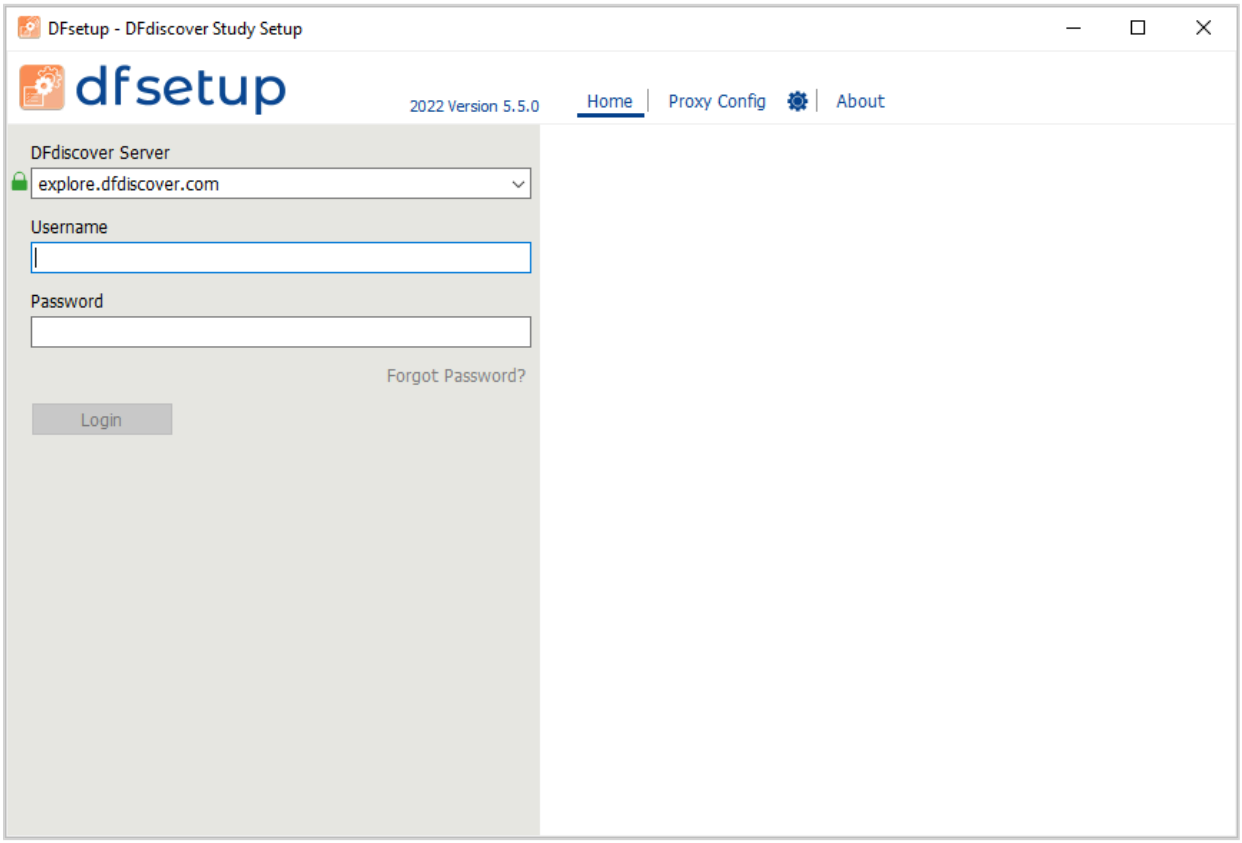
Chapter 2. Using DFsetup - An Overview

This chapter covers login, generic features and provides an introduction to the main window used to display CRF plates and define data fields.

2.1. Login

DFsetup is available for Windows 10, macOS 10.14 (Mojave) or later, and Linux.

Starting **DFsetup** displays the login screen. Enter the value of the **DFdiscover Server** provided by the study coordinating site. If the system administrator has defined a customized login screen, the contents of the right-side panel refresh to show this screen. A green lock icon also appears next to the **DFdiscover Server** field. The icon is confirmation that a secure connection with the server has been established. Optionally, click the icon and review the details of the connection.



Optional: In some configurations, your local IT infrastructure may require a proxy server for applications that connect to the internet. Your local IT department will supply the necessary information. To configure local proxy server settings, click **Proxy Config** at the top of the login screen. Complete the required fields in the sub-dialog. Click **Home** to return.

Your **Username** and **Password** will be the same for all DFdiscover programs (**DFsetup**, **DFexplore** and **DFsend**) that you have permission to use on the specified server, but may differ across servers. Enter them in the matching fields and click Login.

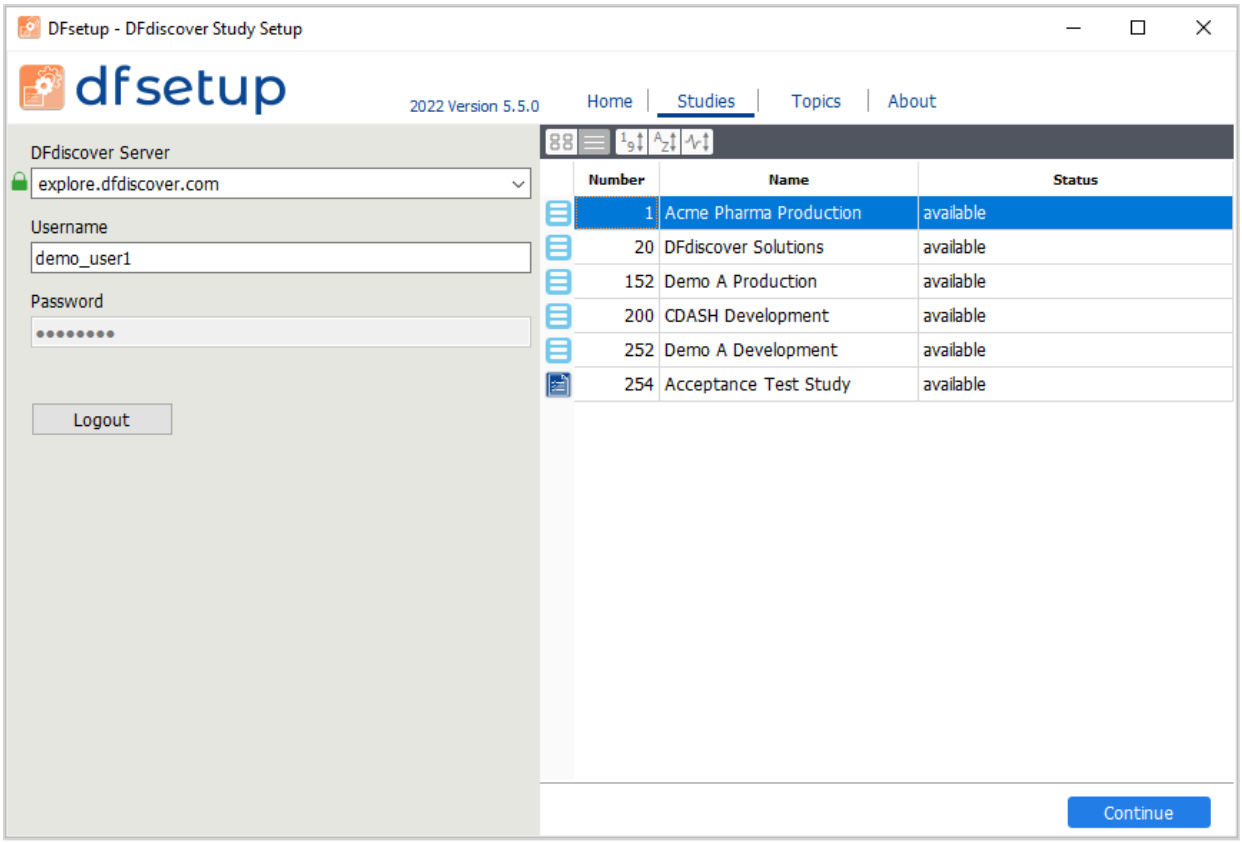
The administrator of the DFdiscover Server may have enabled two-factor authentication for your login account. If it is enabled, after successful authentication the login screen updates to request the security code. Independently you will receive an email with the 6-digit security code. The email is sent to the email address on file for the account. Enter the 6-digit security

code in the field labeled **Please enter the security code that we just sent you**. After entering the code, click Login again to complete login.

DFdiscover includes password aging, an FDA regulatory requirement. Password expiry may be any time between 1-9999 days and is set by the DFdiscover Administrator in **DFadmin**. After the expiry period has elapsed you will be prompted to reset your password the next time you login. To reset your password, enter the server name and your login name into the login dialog, then click Help and select **Password Reset**. You are asked for your email address, which must match the email address for you on this server. A single-use password will be emailed to the address, which can then be used to complete the password reset operation.[1]

In the login dialog, About can be used to access version information about **DFsetup**. After successful login, the **DFsetup** user guide may be accessed from **Topics** in the same login dialog or **Help** > **Topics** from the application menu.

If login is successful you will see the study selection dialog.



- Each study database is identified by a number and a name.
- Only studies for which you have been granted some level of access will be listed.
- Select a study and click OK to create, edit or review the study setup information.
- This dialog will timeout if a study is not selected within 60 seconds.

2.2. Setup Version

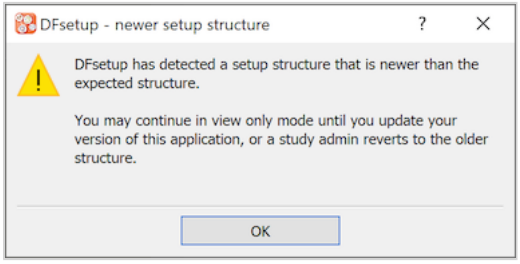
The properties of a study setup adhere to a specification that lists all of them, what attributes and values they have, and how they are related. This specification is internally identified with a setup version that is defined by DF/Net Research, Inc..

[1]There is a timer in the Password Reset dialog, which closes the dialog after 60 seconds of inactivity for **DFexplore**, **DFsend** and **DFsetup**.

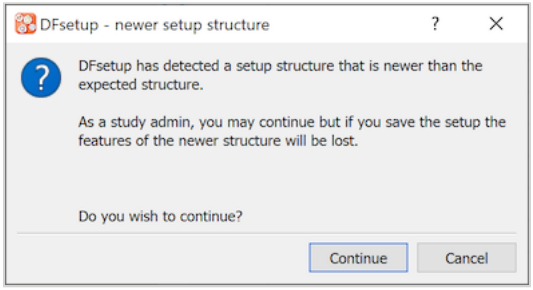
Users will likely never know this exists. The exception to this occurs when DF/Net Research, Inc. changes the specification. This may occur from time to time when new properties are added to a study definition (a new data type for example). Note that these are not properties that a user would define as part of study setup and during their daily use. Instead these are properties that apply to every study definition at a structural level.

When such changes are necessary, they are always introduced in a non-destructive way so that new application versions and study definitions take advantage of them in a seamless way. It may however be difficult to achieve a seamless result when a new study definition is opened with an older version of the **DFsetup** application. For example, the new study definition may include a property that the older version application does not recognize. To ensure that new properties are not accidentally deleted, **DFsetup** detects when an older version of **DFsetup** is being used to read a new version of a study setup definition, and it presents one of two options:

1. For the standard user, changes are not permitted and the user is able to view the existing setup definition.



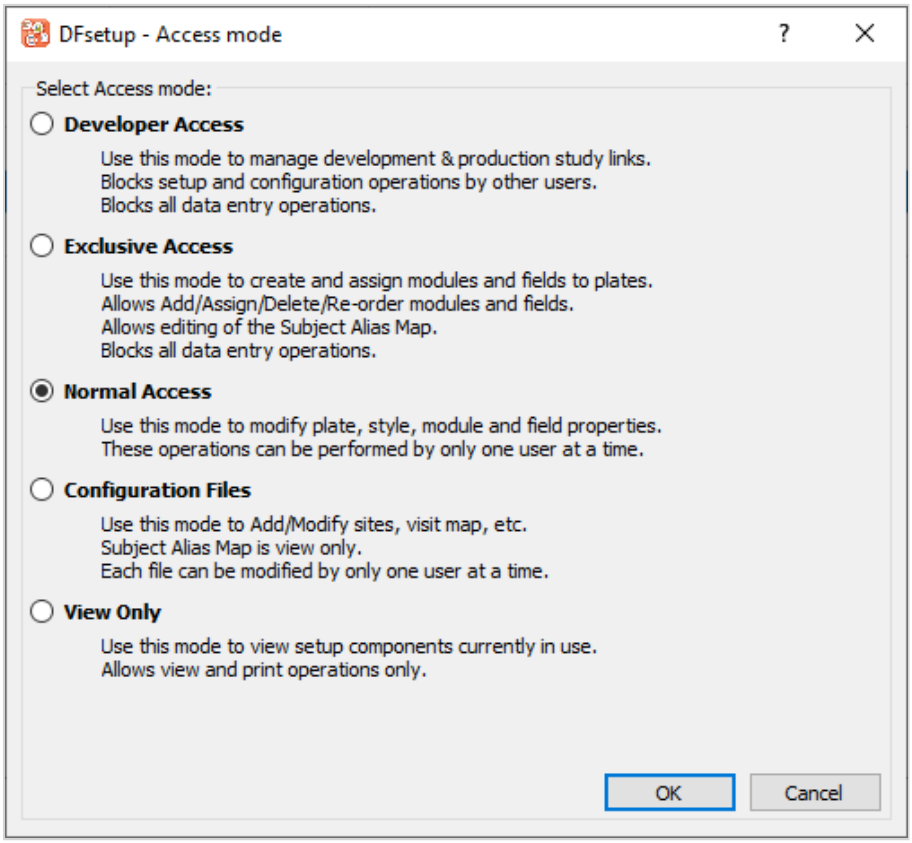
2. For the admin user, changes are permitted and the user is able to edit and save the existing setup definition. In this case the version of the setup definition is "downgraded" to the version that matches the older **DFsetup** application. Any changes that were made to include new properties will be lost when the setup definition is saved.



The preferable solution is for all users to stay up-to-date with their **DFdiscover** applications. When this is true, the version mismatch can never occur and the user will never see these dialogs.

2.3. Access Modes

If the study is available and you have permission to access any of the study setup components, the access mode dialog appears.



- The modes are ordered, bottom-to-top, by increasing complexity and permission. **View Only** places no restrictions on access by others while **Developer Access** requires exclusive use.
- The first 3 modes can be used by only one person at a time.
- Some modes may be unavailable because they are not permitted or the mode is already in use.
- To avoid blocking other users, select the least restrictive mode that is compatible with your objectives.

DFsetup provides 5 access modes. The modes, as listed in the dialog, are presented from most capable at the top to least capable at the bottom. A study role may not include permission to use all of the access modes. Select the lowest mode needed to accomplish the task at hand, to avoid blocking other users.

The following is a description of each mode that includes intended use, user permission requirements, and the restrictions it imposes on other study setup and database functions.

Developer Access	
Use	• create a development-production link between 2 study databases. • publish setup changes from the development study to the production study. • review differences between the development and production study setups. • revert the development setup back to the current production version. • break the link between development and production study databases. For a full description of how development - production study links are created and used refer to Link.

Permissions	• this mode can be used by user `datafax` and study administrators only. • users must have 'Setup-Plates' permission.
Restrictions	• only one user can have Developer Access mode at a time. • this mode will be unavailable if any other user is already in Developer, Exclusive or Normal access mode. • no **DFexplore**, **DFimport.rpc** or **DFbatch** logins can be active.
Exclusive Access	
Use	• adding new plates and data fields to a study setup. • revising data field layout on existing plates by adding, inserting, deleting or reordering data fields. • editing the aliases that may be in use and mapped to numeric Subject IDs.
Permissions	• 'Setup-Plates'.
Restrictions	• only one user can have Exclusive Access mode at a time. • this mode will be unavailable if any other user is already in Developer, Exclusive or Normal access mode. • no **DFexplore**, **DFimport.rpc** or **DFbatch** logins can be active.
Normal Access	
Use	• create and modify styles. • modify existing field properties. • add and test edit checks.
Permissions	• 'Setup-Plates'.
Restrictions	• only one user can have Normal Access mode at a time. • this mode will be unavailable if any other user is already in Developer, Exclusive or Normal access mode. • Normal mode may be used while there are active **DFexplore** logins. Any saved setup changes only become active for new **DFexplore** logins.
Configuration Files	
Use	• create and modify the configuration files (sites, visit map, edit checks, etc.) available under the **View** menu.
Permissions	• separate permission is granted for each configuration file in the study roles.
Restrictions	• this mode can be used by more than one user at a time, but each configuration file can be opened by only one user at a time. • editing of the Subject Alias Map is not permitted. • this mode can be used while another user is in Exclusive or Normal Access mode. • this mode will be unavailable if any user is in Developer Access mode.
View Only	

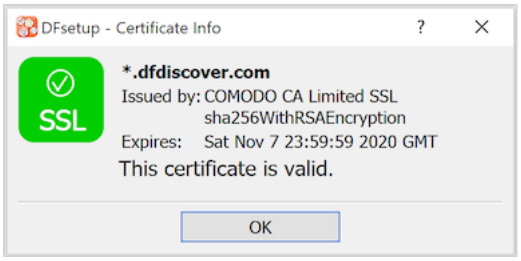
Use	• open and view all study setup components.
Permissions	• 'Setup-View'.
Restrictions	• none.

2.4. Certificate Info

DFsetup communicates with the DFdiscover server using HTTPS on port 443. This port must be open on any firewalls between the local computer and the study server.

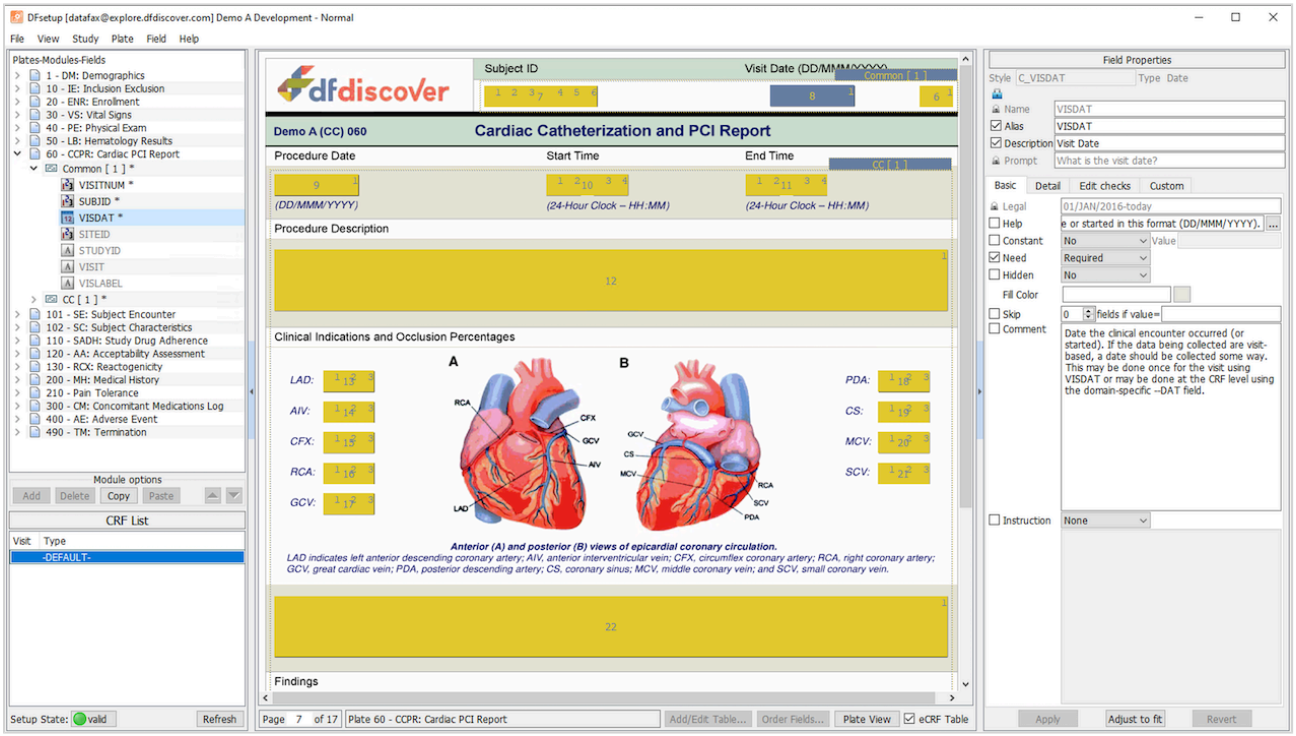
This is industry-standard technology that encrypts the bi-directional communication using a 'certificate of trust' provided by the server. It is the same technology used by banks and the majority of secure, global web services.

You can visually confirm that the communication is secure. Select **Help** > **Certificate Info** and look for the green checkmark.



2.5. Main Window - CRFs

If your login was successful the main window will be displayed.



The main **DFsetup** window has 3 major components, which are from left to right:

- a plate-module-field navigation window which lists the CRF plates that have been imported or added as eCRFs, each of which can be opened to show the list of modules and data fields defined on that plate,
- the CRF image window which shows the CRF plates that have been imported or added as eCRFs and the modules and data fields that have been defined for data entry, and
- the field properties window which shows the properties that have been specified for the current field (highlighted in the CRF image window). When a change is made Apply and Revert at the bottom of this window become active. Select Revert to undo all changes made to the field since the last Apply, and select Apply to save changes to your computer memory. Changes to field properties are not saved to the study server until you select **File** > **Save**.

The main window title bar shows: the user, DFdiscover server and study names, and whether there are any unsaved changes to the study setup.

The main window footer always includes the current page number, plate label, the Add/Edit Table button, the Order Fields button, a button used to switch between the Plate View, eCRF Preview, and Field List views, and a **eCRF Table** checkbox. Plate View with eCRF Table checked is the default view. When you switch to a new view, that view is remembered across plates and sessions.

On the left side is the Setup State button. This button displays a colored icon of yellow or red if setup problems are detected and green if setup is valid (no problems). If problems are present, a problem count will also be displayed to the right of the icon. Clicking on this button will open a window at the bottom of the screen listing each problem type and details. If no problems are present, the window will open but be empty. Clicking again will hide the window.

When problems are detected, the Setup State icon displays the status of the most critical problem in either yellow or red. The setup problem window displays an entry for each problem with its respective type (Warning or Error) and details. When both Errors and Warnings exist in setup, the Errors are always displayed first.

Problems are displayed in the Setup State window with their corresponding problem type, plate and, if applicable, field number. A brief description of the problem follows. Double-clicking any problem entry automatically takes the user to the problem plate. If the problem exists for a specific field, that field becomes highlighted/active on the current plate.

It is possible to resolve each problem while the problem window remains open. Each time a problem is resolved, the problem entry disappears from the list and the problem count drops. The Setup State window is automatically updated. It is not necessary to save setup in order to update the Setup State window/problem count.

2.5.1. Modifying the appearance with keyboard shortcuts

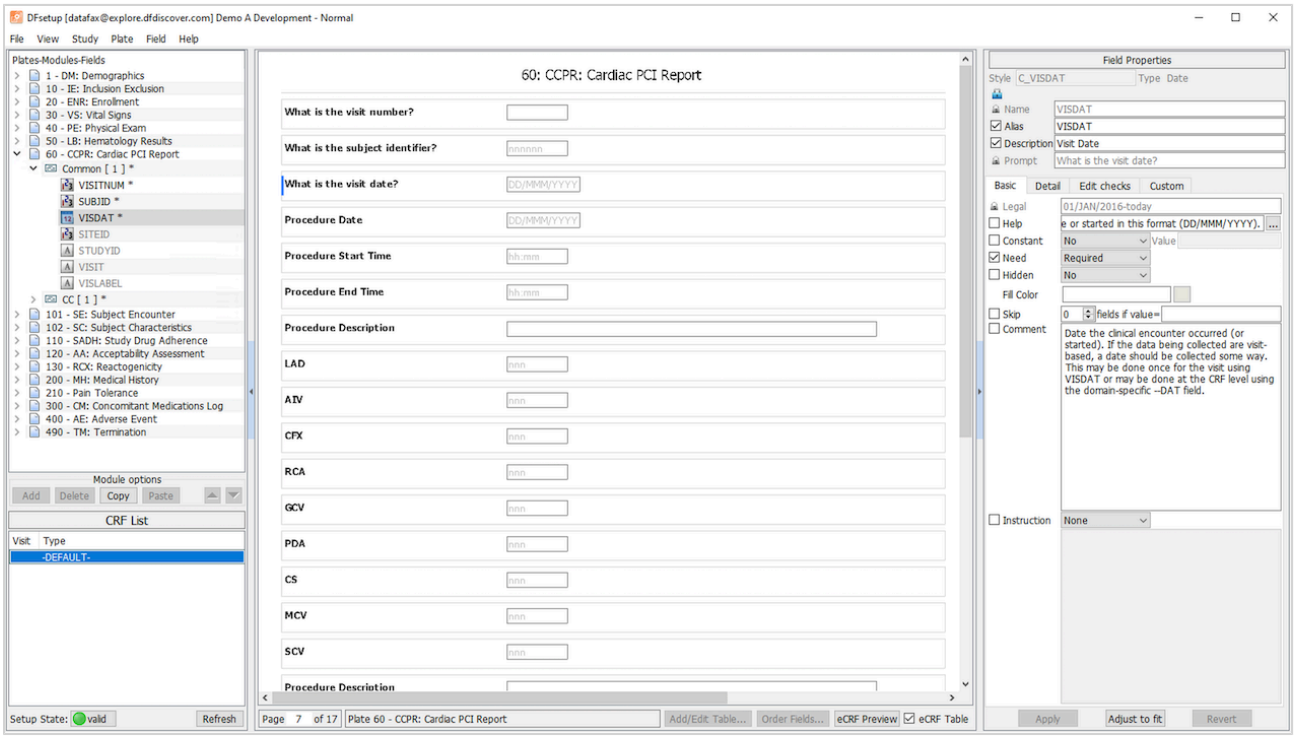
It is possible to change the visibility of features in the main window via several keyboard shortcuts, in addition to using the Plate View and eCRF Preview options. For each of these keys, the appearance is changed only while the key is pressed; the original appearance is restored as soon as the key is released.

- H hides the data field widgets to show the underlying CRF image.
- R changes the field widgets to show the 'need' property for all fields on the current page: solid = essential, stippled = required, and outlined = optional.
- M hides all fields except those belonging to same module instance as current field.
- I hides all modules except those that are instances of the same root module as current module.
- C hides all module names from CRF view.

2.6. Main Window - eCRF Preview

Selecting the eCRF Preview option from the Plate View button at the bottom of the main window switches the center window to a preview of the current CRF plate as an eCRF data entry screen.

This preview shows a CRF without a background as it would appear in **DFexplore** and an approximation of how it would appear in **DFweb** and **DFcollect**. For CRFs with a background, this preview shows an approximation of how the CRF would appear in **DFcollect** (with all screens included on one page). With **eCRF Table** checked, any tables defined on the plate will be displayed. With **eCRF Table** unchecked, any tables defined will not be displayed, and the preview will show an approximation of how the eCRF appears in **DFweb** and **DFcollect** where tables are not supported.

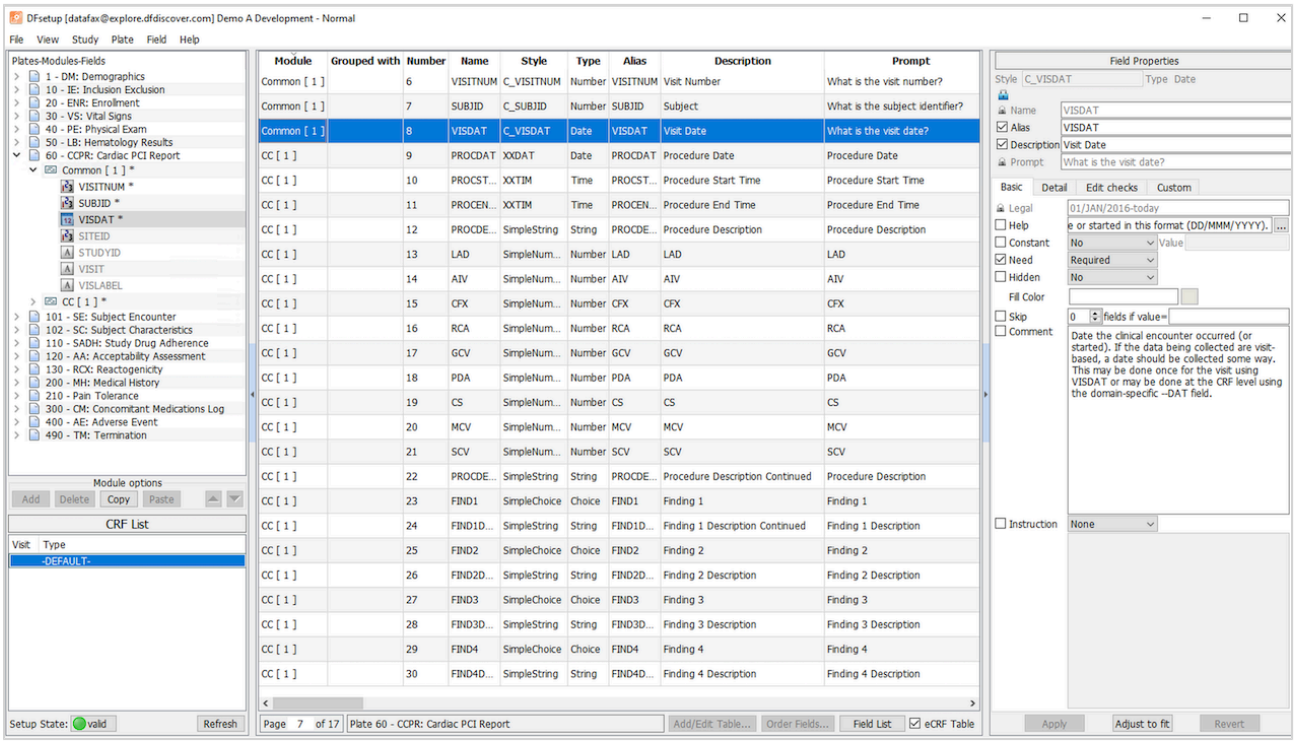


This view is useful for reviewing and comparing the appearance of CRF plates across different DFdiscover data collection and management applications.

Click any field in the center window to put that data field in the **Field Properties** panel where changes can be made.

2.7. Main Window - Field List

Selecting Field List at the bottom of the main window toggles the center window between the CRF view (shown above) where the focus is on 1 data field at a time, and the Field List view (shown below) where the field properties of all data fields defined on the current plate can be reviewed.



This view is useful for reviewing and comparing data field properties.

Click any column name to sort the table on that column.

Right-click any column name and then select **Show/Hide columns** to select the field properties to be displayed.

The cells of the spreadsheet are view only. Clicking anywhere in a row puts that data field in the **Field Properties** panel where changes can be made.

2.8. Entering Study Setup Specifications

Data fields are defined in the main window (Main Window - CRFs). This is fully described in Defining Data Fields. All of the other setup components: visit map, missing value codes, edit checks, etc., are defined using separate dialogs available from the **View** menu, as described in View Menu.

2.9. Menus

This section lists all menu options available from the menubar at the top of the main window.

File Menu

- Preferences - **DFsetup** user preferences
- Link - manage development-production study links
- Print - print annotated CRFs, style, module and field properties
- Save as PDF - save annotated CRFs, style, module and field properties as PDF files
- Save - save all changes to the DFdiscover server

- Export Setup - export study setup as Excel, JSON or XML file
- Verify All - check setup specifications for obvious errors
- Review Changes - show unsaved changes made to field specifications
- New Study - start another **DFsetup** session
- Close Study - close connection to the current study
- Exit - quit **DFsetup**

View Menu

- Modules - module and field definitions
- Plates - plate specifications
- Styles - create and manage data styles
- Edit checks - edit check editor
- Lookup Tables - lookup table editor
- Sites - specify clinical sites
- Missing Value Codes - specify missing value codes
- Query Category Map - design custom query categories
- Sort Map - set query sort order for Query Reports
- Lookup Tables Map - lookup table map editor
- Visit Map - subject scheduling and CRF plate requirements
- Page Map - plate/visit labels for Query Reports
- CRF Type Map - create CRF types to categorize CRFs
- CRF Background Map - specify which visits share the same background
- Conditional Terminations - rules for ending subject follow-up
- Conditional Cycles - specify cycle requirement rules
- Conditional Visits - specify visit requirement rules
- Conditional Plates - specify plate requirement rules
- Query Titles - titles for Query Reports
- Query Covers - cover sheets for Query Reports
- Query Messages - messages for Query Report cover sheets

Study Menu

- Global Settings - general study specifications

- Import CRFs - import blank CRF plates from PDF/PS files
- Import Definitions - import style and field definitions from other studies
- Add eCRF Plate - add a new eCRF plate
- Modify Plate Number - change plate number for current CRF page
- Delete Plate - remove current plate definitions

Field Menu

- Copy - copy selected data fields
- Paste - paste copied data fields to create new fields
- Delete - remove data fields
- Select All - select all data fields
- Order - reorder data fields on the current plate
- Group - group fields on the current plate
- Ungroup - ungroup selected fields

Chapter 3. Defining Data Fields

3.1. Preparation

The following steps need to be completed before starting to define new data fields in **DFsetup**.

1. **Global Settings**

   Global study specifications have an impact on field definitions and thus need to be considered before defining fields.

2. **Import CRFs**

   The CRF plates on which the data fields are defined need to be brought into **DFsetup**. Alternatively, if data is collected via EDC then eCRF plates can also be added using **Study** > **Add eCRF Plate**.

3. **Import Definitions**

   When possible, importing style, module and field definitions from other studies, or a special CRF library database, saves a lot of work. All you require is view permission for the source study setup.

4. **Plates**

   A label and other properties need to be specified for each imported plate.

5. **Styles**

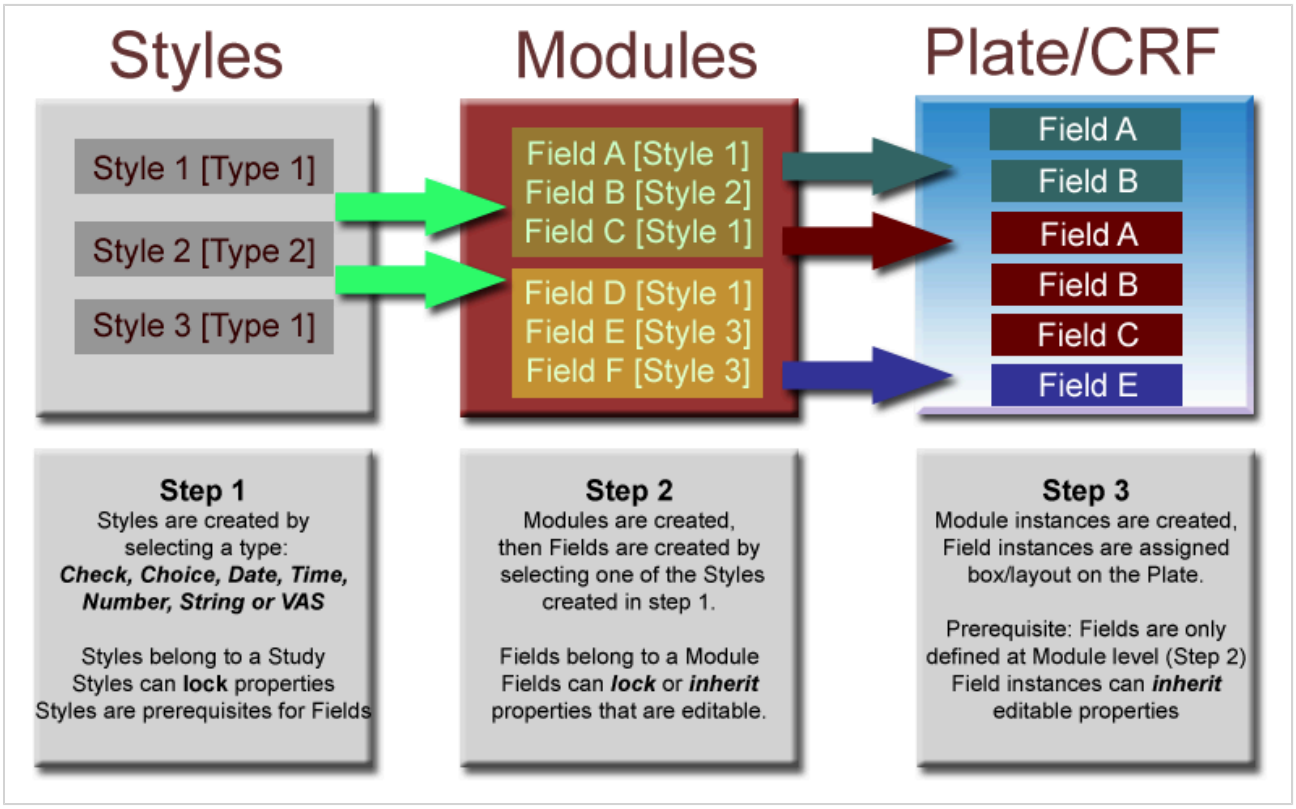
   Every data field needs a style.

6. **Modules**

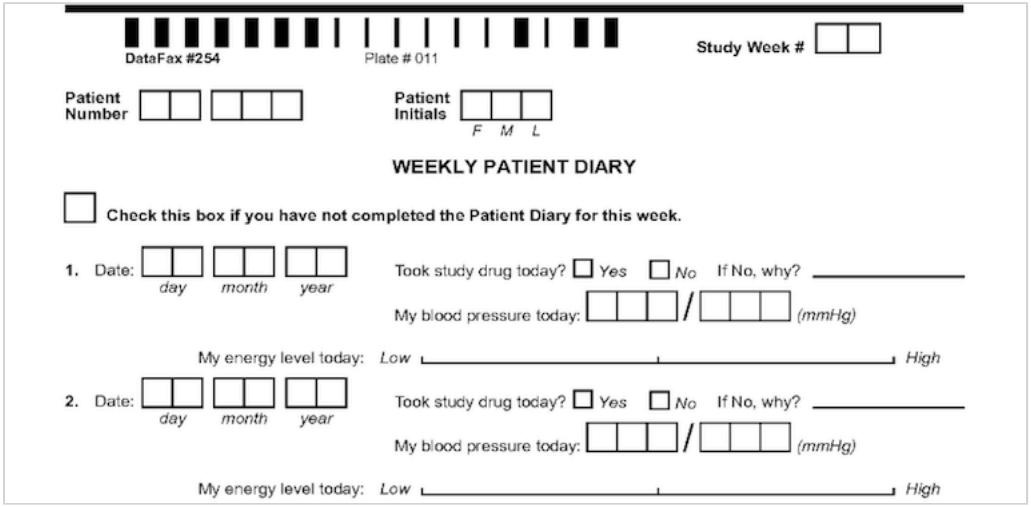
   All fields must be defined within a module before mapping them to a plate.

3.2. Module and Field Definition

All fields on a plate require a definition within a module or modules. Each field is derived from a style that describes the common properties for fields of any given type or usage. Modules are created and edited using **View** > **Modules** from the main menu. The following diagram provides an overview of styles, fields and modules.



Modules are groups of fields that may repeat on a plate or plates. The following example shows a CRF window that contains 10 unique data fields out of which 6 diary fields repeat for each day of the week. Thus, a module for 6 diary fields can be created and it's instances are added to this plate, mapped to each group of diary fields.


DataFax #254 Plate # 011 Study Week #

Patient Number Patient Initials F M L

WEEKLY PATIENT DIARY

Check this box if you have not completed the Patient Diary for this week.

1. Date: day month year Took study drug today? Yes No If No, why? _____
My blood pressure today: / (mmHg)
My energy level today: Low High

2. Date: day month year Took study drug today? Yes No If No, why? _____
My blood pressure today: / (mmHg)
My energy level today: Low High

3.2.1. Key Fields

Every plate has key fields. If the visit number is not part of the barcode, then the first required field on the plate is the visit number for the plate. The next required variable is the subject ID. These variables recur for each plate and are easily grouped into a module. Let's call this module "Keys". Subject initials are sometimes included on CRF pages and can also be included in the "Keys" module. Our example also has a date field for recording the actual date for the Sunday of the diary entry. This can also be included in the "Keys" module. Define four fields - the repetition number (representing the study week #), the subject ID, the subject initials and the date of the first diary entry. Lets call these fields DIARYWK, SUBJID, PINIT

and PDDAT. Depending on where you are in the study definition process, SUBJID and PINIT as well as the Keys module may already be defined. When you define DIARYWK, keep in mind that you can have one or more repetition number fields defined in your module as different plates use repetition numbers in different ways. In our example, the repetition number for this plate cannot overlap with visit numbers used for subject scheduling or for other recurring plates. The required key fields DIARYWK and SUBJID must be defined with the following properties:

Visit.

- used to identify each visit for a plate, e.g. clinic visits numbers, AE report numbers, log form numbers, etc.
- integer in the range 0-65535
- located either in the barcode immediately following the plate number, or as the first data field on the plate.
- if in the barcode, the visit number is limited to the range 0-511
- if in the first data field, the field type must be numeric (i.e. number, choice, check or VAS), and appears as data field #6.
- Need: essential.
- Hidden: no.
- Alias: if fixed in the style it must include the plate number to make it unique, e.g. VISIT$(plate).
- Display & Store: must both be the same length (1-5 digits).
- Format: optional, may contain a fixed component, e.g. 7nn, but may not contain delimiters, i.e. no '.', '-', or other characters.

Subject.

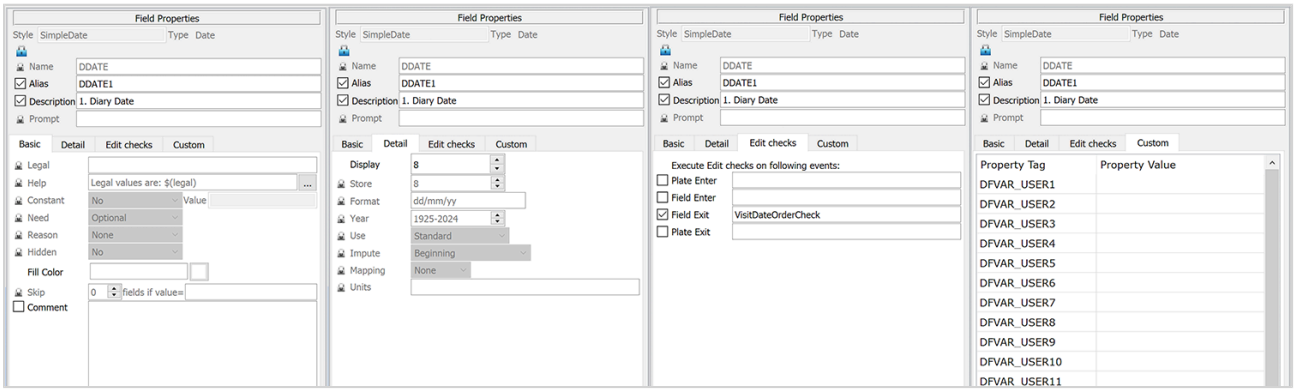
- used to identify each subject in a clinical database, or each clinic and clinic personnel in a trial management database
- must have the same field properties on every plate, and thus typically has all properties locked in a user defined style
- integer in the range 0-281474976710655
- located immediately after the visit number, and appears as data field #7
- Need: essential
- Hidden: no
- Alias: if locked in the style it must include the plate number to make it unique, e.g. ID$(plate)
- Legal: optional, may be defined using $(ids) which evaluates to all subject IDs specified for each site in the sites database file. If used, the $(id) meta-word cannot be combined with any other specifications.
- Display & Store: must both be the same length (1-15 digits)
- Format: optional, may contain a fixed component, e.g. nn5nn, but may not contain delimiters, i.e. no '.', '-' or other characters

The following example shows the properties for the diary start date on the CRF in the preceding example. The field properties supported by DFdiscover are described in Styles, and are not repeated here.

The definable properties are grouped in 4 tabs:

- Basic: generic properties that apply to every field.
- Detail: properties that are specific to the field's data type.

- Edit checks: name(s) of previously defined edit checks, referenced to indicate when they execute.
- Custom: optional, user-definable field properties.



- A field named 'DDATE1', previously created in the modules dialog for this particular field was selected from the Plates-Modules-Fields list.
- Several properties are locked in this style and cannot be changed.
- An open check box appears beside unlocked properties. All of these properties should be set and locked at the field instance level.

To modify field properties after a field has been defined, select the field, make the revision and click Apply to save it. More than one field can be modified at the same time if they share the same style using Shift-Select to select all of the fields, making the revision in the Field Properties dialog and then clicking Apply. A confirmation dialog appears indicating the field properties that have been changed. Properties can be deselected in this dialog. Only those properties that remain selected are applied to the selected fields when OK is selected to save the changes.

3.2.2. CRF Field Properties

Other fields on the CRF also require modules and styles. In our example, the next 6 variables represent diary data recorded for the first day. These variables repeat for subsequent days. Since modules can repeat on a CRF page, a single module for the fields in a diary daily entry can be reused for each day. Let's call this module the "DIARY" module. The six variables are defined in this module based on the data requirements implied by the CRF page.

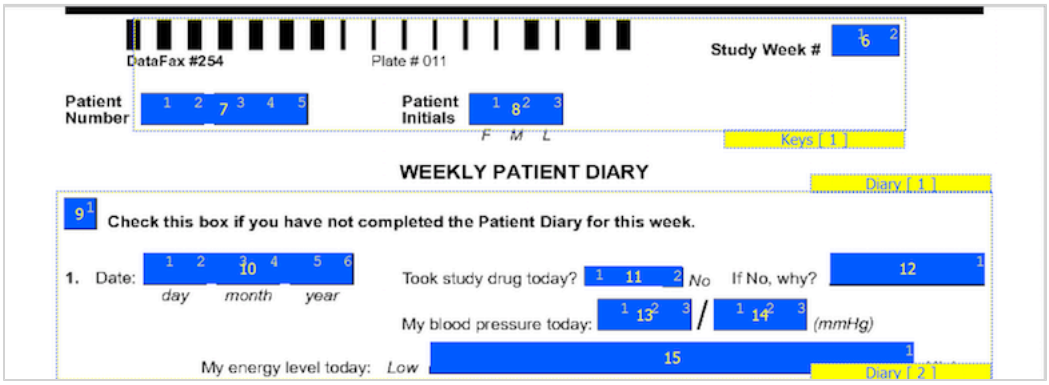
3.2.3. Arbitrary CRFs

In pure EDC studies, where the data may never exist on paper, data entry screens can be created for **DFexplore** by using Add eCRF Plate option in **DFsetup**. Data fields can be added to eCRF plates by dragging fields or modules from the plate-module-field navigation window and dropping them on the eCRF plate.

On CRF plates, data fields can be added by dragging out data entry widgets and selecting respective field from plate-module-field navigation window. The following example shows what the data entry widgets look like after defining the 10 CRF data fields. The field number appears in the middle of each data entry widget.

The first 7 fields on each CRF page are always the same. There are 5 fixed header fields: record status, workflow level, image link, study number, and plate number which do not appear as data entry widgets. Field 6, the key field used to distinguish repetitions of a study plate (due to the plate repeating over several visits), may appear in the barcode, but in this example it is the first data field defined on the page. Field 7 is always the key field for the subject ID.

The order in which boxes on the CRF were clicked is shown in a smaller font inside each box and should be reviewed to confirm that the boxes have been clicked in the correct order (left to right).



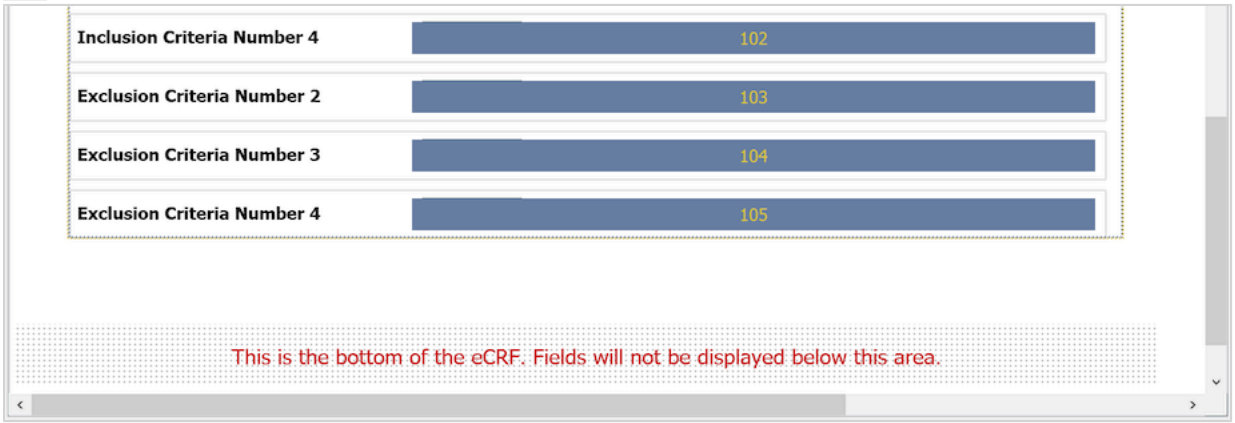
3.2.3.1. Page Limits

Paper-based CRF backgrounds. By default, for a CRF with a paper-based background, the boundary is an 8.5 x 11 inch space. The boundary is expanded if a larger CRF image is imported for the data entry background. **DFsetup** prevents fields from being created or dragged outside this region and any fields pasted from another plate that would fall outside this region are automatically positioned within the boundary. It is possible for data fields to fall outside the boundary if they were defined on a larger version of the CRF background or imported from another study. Such fields are visible outside the boundary and should be re-positioned within it, otherwise they may not be visible in **DFexplore**.

DFsetup and **DFexplore** can accommodate both landscape and portrait backgrounds. However DFdiscover is able to perform ICR, and **DFexplore** is able to print CRFs, only for CRFs in portrait orientation and sized for US letter and A4 pages. Landscape and larger backgrounds must only be used for electronic data capture. Testing has included page sizes up to 14 inches (both horizontal and vertical).

eCRF. An eCRF is limited in length to 4 US letter pages, or 44 inches, and 8.5 inches in width when the plate property Layout is set to Portrait. When the plate property Layout is set to Landscape, the eCRF width is 11 inches in width. There is no specific number of fields in this restriction; the actual number of fields will vary based upon the sum of the lengths of the individual fields.

As modules and fields are added to an eCRF, a warning will be displayed as the page length approaches the limit. The warning has the following appearance. As the warning indicates, additional fields beyond the limit are not visible. Use Field List to display and review all of the fields, ideally deleting some so that the total layout length is within the 44 inch limit.



3.2.4. Custom Properties

Custom properties may be assigned at the level of the study, plate, module instance, and data field. These properties allow the storage of additional information about the data, For example, custom properties could be used for non-collected CDASH or SDTM variables, risk-based monitoring details, database specifications, or CRF versioning.

Each custom property may have a tag assigned, as described in Custom property tags. Study custom properties are assigned values under Global Settings, as described in Global Settings. Plate custom properties are assigned values under Plate Properties, as described in Plates. Module custom properties are assigned values under Module Properties for a module instance, as described in Modules. Field custom properties are assigned values under Field Properties for a data field on a plate, as described in Common Properties.

The values are accessible via edit check functions, specifically `dfstudyinfo`, `dfplateinfo`, `dfmoduleinfo`, and `dfvarinfo` using either the property tag or default name. Custom properties can be included in DFexplore List View as metadata and exported from DFexplore and from DFexport.

3.3. Field Location

The first step in mapping field is to add the modules containing the field definitions for the current plate. The next step is to map a field to its location on the CRF plate. This is done by clicking the boxes that make up the field or by dragging a data entry widget over the desired location in the CRF window. Once a data entry widget has been created it can be moved using the arrow keys or by dragging it with the mouse. Holding the shift key while using the arrow keys moves the data entry widget in large steps. To cancel the creation of a new data entry widget at this stage click anywhere in the screen background. A new data field only becomes permanent when a field is selected from a module included for the current plate and applied.

Table 3.1. Locating data fields in the CRF window

Field	Type	CRF Design	To Locate Data Field	Module
Study Week #	number	2 boxes	click the left box then the right box	KEYS
Subject ID	number	5 boxes in 2 groups	click each box from left to right	KEYS
Subject Initials	string	3 boxes	click each box from left to right	KEYS
Start Date	date	6 boxes in 3 groups	click each box from left to right	KEYS
Took study drug today?	choice	2 check boxes	click each box from left to right	DIARY
Why not	string	7 cm line	drag data entry widget from left to right	DIARY
Systolic BP	number	3 boxes	click each box from left to right	DIARY
Diastolic BP	number	3 boxes	click each box from left to right	DIARY
Time	number	4 boxes in 2 groups	click each box from left to right	DIARY
Energy	VAS	10 cm line	drag data entry widget from left to right	DIARY

The following sections describe how to locate each data type in more detail.

Number & Date

Numbers are typically designed with a separate box for each digit. The individual boxes are necessary if you plan to fax/scan the forms and want the ICR software to read them. A data entry widget is created by clicking each of the boxes from left to right. The order is important and must always be left to right.

Dates are created in the same way, by clicking the boxes, from left to right, for each of the 3 parts: day, month and year. Dates must include all 3 parts, but the parts can appear in any order, have a part delimiter (e.g. a slash) or not, and can use numeric or 3 character months, which can appear exactly as entered or be mapped to upper- or lowercase for consistency.

Both dates and numbers can include delimiters but they are not relevant when setting field location. Instead delimiters are added in a format specification (described later).

Time

Times are similar to dates. The CRF is designed with a separate box for each digit. Times include two or three 2-digit parts: hours and minutes and optionally seconds.

Choice

Choice fields consist of 2 or more mutually exclusive options and come in 2 versions:

1. **Choice Boxes:**

   These fields appear on a CRF page as a set of labeled check boxes which can be read by the ICR software. A data entry widget is created by clicking each of the boxes from left to right for horizontal layouts like the yes/no question in the example above, and top to bottom when the boxes are arranged in a vertical column. The order is important. In **DFexplore** the desired choice can be selected using the mouse or the keyboard. When the keyboard is used the user enters the box order number to select it; defining the data field by clicking left to right, or top to bottom corresponds to most users' intuitive sense of the box order number.

2. **Drop-Down Lists:**

   These fields appear on a paper CRF as a box or line where the user writes a response, but in **DFexplore** they appear as a data field with a drop-down button of response options. The selected option is then displayed in the data field. To create a data entry widget for these fields, click the box or drag one out over the CRF background where the field appears. After dragging to create a field you can use Adjust to Fit to resize it to match the longest choice option. If there is not enough room on the data screen to accommodate the longest option it is truncated when displayed in **DFexplore** but the complete option label can always be seen using the drop-down button.

Check

Check fields appear on a CRF page as a single box which can be checked or left blank. A data entry widget is created for these fields by clicking the box and then entering a numeric code and a text label for the 2 response options: blank and checked.

Text

Text fields typically appear on a CRF page as horizontal lines on which the users print their response. A data entry widget is created by dragging out a rectangular box over the text line. This is done by clicking one corner of the desired location and holding the left mouse button down while dragging the mouse to the opposite corner.

While dragging in an upward direction (lower-left corner to upper-right corner), widget height is constrained to the DFdiscover default of .25 inches. While dragging in a downward direction (upper left corner to lower-right corner), it is constrained by the choice of 'Maximum Box Height' in the 'Guides' section of Global Settings.

When the mouse is released a new widget is created in the foreground color. The widget can be moved if necessary to adjust its location in the CRF window by holding the left mousebutton down over the widget and dragging it to the desired location.

Multi-line text widgets can also be created. This is done by holding the Control key (or Command key on Macs) while dragging out a rectangular box over a blank area of the CRF. If space is not available on the CRF for a data entry widget large enough to hold the desired text a single line widget can be used with a database store length greater than the display length on screen. In such cases the height of the text field automatically expands into a multi-line text box when the the user enters the field in **DFexplore**, and returns to its display size when the user leaves the field.

VAS (Visual Analog Scale)

Data entry widgets are created for horizontal Visual Analog Scales, like the 'energy level' question in the example above, by dragging out a rectangular box the same length as the VAS scale and positioning it on top of the scale line.

DFexplore also supports vertical VAS fields but these fields cannot be read by the ICR software. Data entry widgets for vertical Visual Analog Scales are created by dragging out a box at each end of the scale. The boxes must we wide enough to display a numeric value and must be positioned so that the top edge of the upper box corresponds to the top end of the scale and the bottom edge of the lower box corresponds to the bottom of the scale.

Hidden Field

Data entry widgets can be created for fields that do not appear on the CRF page by dragging out a data entry widget anywhere in the CRF window and defining it like any other data field. Such fields are sometimes used for coding or other central office purposes. Whether a user sees such fields in **DFexplore** is determined by whether their study role includes permission to see hidden fields.

All field types can be created this way. For choice fields a box for each choice must be created, but all other field types can be created by dragging out a single data entry widget.

In most cases, you do not want to print labels and boxes for hidden fields on the CRF pages imported into **DFsetup** because all users can see what is printed on the CRF backgrounds. The only thing that is hidden for a hidden field is its database value.

3.4. Creating New Fields by Copy & Paste

Any number of fields on the current plate can be copied into the **DFsetup** clipboard, and then pasted to create new data fields on the same plate or another plate in the study setup. Copied fields remain in the clipboard until replaced by another copy, and can be pasted as many times as needed to create new fields. The copy and paste steps are as follows.

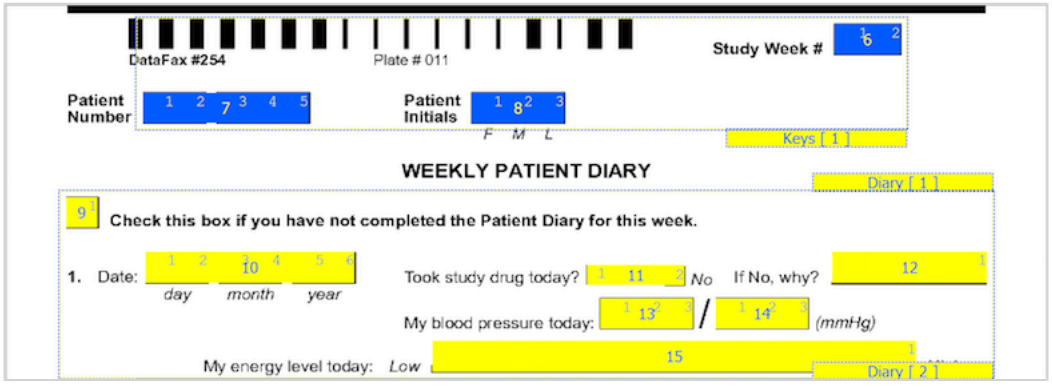
1. **Select Fields To Be Copied**

   Fields in the current selection set are shown using the foreground color. A field is selected by clicking it with the mouse. Other fields can be added to create a selection set by holding down the shift key while clicking each field.

   A group of fields can also be selected by holding the shift key down and then clicking and holding the left mousebutton while dragging a box around the desired fields.

   You can also select all the fields in a module by clicking the module label or selecting the module from the **Plates-Modules-Fields** list.

   There is no method for removing a single field from the current selection. Instead you must cancel the entire selection set and start over. To cancel the current selection click anywhere in the background (while not holding the Shift key).



2. **Copy the Selected Fields**

   Select **Field** > **Copy** to copy the current selection to the **DFsetup** clipboard.

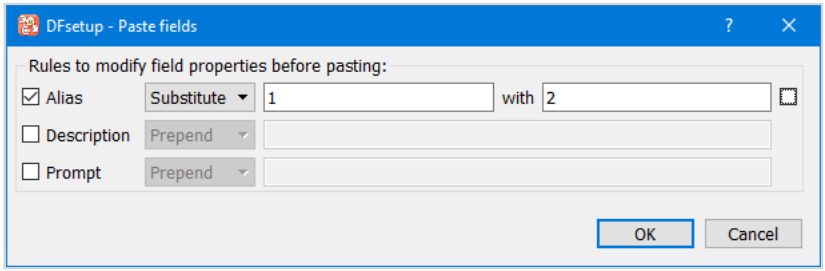
3. **Paste the Copied Fields**

   Select **Field** > **Paste** to create the new fields. The dialog shown below appears first.

   The names and descriptions of the copied fields can be modified for the new fields by: pre-pending, appending or substituting specified strings. This is optional, and does not change the values in the clipboard copy.

   To make the same change to all three field properties, check the box beside the first row. This propagates changes made to this row to the other 2 rows.

   Click OK to paste the new data fields onto the page or Cancel to abort the paste operation.

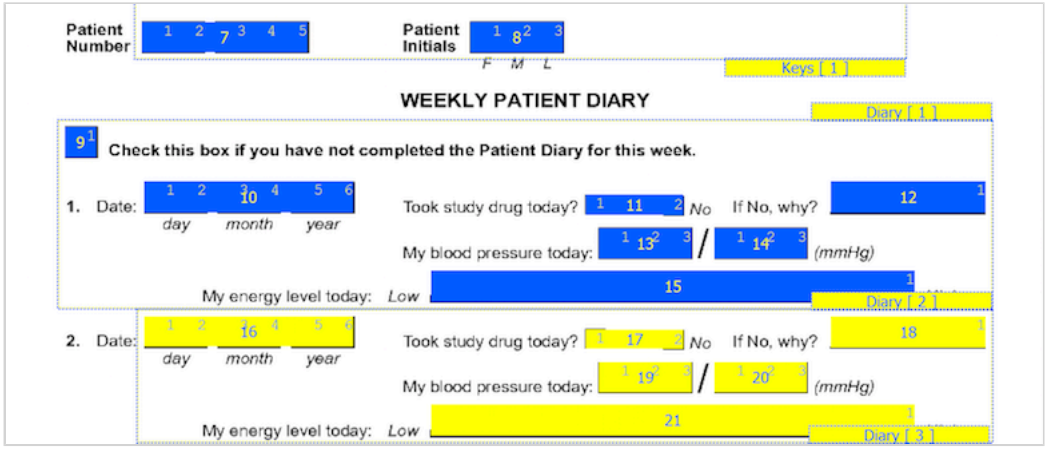


   - In this example all occurrences of '1' in the copied fields are replaced with '2' in the field names of the pasted fields.

4. **Position the New Fields**

   The pasted data fields appear offset just below the copied fields and must be moved into position on the page. This is done by clicking and holding the left mousebutton over any one of the pasted fields and then dragging the entire set of fields into position.

   To end the paste operation, leaving the pasted fields in their current position, left-click anywhere in the CRF window background. Individual fields can then be selected and dragged if necessary to adjust their final position.



5. **Edit the New Fields**

   After creating the new fields you may need to edit some of the field properties to finish their definition. This is done by selecting each field and editing in the **Field Properties** panel.

3.5. Positioning and Aligning Data Fields

This section does not apply to data fields on plates that are defined as eCRF.

Data fields are typically positioned to overlay the underlying data collection area on the background CRF. Any data field can be selected and then moved to re-position it. After selecting a data field simply use the mouse to drag the field to it's new position. Releasing the drag will update the position of the data field.

In some cases, the data fields do not have an underlying CRF to guide positioning. Approximate positions for individual fields can be specified by dragging and dropping fields. Where multiple fields need to be accurately positioned relative to each other, alignment tools are available.

Whenever two or more data fields are selected, the alignment ribbon is displayed at the bottom of the main window. The order in which the data fields are selected can be important - when aligning, positioning is always done relative to the data field selected first.

Each icon in the ribbon represents an alignment action - when clicked, the positions of the selected data fields are updated to reflect the requested alignment. Clicking the rightmost icon undoes the last alignment action (note that only the last alignment action can be undone).

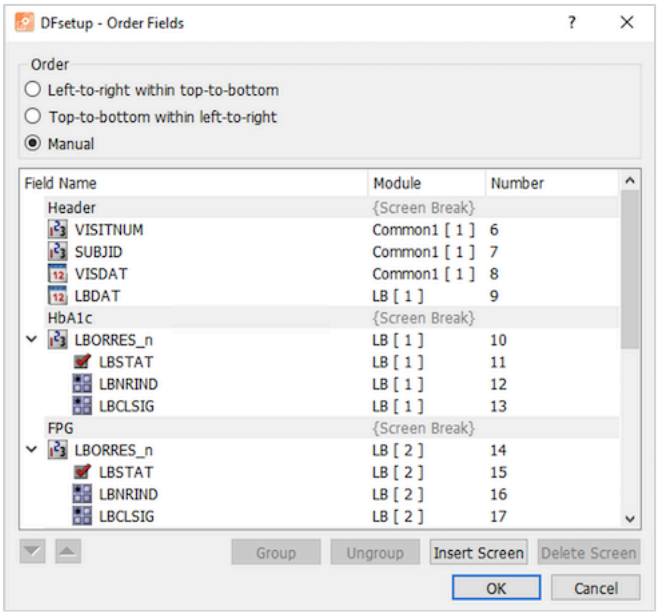
The alignment actions are also available from the context menu when two or more data fields are selected.

3.6. Reordering Data Fields, Screens and Grouping

The field number displayed in the center of each data entry widget identifies the order in which fields are traversed during data entry. When setting up a new plate it is important to review the field numbers and reorder them as needed to achieve the desired field traversal.

> **Important**
>
> The field order displayed on each plate is also the order in which fields are stored in data records and the audit trail journal files. It also reflects the order in which fields appear on eCRFs. Revising field order after data has been collected is possible. DFdiscover reorders fields on data records and saves the previous study setup for use in creating audit trail reports. However all necessary revisions cannot be automated; edit checks for example might contain code referring to fields by position and/or number that need to be revised. Before proceeding it is essential to consider all of the study components that may be affected and follow the instructions described in Modifying Plates that Contain Data.

To reorder data fields select **Field** > **Order** and use the dialog illustrated.



- The dialog allows moving fields up/down either by drag and drop operation with the mouse or by using the up/down buttons. For plates with CRF backgrounds, fields may also be ordered automatically according to their placement on the CRF background, either left-to-right within top-to-bottom or top-to-bottom within left-to-right. The field number displayed in the last column identifies the order in which fields are traversed during data entry.

- Fields can also be grouped/ungrouped using the Group and Ungroup buttons. Multiple consecutive fields must be selected in the field list to enable Group. At least one grouped field must be selected to enable Ungroup. Drag and drop using the mouse can also be used to add/remove fields in a group.

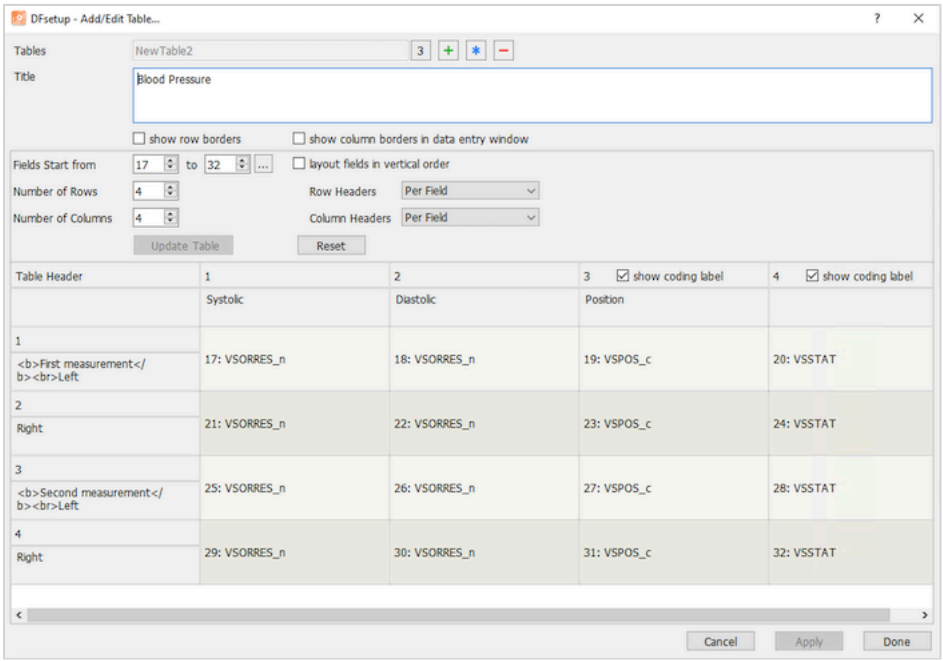
- To add screens, select the field that begins the next screen, and click Insert Screen. The screen break can be titled, typically for use as a header for different sections in an eCRF. Any screen without fields is discarded without warning.

Screens provide the software with information about the ideal place in the eCRF to split the screen content to accommodate viewing on smaller screen sizes in **DFcollect** and **DFweb**.

3.7. Adding Tables to eCRFs

On eCRFs, fields can be displayed within a table instead of the default display of one field per row. Tables can include a custom number of rows and columns with custom table title, table header, and row and column headers. Each cell in a table may contain a single data field or contain custom text (including basic HTML formatting), or remain blank.

To add or edit tables on the current plate, select **Field** > **Add/Edit Table** and use the dialog illustrated.



- The dialog allows the definition of one or more eCRF tables on a plate. Tables are assigned default names which may be customized. Use the icon next to the table name to see the list of all tables defined on the current plate. Use the icons to add, rename, or delete tables on the current plate.

- The table title, if defined, is displayed above the table on the eCRF. Either or both row and column borders may be shown.

- Define the table's contents by specifying the range of fields to include in the table and the number of rows and columns. The column widths are equally distributed automatically across the width of the eCRF regardless of the contents of each column. The row heights are auto-adjusted based on the contents of each row. Click ... to select from the list of fields on the current plate. Note that a data field may only be included in one table at any given time. Click Update Table after making any changes to update the table in the bottom half of the dialog.

- By default, fields are displayed in horizontal order, row by row. To display fields in vertical order, column by column, select **layout fields in vertical order**. By default, row and column headers are defined per field, but any of Field Name, Field Alias, Field Prompt, or Field Description may be selected to apply to all row and/or column headers. Click Update Table after making any changes to update the table in the bottom half of the dialog. Click Reset to change back to the default or last saved selections.

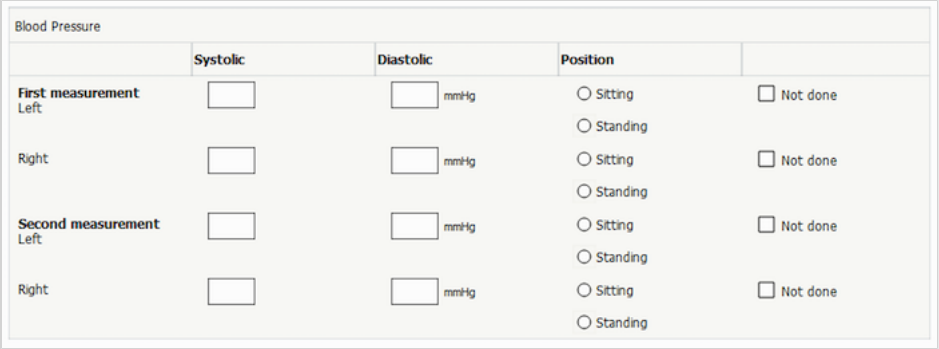
- Headers for the table, rows, and columns are optional and may be manually entered or any of Field Name, Field Alias, Field Prompt, or Field Description may be autofilled via the ... within the selected header cell. (Note that subsequent updates to the name, alias, prompt, or description will not update the headers automatically.)

- The table and column headers are formatted as bold text by default, while the table title, row headers, and custom text in cells are formatted as plain text by default. Simple HTML may be used for additional formatting, such as using bold and italics, changing the font color, or adding line breaks, spacing, or symbols. The same HTML formatting allowed in the Prompt and Instruction field properties is also allowed here. Use the Open Editor option under ... for a larger window to edit the text.

- Where choice or check fields are included in a column, the **show coding label** option will appear to control whether the label is displayed for each check and choice field in that column. Use the Show As field property to customize the appearance of choice fields (see Specific Properties for Choice Fields for details). Column headers may include a backslash (\) to subdivide headings within the column, for example for choice field options without labels.

- The cells within a table may include a single data field or custom text, or may remain a blank cell. Click ... in a selected cell for options to insert or delete a blank field and to insert or delete a blank column. When a blank field is added, a blank cell is created and the data field is moved to the next cell, whether in the current row or the next row.
- To add custom text within a table, create a blank field, select the cell, and begin typing. Simple HTML formatting is allowed as for the Prompt and Instruction field properties. Use the Open Editor option under ... for a larger window to edit the text.
- To save your changes and return to the main **DFsetup** window, click Apply and then Done. To close the dialog without saving your changes and return to the main **DFsetup** window, click Cancel.

The table definition above results in the below table as shown in **DFexplore**:



Tables are supported in **DFexplore** where eCRFs are displayed with a fixed width layout. Uncheck the **eCRF Table** option at the bottom of the main **DFsetup** window to see an approximation of the eCRF appearance in **DFcollect** and **DFweb** where tables are not currently supported, due to the responsive layout design for smaller screens on tablets and mobile devices.

3.8. Modifying Plates that Contain Data

Because the field numbers on each plate match the order in which fields are stored in the study database, inserting, or deleting fields after data has been collected must be done with care as described in this section.

DFdiscover provides support for field numbering changes that occur after data has been collected, because fields have been reordered, added, inserted or deleted. Such changes can only be made using the Exclusive or Developer modes in **DFsetup**. When setup changes are saved DFdiscover will:

- permanently and irretrievably delete any data fields that have been deleted in the study setup
- add blank values for any data fields that have been added to the setup
- reformat existing data records to match the new field order
- update queries and reasons so they remain attached to the correct data fields
- save the old setup specification, and add setup revision records to the study journal files so that the audit trail report are able to keep track of the field changes

But this is all that DFdiscover can do automatically. In addition you may need to make the following revisions manually:

- Edit checks that refer to fields on the revised plate by number or using the @T relative position notation may need to be revised because of field number and/or position changes.
- Shell scripts that use **DFexport.rpc** to export data records for custom reports, plate arrival triggers, `dfexecute` edit check functions, or other tasks performed by custom programs may need to be modified to deal with the new field order.

- **DFsas** job files used to create SAS® data sets containing data from revised plates may need to be modified to account for any new or deleted fields.
- Task and List View definitions for the revised plate may need to be corrected.
- If **DFsqlload** is used to export study data to an SQL database the table definition is updated and the restructured data reloaded the next time **DFsqlload** is run. However, you may need to revise SQL procedures stored in your SQL database to deal with the plate revisions.

If you need to modify the data field list on a plate but are unable or unwilling to make the manual modifications required by a change in field numbering you may want to make your changes in a way that preserves the existing field numbers. This can be accomplished as described below.

1. **Enter DFsetup in Exclusive Access Mode**

   No other user can be in the study database while these changes are being made.

2. **Import the Revised CRFs**

   Import the revised CRF pages as described in Import CRFs. Because the plate number on a revised CRF already exists in the study setup, the import dialog will show **Replace** beside the plate number. This indicates that the new CRF page will replace the existing one. You do not need to import all CRF pages, only the ones to be modified.

3. **Adjust Field Positions**

   Any fields which no longer appear in the correct locations should be selected and moved to their new positions on the CRF. When doing this be sure you do not change the field numbers. If you need to change the field traversal order this can be done by applying edit checks as described below.

4. **Deleting Fields or Modules**

   Do not delete data fields or modules if you want to be able to retrieve this data in the future or you want to be able to create an audit trail tracing the history of these fields using **Field** > **Show History of All Changes to This Field** in **DFexplore**. When a field or module is deleted from the study setup, DFdiscover permanently deletes any values that exist for these fields, and the fields themselves, when the setup changes are saved. Also it is important to realize that you cannot re-create deleted fields in the setup to bring them back, because each new field gets a new internal field tracking number, even if the field is given the same name and appears on the same plate. A re-created field has no link to any previously deleted version of that field.

   Instead of deleting fields you no longer want to appear in **DFexplore** we recommend that you hide them, either by setting the Hidden field property to 'Yes' so that users without permission for hidden fields do not see them, or by using edit check function `dfaccess`.

   If you decide to delete fields or modules, and then use report **DF_ATmods** to produce an audit trail showing the history of all changes on the plate that contained these fields, the report shows the history of the deleted fields and the point at which they were deleted. This is true even if you also delete the plate itself.

5. **Adding Fields or Modules**

   When a new field is created, it is automatically assigned the next field number after the last one on the plate and thus does not change the numbering of existing fields. The new field must already be defined in a new or existing module.

   If you do not want the new fields to be positioned at the end of the plate drag them to the desired location on the CRF page, but leave the field numbers unchanged. We will fix field traversal in the next step.

   When fields are added to the end of a plate DFdiscover will reformat the existing data records to insert the new fields with blank values. Thus the data records will change but the field order will not change for all of the data collected so far.

6. **Changing Field Traversal Order**

   By default **DFexplore** traverses data fields in field number order when a user tabs through the data fields on the plate. If you have reordered fields or positioned new fields between existing fields, this is no longer the desired field traversal order. An edit check can be used to achieve the desired order, without changing field numbers, as illustrated below.

```
# Run: on field exit
# Use: change backward and forward fields traversal
# Arg: f1 = next field on backward traversal
#      f2 = next field on forward traversal
edit GOTO(number f1, number f2)
{
       if( dfdirection()>0 ) dfmoveto(@[f2]);
  else if( dfdirection()<0 ) dfmoveto(@[f1]);
}
```

   For example, if a new field numbered 44 has been positioned between fields 15 and 16 use edit check GOTO as follows:

   - On exiting field 15: GOTO(14,44)
   - On exiting field 16: GOTO(44,17)
   - On exiting field 44: GOTO(15,16)

   A similar approach can be used to skip over "deleted" fields (which are not really deleted, just no longer relevant). For example field 22 could be skipped by adding GOTO(21,23) as a field entry edit check on field 22. This has no effect if the user clicks on field 22 with the mouse but is skipped when keyboarding through the fields.

7. **Test The Revised Plate**

   First review the revised plate in the **DFexplore** List view to verify that any new fields appear at the end of the data records, as expected. Then build a task set containing the revised plates using **Select** > **By Data Fields** in the **DFexplore** Data view, and tab through the fields to check: field traversal order, edit check behavior, and data field values.

8. **Other Revisions**

   Although this procedure does not change the numbering of existing data fields you may need to make revisions to include any new fields you have added in: edit checks, **DFsas** jobs, shell scripts, and task and list view definitions. Any required modifications are much less complicated than would have been the case had you changed field numbering on the plate.

3.9. Adding & Deleting Plates

1. **Deleting Plates**

   A plate can be deleted from the study setup by selecting **Study** > **Delete Plate**, and then removing the plate from the study Visit Map using **View** > **Visit Map**. Deleting a plate deletes the plate definition from the study setup and makes it impossible to enter new data records or retrieve existing data records for the deleted plate.

   Deleting a plate has no effect on other plates.

> **Important**
>
> Do not delete a plate (or data field) and then re-create it. Each new data field gets a new internal field tracking number, thus any such re-created fields would be considered new with no link to the deleted fields, even if you define them with the same field name on the same plate number.

When the setup is saved DFdiscover deletes the data values for all deleted fields. It thus becomes impossible to retrieve them. Also since the study setup no longer includes the definition of deleted fields it is not possible to generate an audit trail report that focuses just on the deleted fields. To show the history of deleted fields **DF_ATmods** must be run to show all changes on the relevant plate.

2. **Hiding a Plate**

   Instead of deleting a plate you may simply want to hide it from some or all users. A plate can be hidden from selected users by changing the role definitions set for all **DFexplore** users in **DFadmin**.

   Simply removing a plate from the study visit map, while making it impossible to enter new data records for that plate, does not prevent users from retrieving and changing existing data records, unless the records are hidden using `dfneed` in the DFopen_patient_binder edit check. If this is not done the existing data records are shown as unexpected (by the current study visit map) in the subject binder.

3. **Adding New Plates**

   Since each plate is stored in a separate data file a new plate can be added at any time with no impact on existing plates. The process is the same as the steps followed to create the original study plates. PDF files containing the new CRF images are imported, and given new plate numbers. New styles, modules and data fields can then be created as needed for the new plates. As soon as the new plates are added to the study visit map, they become available in **DFexplore** for data entry. This can be done while users are logged in to **DFexplore** but such users do not see the new plates unless they exit and re-enter the study, or the next time they login.

4. **Adding New Versions of Existing Plates**

   Sometimes the revisions required to an existing CRF plate are so extensive that it is not possible to simply modify the existing plate as described in Modifying Plates that Contain Data. Instead a new version of the CRF page with a new plate number may be required. As described above adding a new plate can be done at any time, but how do we tell DFdiscover that this new plate is a new version of an existing plate?

   To accomplish this we need to make some changes to the study visit map and add some new cases to the conditional visit map. First, if the old version of the plate was required at some visits the visit map must be changed to make the new version required and move the old version to the optional plate list. And if the old version was optional the new version needs to be added to the optional list so that both versions appear in the optional list.

   The final step is to add two new conditions to the conditional visit map, one that declares the new version unexpected if the old version has been entered, and another which declares the old version unexpected if the new version has been entered.

3.10. Modifying Style, Module & Field Definitions

During a new study setup you may decide to create and apply a new style to fields you have already defined, or modify some of your previous style, module or field properties. Changes you make at the style level follow these rules:

- properties you lock at the style level are automatically applied to all modules with fields that use that style.
- style properties you define but do not lock (i.e. default suggestions) are automatically applied to any field that uses that style if that property is not already locked at the module or field level.

Style default suggestions can be useful when defining new fields, but in most cases you should apply the field level lock to all properties not locked in the style or module, so that any change in the default style suggestions are not applied to existing data fields.

Important

Style, module and field properties can be modified in any way you want before data collection begins, but if data has already been entered you must not change field, module and style properties in a way that would conflict with the storage or interpretation of the existing data values.

If data exists in the database for fields you plan to redefine follow these rules:

- Do not change the label of coded fields if this would change the meaning of existing data values. For example, if a field is coded 1=yes and 2=no, do not change the coding to reverse the meaning to 1=no and 2=yes.

- Do not change the field storage length if values already collected would not fit in the new storage length. DFdiscover does not automatically truncate all existing values in these fields, but if someone resaves the data record, either after modifying some field or just to move it to a new workflow level, any oversized fields are truncated.

- Do not change the format of date fields if this would make existing data values invalid. For example, changing a date format from dd/mm/yy to mm/dd/yy would switch the meaning of the day and month components of all existing date values.

- Changing a numeric format in a way that expands the values that can be saved does not change the numeric value of existing database values. For example, if the format is changed from nn.nn to nn.nnn, an existing value of 12.34 would simply become 12.340 if the record is retrieved and resaved in **DFexplore**. But doing the opposite, reducing the size or precision of a data field, does make existing values incompatible with the new format. For example, changing the format from nn.nnn to nn.nn, would truncate existing numeric values and changing a format from nnn to nn might mean that some current values no longer fit in the data field.

  Changing the format does not automatically change the data fields in the database. When an affected data record is retrieved, **DFexplore** displays a warning dialog showing the affected fields. Then, after reviewing and perhaps correcting the changes, the record must be saved to store the newly formatted values in the study database. If you are sure that all values fit in the new format and do not want to review the changes, the records can be retrieved and batch validated in **DFexplore**.

- Do not change names unless you are willing to also change all of the places where they are used, e.g. edit checks, SQL snapshots, and **DFsas** job files.

- There is no harm in changing data field descriptions, but there is one consequence you need to understand. When a new data query is created the data field description is added to the query, and becomes part of the query record. The field description in existing data queries is not updated when a data field description is changed. Changing the description only affects data queries created after the change is made.

- There is no harm in changing legal value definitions during a study. It is not uncommon to expand the legal range to allow a wider range of values to be considered legal. Since all existing fields meet an expanded range, an expanded legal range presents no problem.

  Decreasing the legal value list is also allowed and has no effect on values that have already been collected, even if they are now illegal. The only consequence is that some data records that have been saved with status `final` may no longer qualify for this status. Nevertheless, the status of such data records are not changed, unless they are resaved in **DFexplore**. If you want to do this, the easiest way is to open the data records in **DFexplore**'s List View and sort on the modified data field. You can then double-click each now illegal value to jump to the Data View and resave the data record with status `incomplete`, perhaps after adding a data query to the now illegal values.

- There is no harm in adding a new code to an existing choice field. This does not affect any of the existing codes, and thus does not change the interpretation of existing data fields. However, adding new coding options to a choice field may require modifications to the CRF to add check boxes for the new choice options.

3.11. Modifying a Data Entry Widget

If the CRF layout of a data fields changes (e.g. to add check boxes for new choice options, change the layout of existing choice options, change a text field from single line to a multi-line box, increase the number of digits in a numeric field, etc.) you do need to import a modified CRF background and change the data entry widget to match the new field design. Proceed as follows; we'll use adding a new choice option as an example:

1. Modify the CRF page to add the new choice box options and save it as a PDF file.
2. Open the module view and select the module and field you wish to modify. Add the new choice option to the field properties and save it.
3. Import the revised CRF page using **Study** > **Import CRFs** to replace the existing CRF background image. This has no effect on the data fields already defined on this plate.
4. Adjust the position of data fields on the new CRF background, as needed by dragging them to their new locations on the page.
5. Drag the widget for the choice field to be changed out of the way, and then click each of the boxes making up the field, just as though you were defining it all over again.
6. When all choice boxes have been identified, click the choice field widget that you moved out of the way. A dialog appears asking if you want to relocate the data field to the new boxes you have just checked.
7. Click Yes. The data entry widget you moved out of the way vanishes and the revised field definition moves to the new data entry widget you just created.

This same procedure can be used to reassign any data field to a new data entry widget.

> **Important**
>
> Remember that changes to the study setup are not permanent until you select **File** > **Save**. Changes can only be tested after saving, when the new setup becomes available. Remember as well that **DFexplore** reads the setup only once when it starts - to reload a new setup requires restarting **DFexplore**.

3.12. eSignature Module and 21 CFR Part 11 Compliance

The eSignature module provides a fully compliant default record signing mechanism that meets 21 CFR Part 11 requirements without additional edit check programming and testing. The eSignature module can be instanced on any page. The case where a single signoff page covers a range of pages is not implemented. It is possible to batch sign pages, making the job of maintaining compliance a bit easier.

DFdiscover comes preloaded with the eSignature module. When you create a new study definition, or import an existing study definition from an older version of DFdiscover [1], the eSignature module appears in the list of modules.

The eSignature module has five predefined fields:

- `SigName`: the full name of the signer. The system determines this by matching the login username with the user definition in the permissions database.

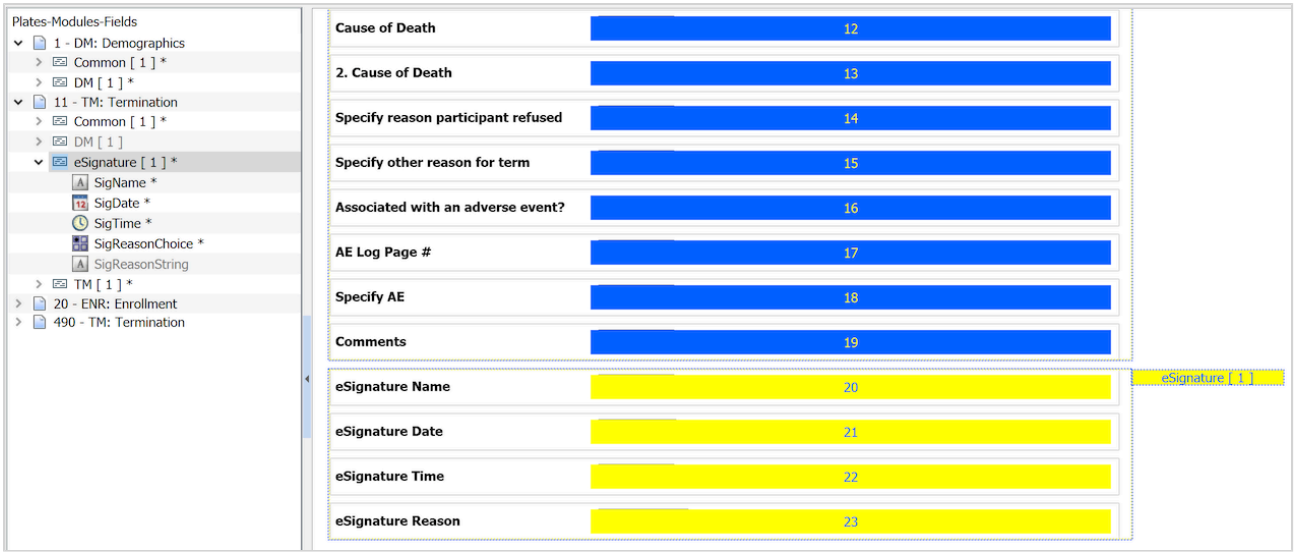
[1] The eSignature module was updated in DFdiscover 2018 Version 5.1.0 to additionally include a reason for the signature. All versions provide name of signer, date and time of signing.

- `SigDate`: the date of signing, from the signer's computer and local timezone.
- `SigTime`: the time of signing, from the signer's computer and local timezone.
- `SigReasonChoice`: the reason for the signing, as a choice.
- `SigReasonString`: the reason for the signing, as a text string.

3.12.1. Requiring eSignature on a Plate

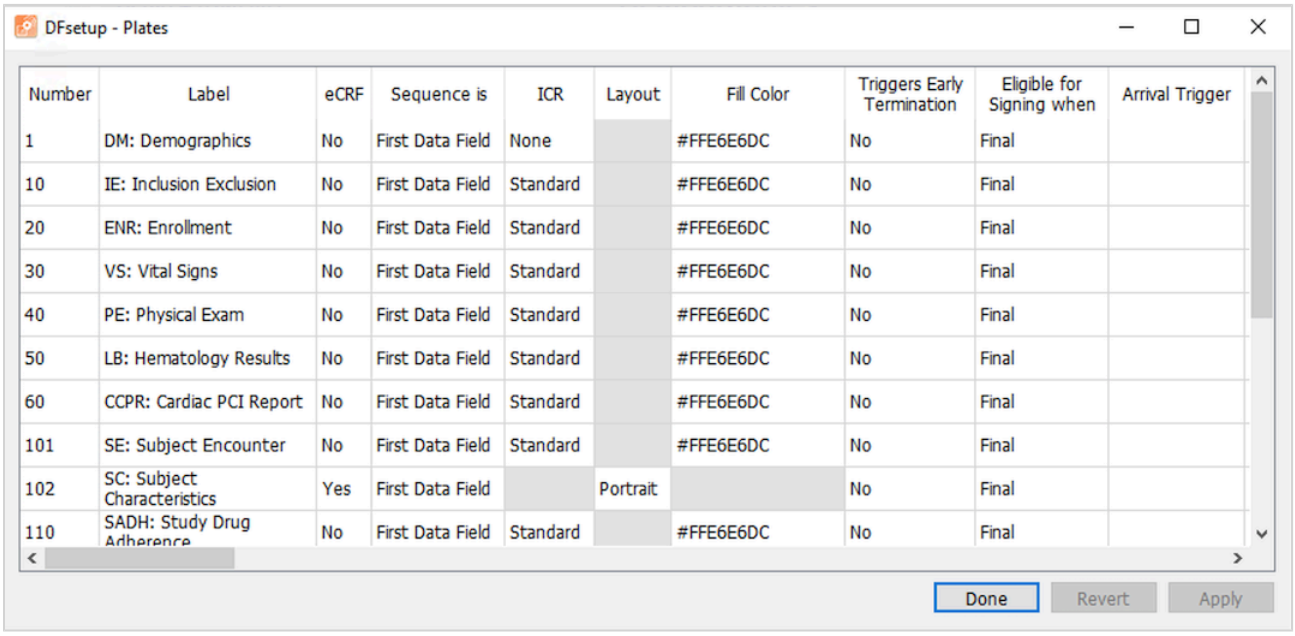
There are several features that must be implemented for proper utilization of an eSignature in DFdiscover.

1. The eSignature module must be added to each plate that requires signing.
2. Each of `SigName`, `SigDate`, `SigTime` and one of either `SigReasonChoice` or `SigReasonString` must be instanced on each such plate. To map the fields to the CRF requiring eSignature, the plate must be designed to accommodate it's use.

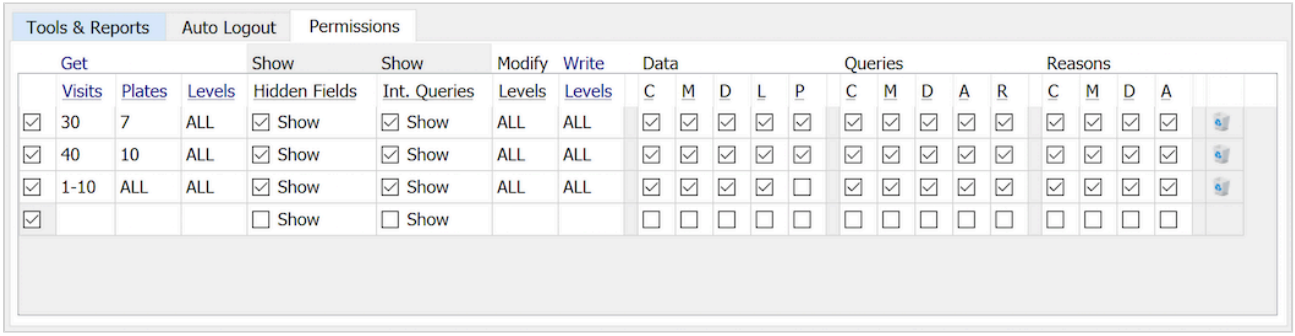


   For an eCRF plate, it is sufficient to drag the needed fields from the eSignature module to the plate definition.

3. In the plate properties dialog, the plates that require eSignature must also have the **Eligible for signing when** choice set to one of **Final** (the default setting) or **Final or Incomplete**. Either choice is acceptable - the proper choice will be defined by the needs of the study workflow.



4. eSignature permission must be granted by an administrator. To be able to sign a page, the **DFexplore** user must, for whatever role they are using, have the **P** permission checked.



In this example, eSignature permission is granted for plates 7 and 10, as defined with two lines in the permissions part of the role definition.

3.12.2. Signing with the eSignature Module in DFexplore

When the current plate contains fields from the eSignature module, the logged-in user has eSignature permission, and the **Eligible for Signing when** condition is met for the status of the current record, the user will be required to sign the record whenever there are changes and they save it. To sign the record, the user is prompted for their username and password credentials on the first record signing during a given **DFexplore** session and then just their password on subsequent signings in the same **DFexplore** session.

Once the credentials are accepted, the eSignature fields are automatically populated. Any changes to the data on a signed plate will remove the eSignature (by blanking the eSignature fields) and will again require signing by a user with permission to e-sign when the eligiblity conditions for signing are met.

If the credentials are not accepted in a **DFexplore** session, the system will re-prompt for the username and password to be entered. Multiple consecutive attempts to sign with invalid credentials will cause the signing to fail, the user to be logged out and a notification email to be sent to the administrator. The specific number of consecutive attempts is determined and set by the administrator.

Chapter 4. Field Menu

This chapter describes the functions located under the **Field** menu.

4.1. Copy

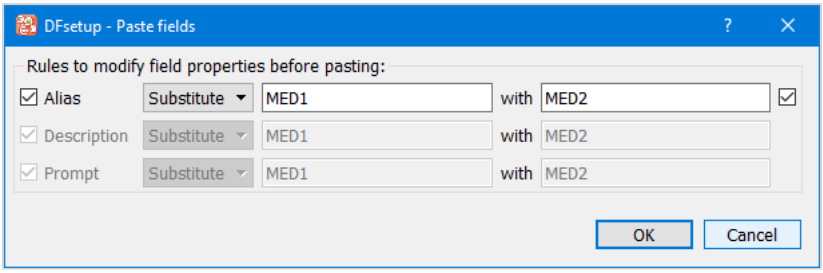
One or more fields can be copied and pasted to create new fields on the same or different plates. To select a single field, click it in the CRF window. To select multiple fields, hold the shift key while clicking each field to be copied, or while holding the shift key, click and drag a bounding box over the desired fields. To select all fields on the current plate use **Field** > **Select All**.

After selecting fields, use **Field** > **Copy** to copy them to the **DFsetup** clipboard. The size, properties, and screen location of the selected fields are all included in the copy.

The clipboard holds only one copy at a time. Making a new copy overwrites any previous copy. Copies are not preserved across login sessions.

4.2. Paste

To create new fields, the most recent copy can be pasted from the clipboard to any plate. When **Field** > **Paste** is selected the following dialog appears which allows you to make changes to the copied names and descriptions while pasting the new fields.



In this example, all instances of 'MED1' in the copied fields properties are replaced with 'MED2' in the pasted fields.

Before clicking OK to paste the new fields onto the current plate, the new field names and descriptions can be modified from the copied values by:

- pre-pending something to the current value,
- appending something to the current value, or
- substituting text in the current value with new text.

Making changes in this dialog is optional and a warning appears after clicking OK if you have not created unique field names using the prepend, append or substitute feature.

After clicking OK the pasted fields appear in the CRF window, offset slightly from the position of the copied fields, and highlighted to indicate that they are selected. The pasted fields can then be moved to the desired location on the page by clicking them, holding the mousebutton, and dragging them as a set. After releasing the mousebutton, click anywhere in the CRF window to end the paste operation, or if you want to cancel the operation at this point select **Field** > **Delete** to remove the pasted fields.

After pasting, the new fields may be selected to modify field properties or change tab order.

4.3. Delete

One or more fields on the current plate can be deleted by selecting them in the CRF window and then selecting **Field** > **Delete**. A confirmation dialog appears. If delete is confirmed, the fields are removed from the CRF window. The change is made permanent the next time you select **File** > **Save** to save all pending **DFsetup** changes to the DFdiscover server.

4.4. Select All

Use this option to select all fields on the current plate.

4.5. Add/Edit Table

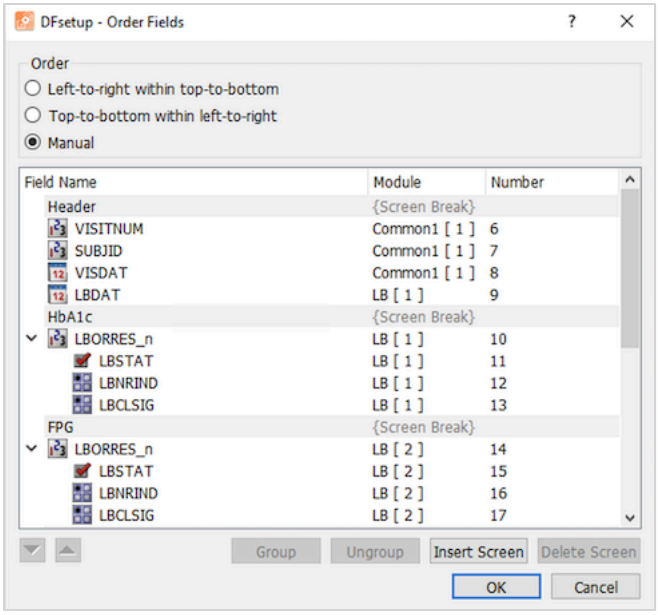
Use this option to add a new table or edit an existing table on the current plate. Tables may be defined on eCRF plates only. See Adding Tables to eCRFs for more details.

4.6. Order

The number that appears in the center of each field widget in the CRF window has the following uses:

- it corresponds to the order in which data fields are visited when using the tab or return keys to traverse fields in **DFexplore**,
- it may be used in edit checks to refer to data fields, and is particularly useful in generic edit checks,
- it may be used to reference data fields in some DFdiscover reports, shell level programs and **DFexplore** views, and
- it corresponds to the order in which data fields are stored in the data and audit trail records in the study database.

To change the field order on the current plate select **Field** > **Order**.



This dialog allows moving fields up/down using either the mouse in a drag and drop operation or using the up/down arrow buttons. For plates with CRF backgrounds, fields may also be ordered automatically according to their placement on the CRF background, either left-to-right within top-to-bottom or top-to-bottom within left-to-right.

Fields can also be grouped/ungrouped using the buttons as labeled. Multiple fields must be selected in the field list before **Field** > **Group** can be selected. Similarly, at least one grouped field must be selected to enable **Ungroup** menu item. Drag and drop using the mouse can also be used to add/remove fields to a group.

Screens provide the software with information about the ideal place in the eCRF to split the screen content to accommodate viewing on different screen sizes in **DFcollect** and **DFweb**.

To add screens, select the field that should begin the next screen, and click Insert Screen. The screen break can be titled, which could be used as a header for different sections in an eCRF. Screen name can be edited by selecting a screen item and double-clicking with the left mousebutton. Any screen without fields is discarded without warning.

Click OK to apply order, grouping and screen changes, or Cancel to make no changes.

4.7. Group

Two or more neighboring fields on the current plate can be grouped by selecting them and selecting **Field** > **Group**. A circle with the order number of the first field in the group appears to the right of each field widget included in the group.

Grouping fields in an eCRF allows for the visual grouping of multiple check or choice type fields. Grouping is ideal when there are several related fields that have code labels. The Prompt of the first field in the group is used as the prompt for the group by default, and the field prompts for all other fields are not displayed.

4.8. Ungroup

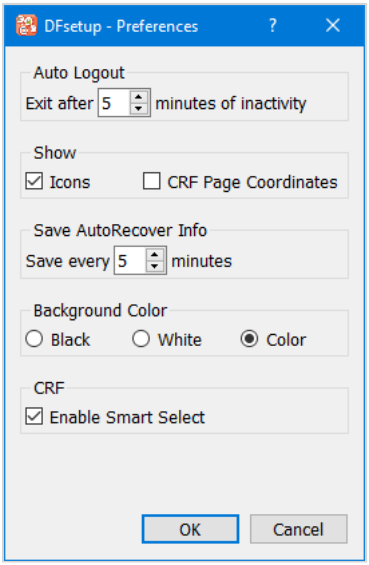
A group of fields can be ungrouped by selecting any field in the group and then selecting **Field** > **Ungroup**.

Chapter 5. File Menu

This chapter describes the functions located under the **File** menu.

5.1. Preferences

While choices made in **Study** > **Global Settings** are study specific, saved on the DFdiscover study server, and apply to all users regardless of what computer they use to run **DFsetup**, choices made in **File** > **Preferences** are user specific, saved on the computer from which **DFsetup** is run, and apply to all studies that the user can access. These user preferences can be changed at any time.

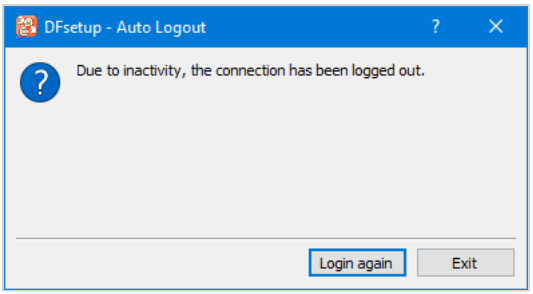


In this example:

- **DFsetup** exits after 5 minutes of inactivity.
- Field type icons are displayed in style, filed, module and plate lists.
- A recovery (auto save) file containing any unsaved changes is created every 5 minutes.
- The CRF backgrounds appear in their original color.
- Smart Select accelerates field definition by automatically selecting horizontally adjacent CRF boxes when any single CRF box is selected.

Auto Logout. For security, **DFsetup** includes an auto logout feature that terminates the DFdiscover server connection after a specified period of inactivity. Values between 1 and 60 minutes may be selected.

If a **DFsetup** session does auto logout a dialog that allows the user to re-connect easily is presented.



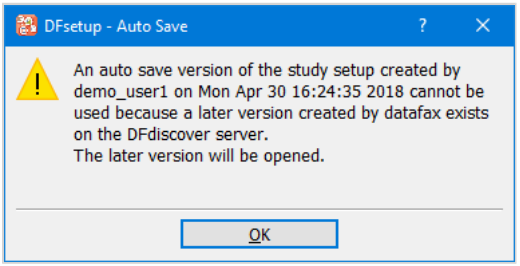
To login again the user must re-enter their password for the DFdiscover server and then reselect the study.

Show/Hide. Data type icons and CRF coordinates can be independently displayed or hidden. Their visibility has no impact on the study setup - they are provided only as optional guides.

Save AutoRecover Info. Under normal circumstances, selecting **File** > **Save** sends all changes to style and field definitions, plate properties and global settings to the DFdiscover study server. It is wise to save work frequently to avoid losing setup modifications because of an auto logout, a dropped internet connection, a power failure, or a computer system crash.

For additional protection, **DFsetup** does, at the specified time interval, create an auto save file for the current study setup, edit checks and lookup tables, in a temporary directory on the local computer. The auto save file is created and updated whenever there are changes that have not yet been saved to the study server. An auto save file is also created, if there are any unsaved changes, immediately before an auto logout. The auto save file is removed when a **File** > **Save** is successfully completed.

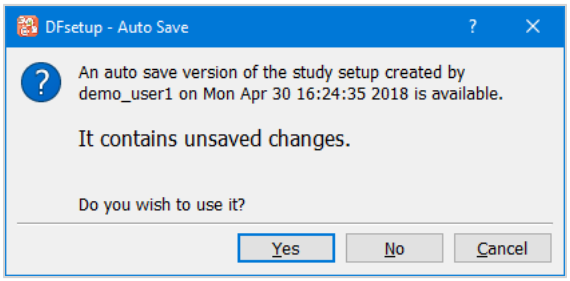
If the computer you use to run **DFsetup** has an auto save file for the specific study, then regardless of who created it, **DFsetup** compares the auto save file with the version of the study setup saved on the server. If the version saved on the server is newer than the auto save version, a warning message is displayed and the auto save file is removed.



This warning appears if an auto save version exists but is out of date.

If the auto save version of the study setup is more recent than the version saved on the study server a dialog appears allowing the user to choose which version should be used.

- If Yes is selected the study setup is loaded from the local auto save file. Remember to choose **File** > **Save** to commit the changes back to the server. At that time the auto save file is also removed.
- If No is selected the study setup is loaded from the study server and the auto save file is removed.
- If Cancel is selected login to the study is aborted and the auto save file remains in place.

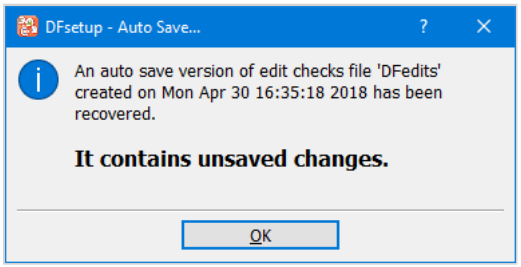


Warning

The auto save mechanism preserves modifications to style and field properties, plate options and global settings, all of which are stored in files `DFsetup` and `DFfile_map`. It also saves unsaved changes to edit check source files and lookup tables. It does not save modifications to other setup components (sites, visit map, etc.), which are saved in separate configuration files immediately upon selecting Save in their respective dialogs.

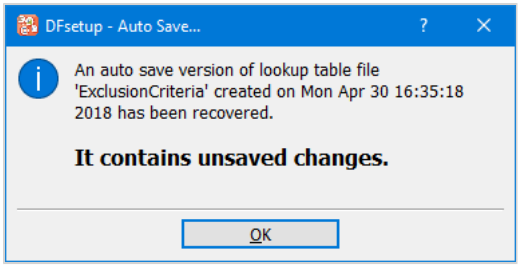
If an auto save version of an edit check source file exists on the local computer, an auto save file recovery notification appears and the file is recovered as a new unsaved file. This notification appears when the edit checks view is opened. The recovered file can be renamed, saved or deleted. Renaming the file follows standard file naming restrictions. Since the default behavior is to prevent renamed files from overwriting existing files, to complete recovery of an edit checks file:

1. Open the recovery file and review the contents. Confirm that the unsaved changes are present.
2. Select the full contents of the file in the editor dialog. Copy the contents.
3. Close the recovery file.
4. Open the original file.
5. Paste the copy contents in the original file, replacing the file contents.



This notification appears if an auto save version of an edit check file exists.

If an auto save version of a lookup table file exists on the local computer, an auto save file recovery notification appears and the file is recovered as a new unsaved file. This notification appears when the lookup table editor is opened. The recovered file can be renamed, saved or deleted. Renaming the file follows standard file naming restrictions.



This notification appears if an auto save version of a lookup table file exists.

Auto save is local

The auto save file recovery notification for edit checks and lookup tables is only triggered when the respective views are opened on the computer that has the local auto save files.

Background Color. Imported CRFs are displayed as backgrounds on which the data field definitions are created. The background color attribute can be changed with this preference.

CRF. Enabling Smart Select allows the user to select multiple CRF boxes with a single click if they are horizontally adjacent. When any such box is selected, all adjacent boxes are automatically selected and ordered from left-to-right to ensure proper layout and ICR.

5.2. Link

Two DFdiscover studies may be linked with one defined as the development study and the other as the production study. When a link is created the production study setup is locked and cannot be changed except by published changes from the development study. This allows the development study to be used to make and test setup changes before implementing them in the production study.

Because it is very restrictive Developer access mode should only be used to perform the functions listed under the **File** > **Link** menu item (described below), and all changes to the development study should be made using Exclusive, Normal or Configuration access modes.

To perform the functions available from the **File** > **Link** menu in Developer access mode users must be a DFdiscover administrator or study administrator for the current study and have at least 'Setup-Plates' permission for both the development and production studies.

The **File** > **Link** actions include:

Create Production Study.

- This option is typically used when setup and testing has been completed for a new study and you are ready to publish the setup to the DFdiscover study (the "production" study) that is used to run the study.
- Before selecting this function the new production study must be created in **DFadmin** and have at least one role.
- Candidate studies are displayed in a selection list.
- Only new study databases (i.e. `DFsetup` not yet defined) for which the user has 'Setup-Plates' permission appear in the list of candidate studies.
- When a study is selected all setup components are copied from the current study to the selected production study.
- No data files, roles or users are copied - just the setup.
- The setup for the production study becomes locked and is available in **DFsetup** view mode only.
- The current study becomes linked as a development study for the selected production study, and all **DFsetup** access modes remain available.

Create Development Study.

- This option is typically used when you need to perform setup changes and testing for a study that is already in production, where the changes are extensive enough that it would be unwise or impossible to make them directly to the production study setup.
- Before selecting this function the new development study must be created in **DFadmin** and have at least one role.
- Candidate studies are displayed in a selection list.

- Only new study databases (i.e. `DFsetup` not yet defined) for which the user has 'Setup-Plates' permission appear in the list of candidate studies.
- When a study is selected all setup components are copied from the current study to the selected development study.
- No data files, roles or users are copied - just the setup.
- The setup for the development study remains unlocked and all **DFsetup** access modes remain available.
- The current study becomes linked as a production study for the selected development study, and its setup becomes locked. The user must thus logout of **DFsetup** and can subsequently only login using View mode.

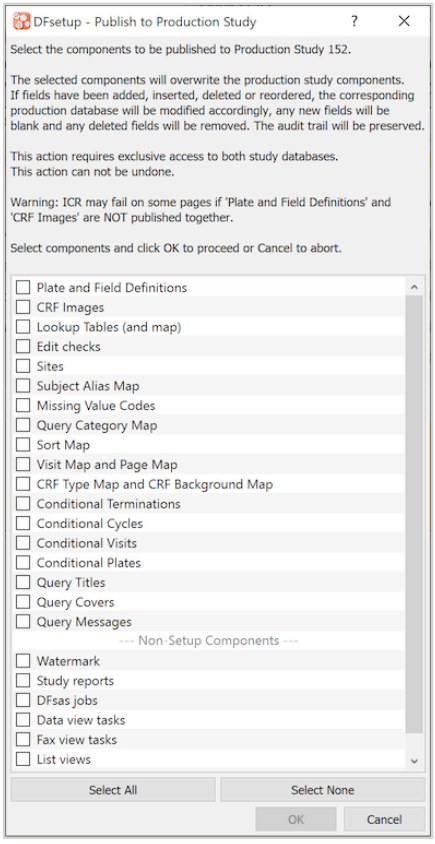
Review Development Changes.

- This option can be used to see the differences between the current development and production study setups. Only plate level changes are displayed, including changes to custom level labels and style and field properties. Changes to configuration files are not currently displayed.
- When this option is selected a dialog appears listing the changes. The results can be printed or saved to a file.

Publish.

- This option is used to copy all or selected setup files from the development study to the production study, including: 'Plate and Field Definitions', 'CRF images' (data entry backgrounds), custom level labels, and each of the configuration files. Publishing can only be successful if both the development and production studies have the same DFdiscover version of the study setup.

  Non-setup components (such as Study reports, Data view tasks, List view) can be selected separately and are not included in the Select All action.



> **Warning**
>
> CRF images and Plate and Field Definitions should always be published together. Publishing one without the other may result in inconsistencies between data entry backgrounds and their corresponding data entry fields.

- Users require 'Setup-Plates' permission on the production study to publish 'Plate and Field Definitions' or 'CRF images', and require permission for each of the configuration files to be published.
- When publishing the CRF background images any extraneous files that exist in the production study `bkgd` directory are removed. This ensures that any backgrounds previously tagged by visit number do not continue to be used by **DFexplore** if visit tagging is dropped in the development study.
- When publishing changes the files to be replaced in the production study must be owned by `datafax`. Problems may be reported for directory `ecbin` which contains scripts that can only be created by a UNIX login account. Files in this directory owned by `datafax` are published, but any files in the production study owned by other users cannot be replaced by `datafax` and thus must be managed manually.
- If 'Plate and Field Definitions' are published and they include critical changes to any of the existing study plates (i.e. new, deleted or reorder data fields), the affected database files are automatically restructured on the production study. This is triggered when the study server determines that a new `DFschema` file has been created that includes critical changes. Any new fields are added to existing data records as blank fields, any deleted data fields are removed, and reordering of data fields occurs if needed to make the final data record structure agree with the specifications for each plate in the revised study setup.
- Because database restructuring is required users must have exclusive access to the production study when publishing plate and field definitions that contain critical changes.
- If database restructuring is required remember to review and correct any edit checks, custom programs, **DFsas** jobs, and **DFexplore** tasks, views and report lists that may be affected, as these changes cannot be automated.

Revert to Production.

- This option can be used to discard all setup changes to the development study and return it to the current production study setup. Both the development and production studies must have the same DFdiscover version of the study setup to perform this operation.
- This cannot be undone.

Unlink.

- This option breaks the link between a pair of development-production studies.
- When the link is broken the 2 studies become regular standalone DFdiscover studies.
- Once broken the link cannot be re-established because a link can only be created to a new study (i.e. one without an existing `DFsetup` file).
- However, a new development study can be easily created when needed for any production study using **Link** > **Create Development Study**. It can then be used to make, test and publish setup changes, and can be discarded by breaking the link and deleting it in **DFadmin** when it is no longer needed.

5.3. Print

The following study setup documentation can be printed.

- Annotated CRFs - prints the CRF image for all or specified plates and annotates each data field with the field number and name. If the name is longer than the bounding box for the field, checking **Entire field name** includes the full name without truncating it at the bounding box. If there are multiple backgrounds defined in the **CRF List**, and annotated CRFs for each background are needed, uncheck **Default pages only**.
- Plate Field Properties - prints field properties for all or specified plates. Prints either a compact report containing just those properties defined at the field level, or a longer report containing all field properties, whether defined at field, module or style levels.
- Module Properties - prints field properties for all or specified modules. Prints either a compact report containing just those properties defined at the module level, or a longer report containing all field properties, whether defined at module or style levels.
- Style Properties - lists the field properties that are locked in each style.

5.4. Save as PDF

The following study setup documentation can be saved as a PDF file.

- Annotated CRFs - saves the CRF image for all or specified plates and annotates each data field with the field number and name. If the name is longer than the bounding box for the field, checking **Entire field name** includes the full name without truncating it at the bounding box. If there are multiple backgrounds defined in the **CRF List**, and annotated CRFs for each background are needed, uncheck **Default pages only**.
- Plate Field Properties - saves field properties for all or specified plates. Saves either a compact report containing just those properties defined at the field level, or a longer report containing all field properties, whether defined at field, module or style levels.
- Module Properties - saves field properties for all or specified modules. Saves either a compact report containing just those properties defined at the module level, or a longer report containing all field properties, whether defined at module or style levels.
- Style Properties - saves the field properties that are locked in each style.

5.5. Save

The **DFsetup** title bar displays '[Save Required]' if there are changes to the definition of styles, data fields, plate options or global settings, which have not yet been saved on the DFdiscover server. Selecting **File** > **Save** immediately rewrites these files: `DFsetup` and `DFfile_map` (used by **DFexplore**), `DFschema` and (used by reports), `DFtips` (used by the ICR software) and the background images used by **DFexplore** (if any field widgets have been moved to new locations).

Modifications to the other setup components (configuration files such as sites, visit map, edit checks, etc.) are saved separately within their individual dialogs.

If **File** > **Save** is selected and there are unsaved changes in any open plates, styles, fields, edit checks, etc. dialog windows, a message dialog appears with options to save or discard the changes in any or all of the open windows by selecting/de-selecting the desired components from the list of unsaved changes.

Typically modifications to a study in progress are only made after putting the study into restricted access mode and waiting for all users to logout. If changes are saved without doing this, setup changes are immediately available to users who connect to the study server after the save has completed, and users who are already connected to the study continue to see the

previous data entry screens, field definitions, edit checks, etc. with one exception: any reports run in Reports View use the updated schema files.

Warning

Before saving changes to a study that is in progress make sure you have read and understood the warnings and procedures described in Modifying Plates that Contain Data.

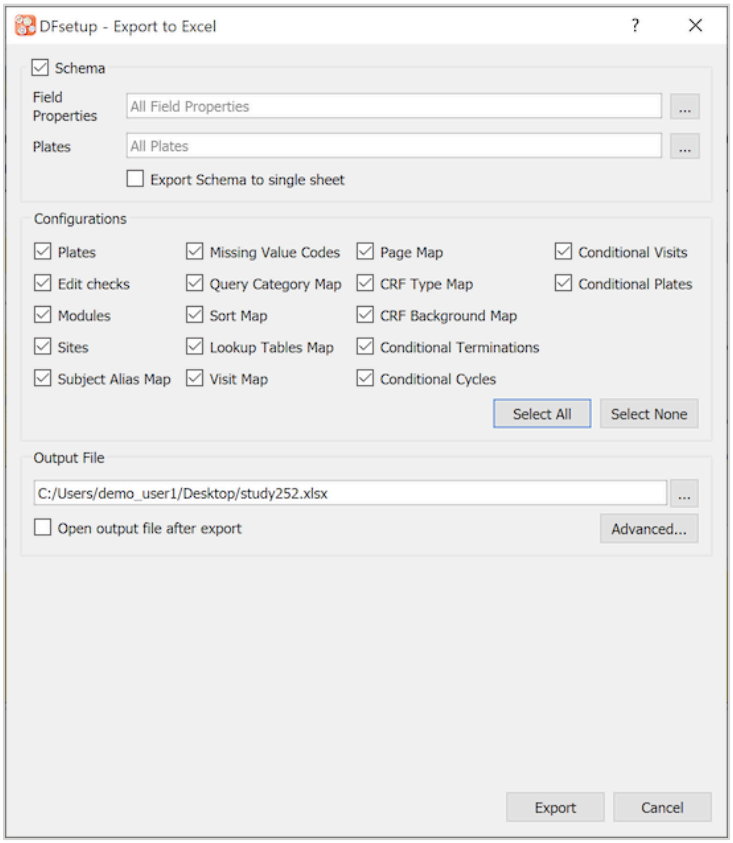
5.6. Export Setup

Use this option to export study setup (Plates, Modules, Fields and Styles properties) in Excel, XML or JSON format. An option is available for exporting study setup in XML format without modules (to keep legacy XML output style offered in previous versions).

5.6.1. Excel Output

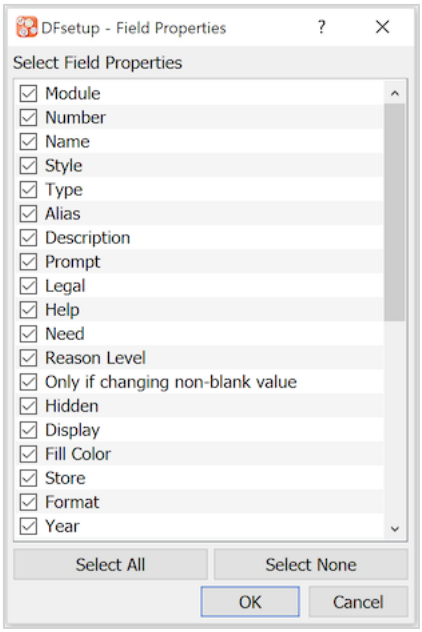
The study setup can be exported to an Excel workbook. The workbook contains a worksheet for the field definitions in every plate, plus all of the additional configuration information such as the visit map, edit checks and the sites definitions. Inclusion of each worksheet is optional and is specified in the dialog for defining Excel exports.

Select **File** > **Export Setup Excel...** to display the dialog for defining Excel exports.



An export definition has three main sections:

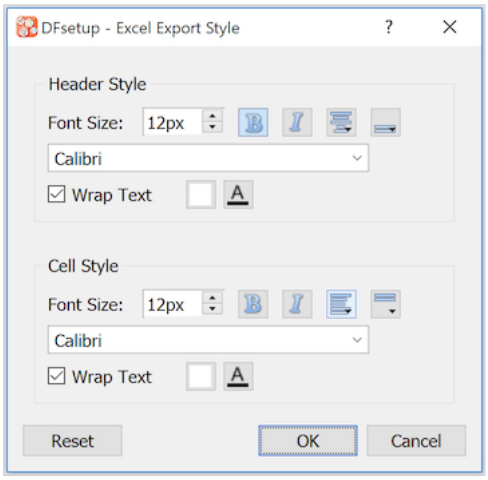
- **Schema.** Indicate which field properties and plates are exported. By default, each plate definition is written to a separate worksheet. Select **Export Schema to single sheet** to override this default and write all of the plate definitions to a single worksheet. All field properties can be included, or a subset can be chosen from the sub-dialog available when clicking ....



Similarly, all plates can be included, or a subset chosen from the sub-dialog, or by entering a single plate number, list of plate numbers or range of plate numbers.

- **Configurations.** Many of the specifications defined in the **View** menu dialogs can be exported. Check the desired specifications to be included. Each is written to a separate worksheet in the workbook.
- **Output File.** The Excel workbook is written to the specified output file. Check **Open Output File after export** to start Excel after the output file has been created and display the workbook.

The Excel export function includes definition of basic Excel formatting specifications. Click Advanced to display the **Excel Export Style** dialog.



Settings in the dialog allow customization of format for the header and body cells of each worksheet.

5.7. Verify All

Select **File** > **Verify All** to run the standard DFdiscover reports that check the current study setup specifications. The results appear in a dialog which shows any warning or error messages for field and style properties, visit scheduling specifications entered in the visit and conditional maps, and clinical site specifications entered in the sites dialog.

NOTE: the setup verification reports check the setup files stored on the DFdiscover server. They do not have access to any pending changes that have not yet been saved.

5.8. Review Changes

Use this option to review changes you have made but not yet saved to plate, style and field definitions. This option does not include any unsaved changes in configuration files: DFedits, Visit Map, etc.

5.9. New Study

A new study session can be started without terminating the current one. This can be useful when comparing setup specifications for different studies.

5.10. Close Study

This option closes the current study and returns to the study selection dialog where another study could be selected. If there are unsaved changes in the current study a warning dialog appears with options to save or discard the changes, or cancel the close request.

5.11. Exit

This option closes the current study and then terminates **DFsetup**. If there are unsaved changes in the current study a warning dialog appears with options to save or discard the changes, or cancel the exit request.

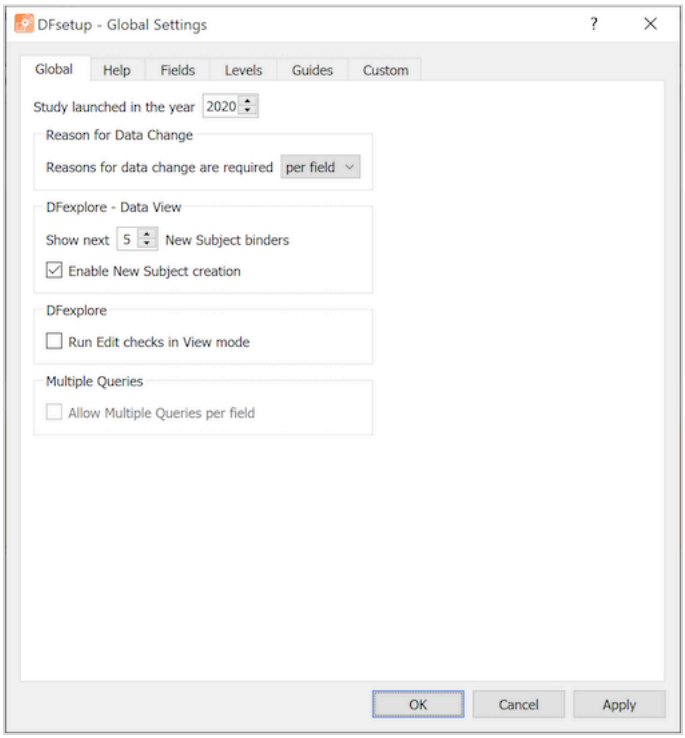
Chapter 6. Study Menu

This chapter describes the functions located under the Study menu.

6.1. Global Settings

Global settings have an impact on subsequent work performed in **DFsetup** and thus should be considered first. Different global settings can be entered for each study.

Select **Study** > **Global Settings** to access the **Global Settings** dialog.



In the dialog, settings are grouped by purpose and presented in separate views under the tabs **Global**, **Help**, **Fields**, **Levels** and **Guides**.

6.1.1. Global view

Study launched in the year. The study launch year is used by some DFdiscover standard reports and is available in DFschema for study specific reports.

Reason for Data Change. Users may be required to enter reasons for data change **per field** (specified at the field level), **always** (for all fields), or **never**.

- **per field**, the reason specification set at the style or data field level is applied,
- **always**, a reason is required for all data fields for any changes made to records at or above the workflow level specified.

  If **always** is chosen, two additional options, **Level** and **Only if changing a non-blank value** become available. The value for **Level** overrides the reason level defined at the style or field level. By default, the value is set to 'Level 1'. By checking **Only if changing a non-blank value** a reason for data change is required only if the changed value was not blank prior

to the change, and its record existed in the database at or above the level specified. By default, this option is unchecked, so a reason for data change is required for changes to all data fields regardless of their original value.

- **never**, the style and data field reason specifications are ignored and a reason is never required, with one exception. If a field already has a reason for the current data value, then any change to the current value must be accompanied by a new reason. This rule applies regardless of the reason specifications at the global, style and data field levels.

DFexplore - Data View. If no New Subject Binders are displayed and the Start New Subject dialog is disabled, the only way a new subject can be started is in Image View, either when entering data from a faxed/scanned CRF or when entering a set of records in EDC Data Entry mode.

Run Edit checks in View mode. This global setting is required if you want edit checks to do something, when a user is working in view only mode or when a record is locked by another user. For example, you might enable this option so you can display a message, set user preferences in `DFopen_study`, or hide certain visits and plates in `DFopen_patient_binder`. If this option is enabled we recommend terminating any edit checks that would make changes to data or metadata when users are working in view only mode by including the following statement before statements that change data or metadata.

```
if( dfmode()=="view" || dfmode()=="locked" ) return;
```

Otherwise, users see a warning message stating that a query could not be added, or a field could not be changed.

Allow Multiple Queries per field. This global setting is required if you want to add multiple queries on a single field. Once this option has been enabled, it cannot be disabled.

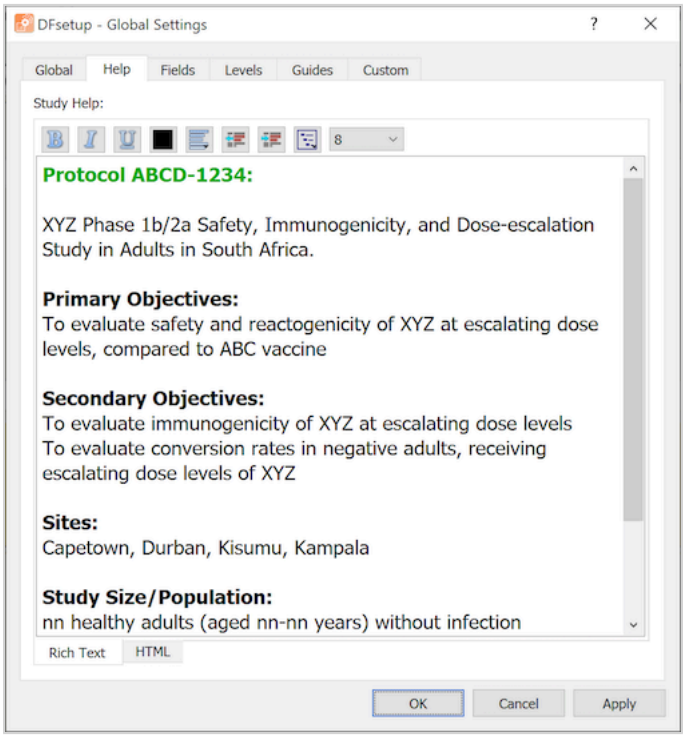
6.1.2. Help view

Help can be defined at the study level, plate level and field level. At each level, help can be defined in plain text, rich text or HTML formats.

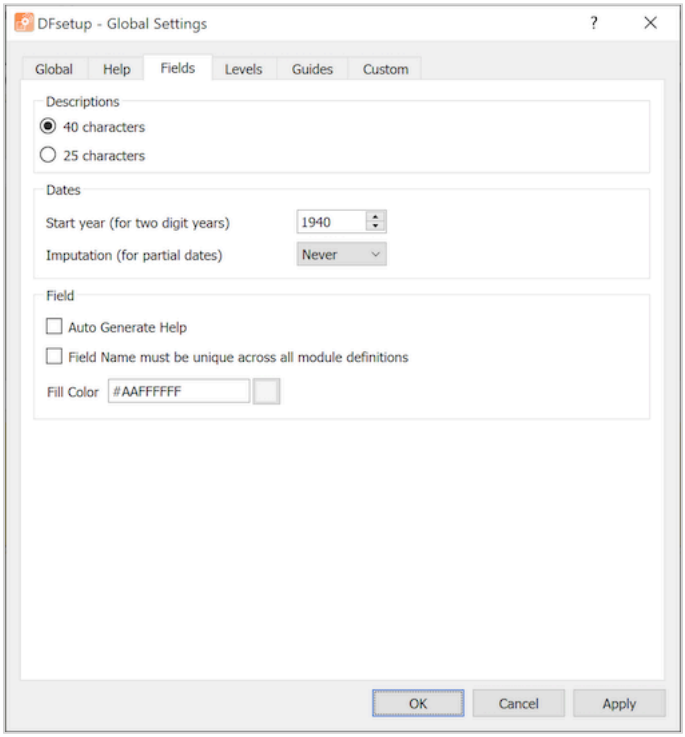
Different formats can be used but they cannot be mixed within the same property. For example, a field can have rich text or HTML help, but it cannot have both. Another field or plate can use a different format. Once you start defining individual help in one format, it is not possible to switch to another format without discarding the existing content and starting over.

To specify HTML help, it may be easier to copy and paste HTML content from a full featured HTML editor. In this view, raw HTML5 is permitted and you can click the preview button () to see the rendered appearance of the HTML.

In this view, study level help is defined. **DFexplore** users see this help in the Dashboard View and when they select **Help** > **Study Help**.



6.1.3. Fields view



Descriptions. Field descriptions may be up to 40 characters long or limited to 25 characters for older versions of SAS® and to create more compact Query Reports. Reducing this setting from 40 to 25 characters after fields are defined does not truncate existing labels in the setup until the field is resaved in **DFsetup**, but oversized labels are truncated in the Query Reports.

Dates. The values specified here for dates, **Start year** and **Imputation**, can be used when defining styles and data fields.

Field. When checked, **Auto Generate Help** applies a default help message, `legal values are: $(legal)`, so that fields may be defined more quickly.

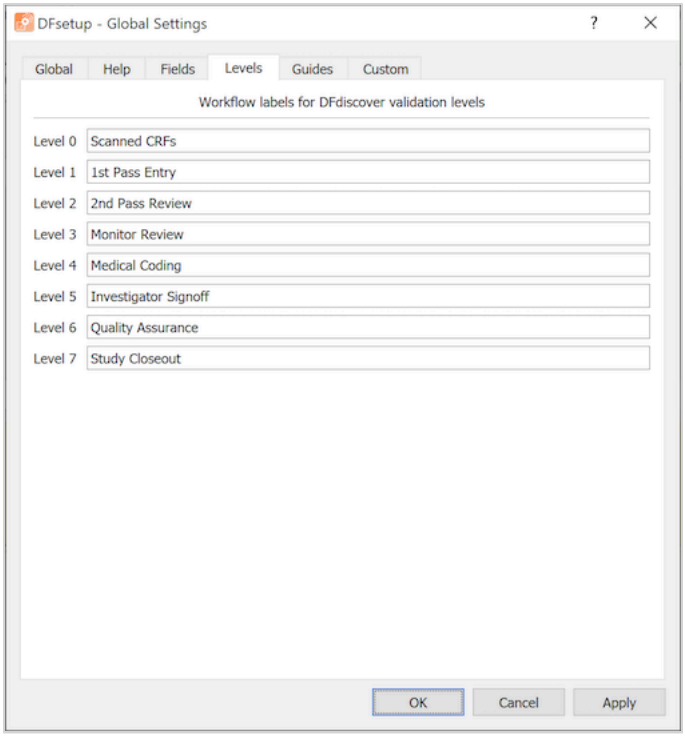
When checked, **Field Name must be unique across all module definitions**, you are warned about any non-unique field names across all modules.

Fill Color is used to paint/fill the difference area between the data entry widget and CRF background color in **DFexplore**. Fill Color defined here is inherited by plates as the default color.

Fill Color has 4 components: alpha (opacity), red, green and blue and is expresed as a hex number with the format #AARRGGBB. Click the square color swatch to select the color (and opacity) from one of several palettes or use the color dropper to sample from any part of the CRF.

6.1.4. Levels view

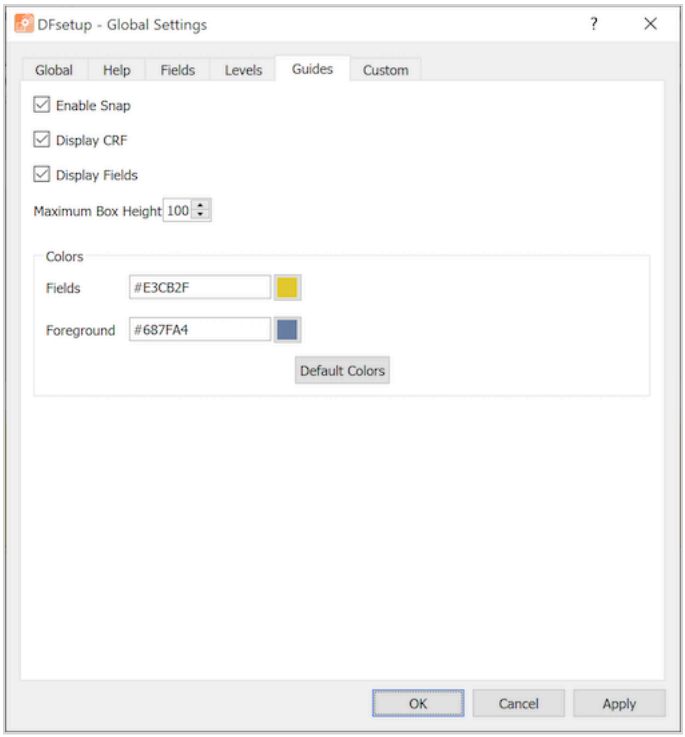
Custom labels can be assigned to each workflow level. Default labels are `Level 0`, `Level 1`, ... `Level 7`.



Labels may be up to 20 characters in length, created from numbers, upper and lowercase characters and symbols. Pipe (|) symbols are not legal and cannot be entered.

Labels can also be left blank when the workflow level associated with the label is not used. Where custom labels are used, they are included in DFschema as attributes for DFVALID, output as WORKFLOWLABEL in xml export file, and appear in **DFexplore** in all views where save levels are specified or displayed.

6.1.5. Guides view



Enable Snap. Data field widgets snap into place when dragged near the boxes defining a data field on an imported CRF page.

Display CRF. When checked, makes the imported CRF visible.

Display Fields. When checked, makes any data fields defined on the CRF page visible.

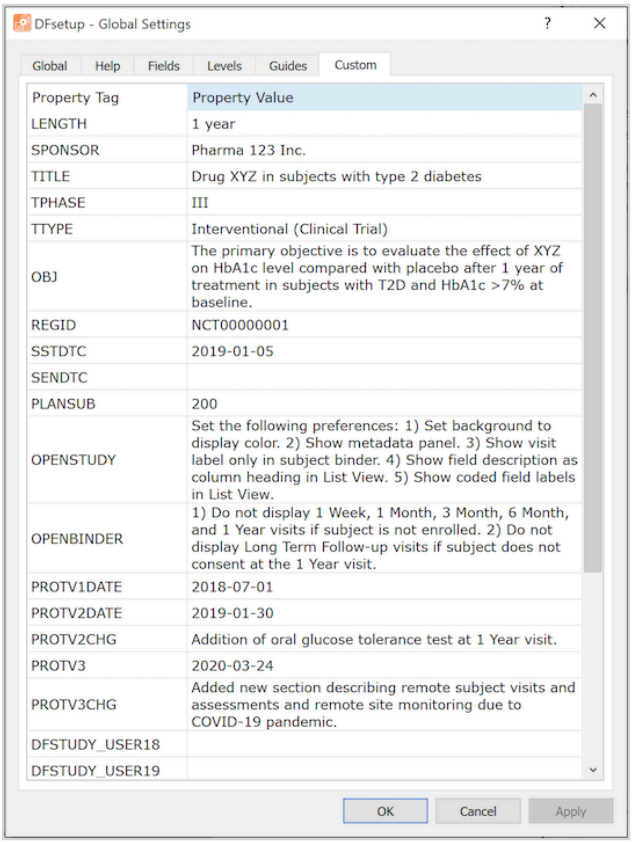
Maximum Box Height. When creating a new data field by dragging out a widget, the widget height is always constrained to the recommended height (25 pixels, about 1/4 inch), but a different constraint can be specified for dragging in a downward direction.

Colors: Fields. Use this color to show the size and location of each data field.

Colors: Foreground. The foreground color is used to print the field number inside field widgets. Ideally, the Fields and Foreground colors have good contrast.

6.1.6. Custom properties view

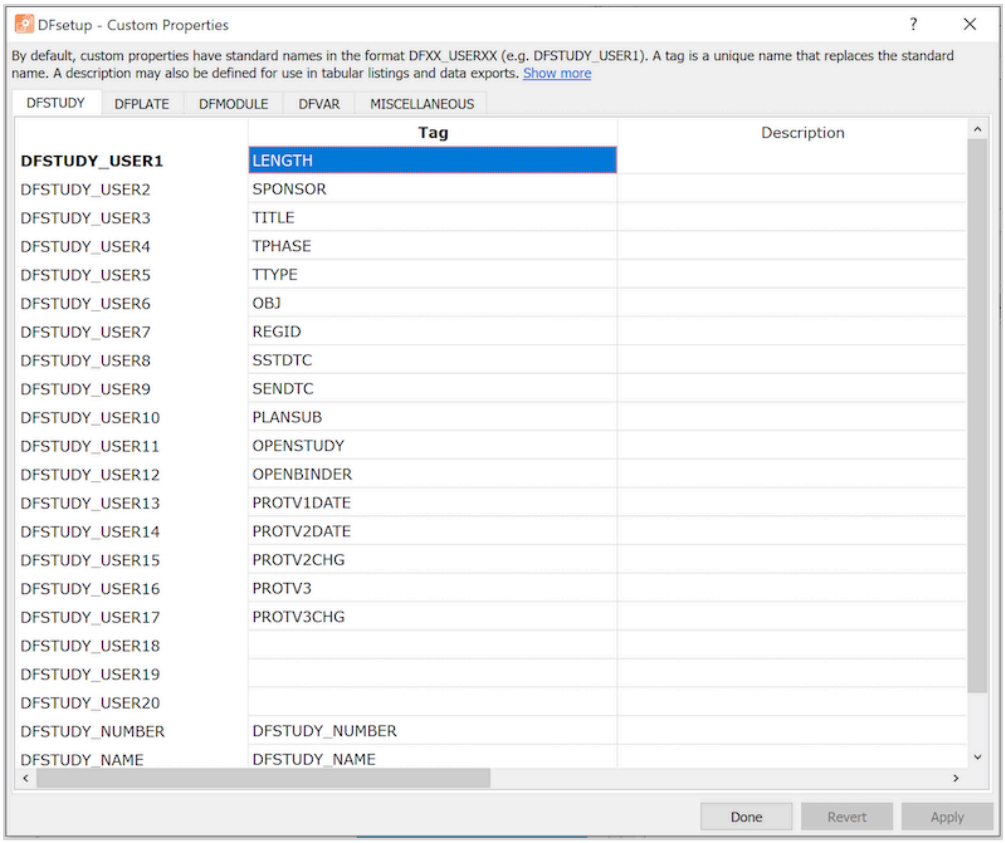
Up to 20 custom properties can be assigned for the study. Each property value is a string and may contain up to 500 characters. The property tags are assigned under Custom property tags.



6.2. Custom property tags

Use this dialog to assign unique custom property tags at the study, plate, module, and field (variable) level. Tags are used as a reference for custom properties, similar to a field alias, and therefore must be unique across all custom properties and field aliases. For custom properties without a tag, the default name (e.g. DFSTUDY_USER1) is used. Tags have the same restrictions as field aliases: cannot start with a number, cannot contain special characters, and can be a maximum of 80 characters. Where tags can be selected and output, an optional Description can be provided for use in the output. [1] This Description serves the same descriptive, readability purpose as the Description property of any database field.

[1] For example, in Tabular Listing reports, the `Display table header as` option controls this appearance.

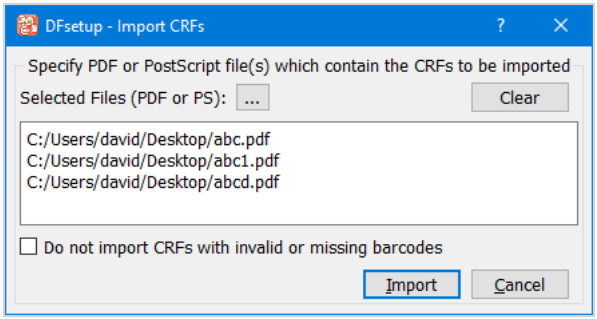


6.3. Import CRFs

Study case report forms (CRFs) can be created in any word processing or graphics package. The data screens used in **DFexplore** are created by importing a copy of the CRFs and then creating data entry widgets over the data fields on each unique CRF page (aka plate). The imported CRF images become the **DFexplore** data entry screens that then look identical to a printed copy of the CRFs.

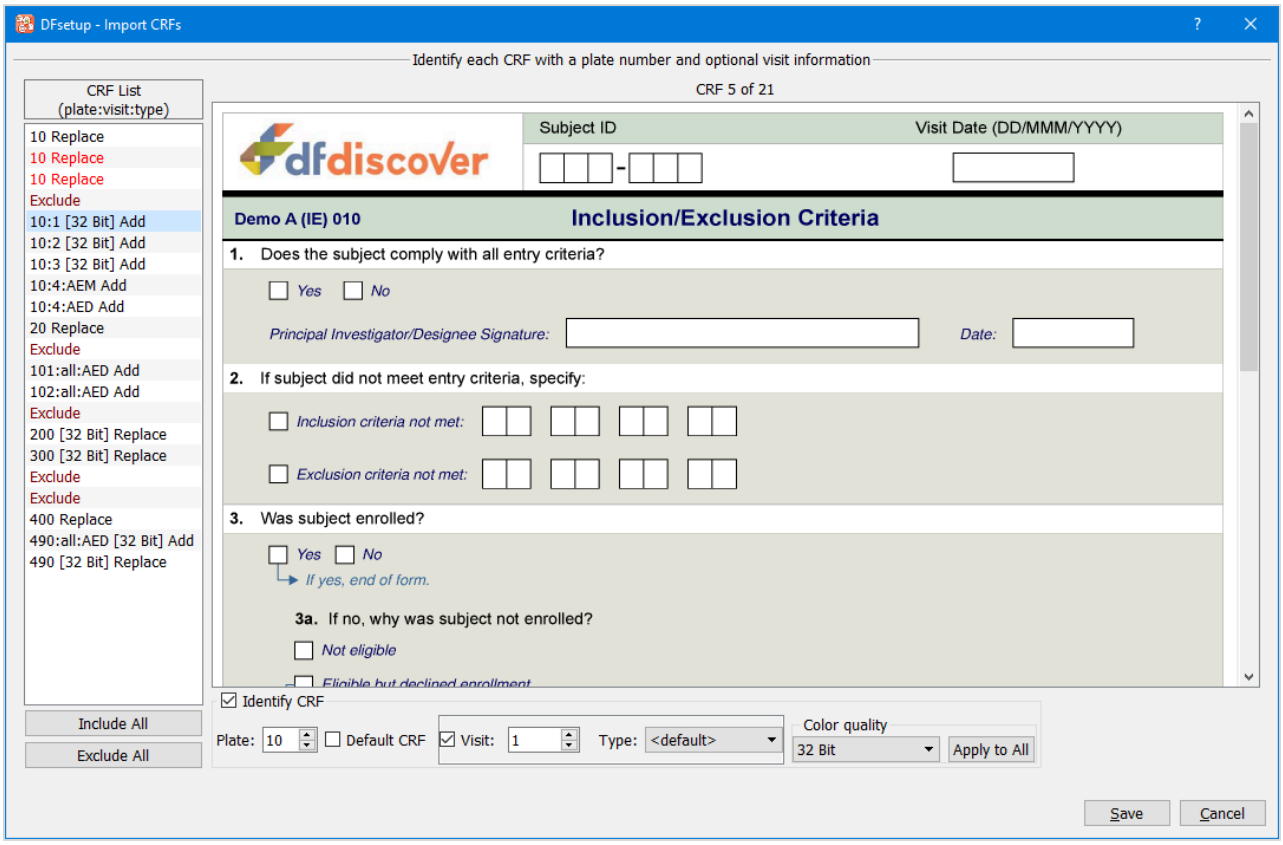
A study setup may include more than one data entry screen for each plate. Each imported CRF page can be tagged with the plate number and optionally also with a visit number and/or a type name. Pages tagged with a visit number are used as the data entry screen for the specified visits, and CRFs tagged with a type (e.g. a language) can be selected as an **DFexplore** user preference. Pages with only the plate number tag are referred to as 'Default' pages, and are displayed when no tagged page is available.

Before importing CRFs that are to be tagged by language, or some other categorization, the CRF Type Map must be defined.



- One or more PDF files may be imported from a folder on the local computer.

- Insert # at the beginning of any entry to skip that file during import.



On importing CRF Pages:

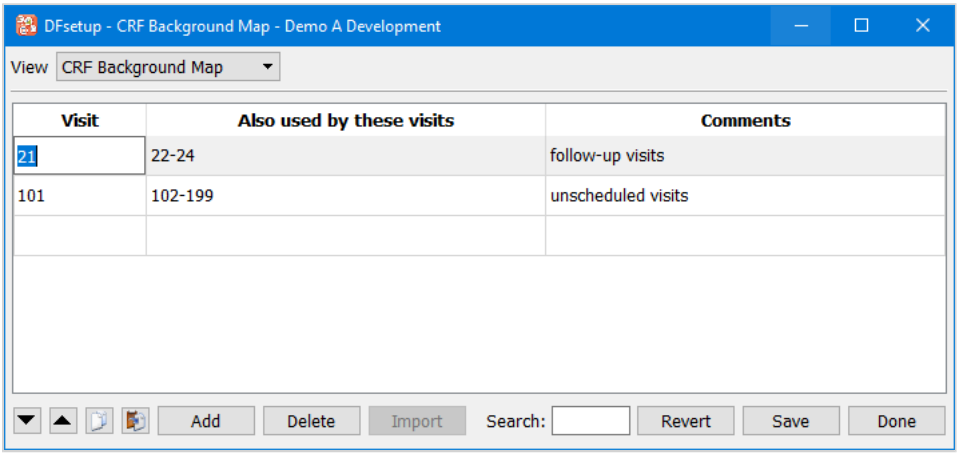
- The file is opened and the barcode (if any) is read on each page.
- Pages are listed by plate number and pending action:
  - Exclude if the CRF Page has no barcode. Excluded plates are not imported.
  - Replace if the CRF Page has already been defined.
  - Add if the CRF Page is new.
- Each page can be marked Exclude or assigned a plate number and optionally tagged by visit number and type, or identified as the 'Default CRF' for the specified plate.
- Exclude All marks all plates as Exclude.
- Include All marks all plates for import that have been tagged with a plate number.
- Only 1 CRF Page can use a given plate, visit and type tag. Repetitions of the same set of tags appear in red and should be marked Discard.
- Keyboard up/down arrows can be used to navigate the CRF List, and the 'E' key can be used to toggle between Exclude and Add or Replace.
- Barcodes are not necessary for any plate that will be entered directly using **DFexplore** and not faxed/scanned. Such CRF pages will be labeled 'Exclude' on import but can be assigned plate number, visit and type tags to identify them.

- Colored CRFs may be imported as plate backgrounds. Color quality may be set to Normal (recommended) or 32 Bit. Once Color quality is set, Apply to All can be used to apply the selected setting to all plates that are tagged for import.

> **Warning**
>
> 32 Bit Color quality should only be used where it is important that color detail in the source document be preserved. Plate backgrounds set to 32 Bit Color quality will require more time to load in **DFexplore**. Normal Color quality is recommended and is suitable for black/white CRFs and those with color graphics and minimal shading.

When the CRF Page specifications are complete, click Save to import the pages. This creates and saves a screen resolution background file for **DFsetup** and **DFexplore** and a higher resolution png file for printing from **DFexplore**. These files are stored on the DFdiscover study server in the study `bkgd` folder.

Although each CRF page can be tagged with only 1 visit number it is not necessary to import and tag a separate page for each visit, only for those that require different backgrounds. A set of visits may share the same background by completing the CRF Background Map.



CRFs tagged with a specific visit number may also be used at other visits by completing this table.

Although multiple CRF pages may be tagged for each plate only one set of data fields can be defined for each plate, and they must appear in the same position on all versions of the data entry screens.

When data fields are positioned on the imported CRF pages to create the data entry screens used by **DFexplore**, any text or graphics hidden beneath the data entry widgets in **DFsetup** are also hidden in the **DFexplore** data entry screens.

It does not matter which version of the CRFs tagged for a plate is used to define the data fields on that plate, but after defining data fields you should toggle between each CRF background to confirm that field positioning works on all versions. If not, you will need to modify one or more of the CRF plates and re-import them.

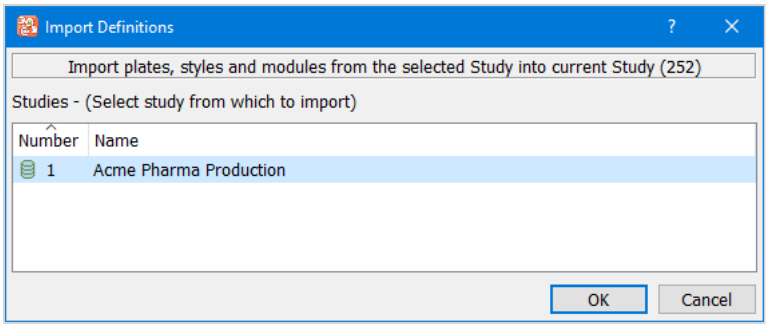
DFexplore selects data entry screens for each plate in the following order:

- CRF tagged with the current visit and type, else
- CRF tagged with the current type only, else
- CRF tagged with the current visit only, else
- the untagged CRF, i.e. the 'default'

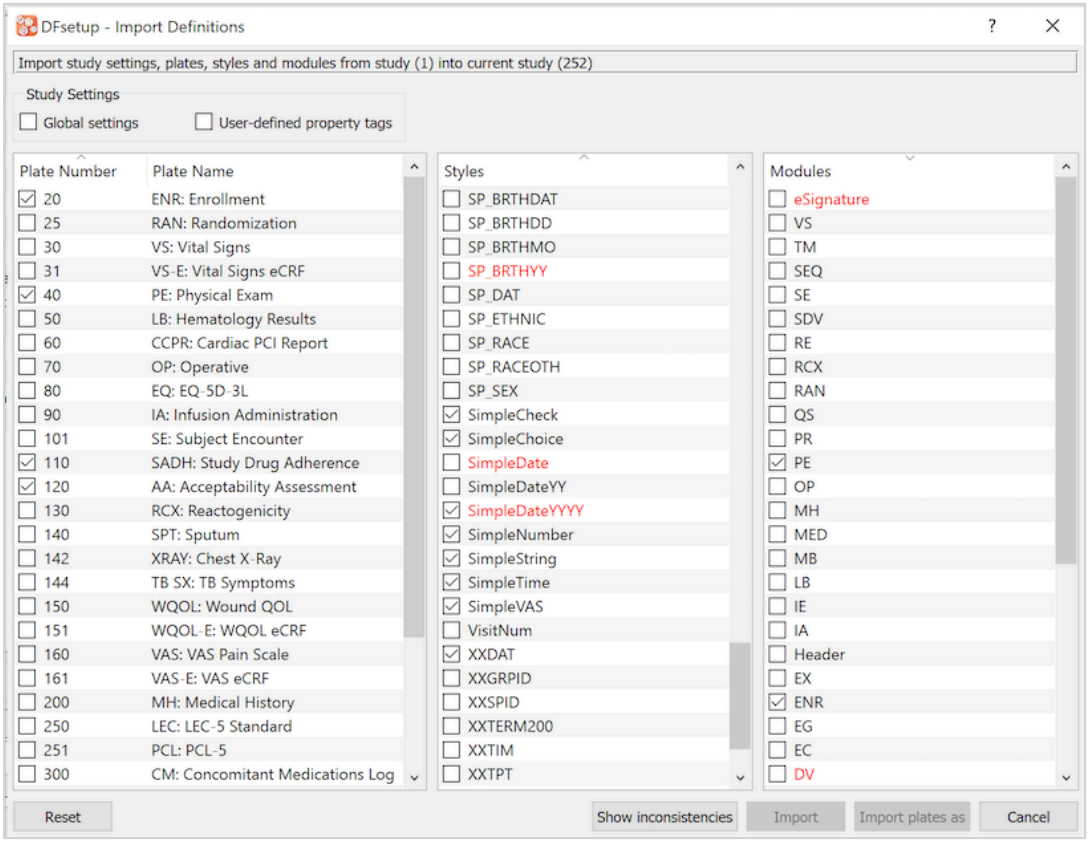
If the last step is reached and no default CRF has been defined for the current plate, the data entry screen displays the data fields on a blank background.

6.4. Import Definitions

Data fields, modules, styles, global settings, and custom property tags can be imported from a central library database or any other study database for which the user has permission to view the study setup.



- This dialog is used to select the DFdiscover database from which you want to import study definitions.



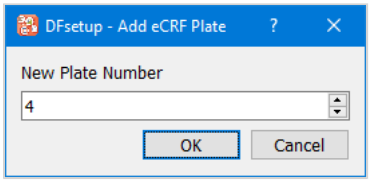
- For each plate selected, styles and modules are automatically selected as well.
- Modules or styles cannot be imported if a module or style with the same name already exists.
- If a style or module is deselected the plate that needs it is automatically deselected.
- Use Import plate as to import a plate to a different plate number in the current study.
- Style or module inconsistencies appear in red. If you want to see what inconsistencies exist, you can mouse over the name (details appear in a tool tip) or click Show inconsistencies if selected.

- Include global settings and/or custom property tags in the import by selecting the appropriate checkbox at the top of the dialog.

6.5. Add eCRF Plate

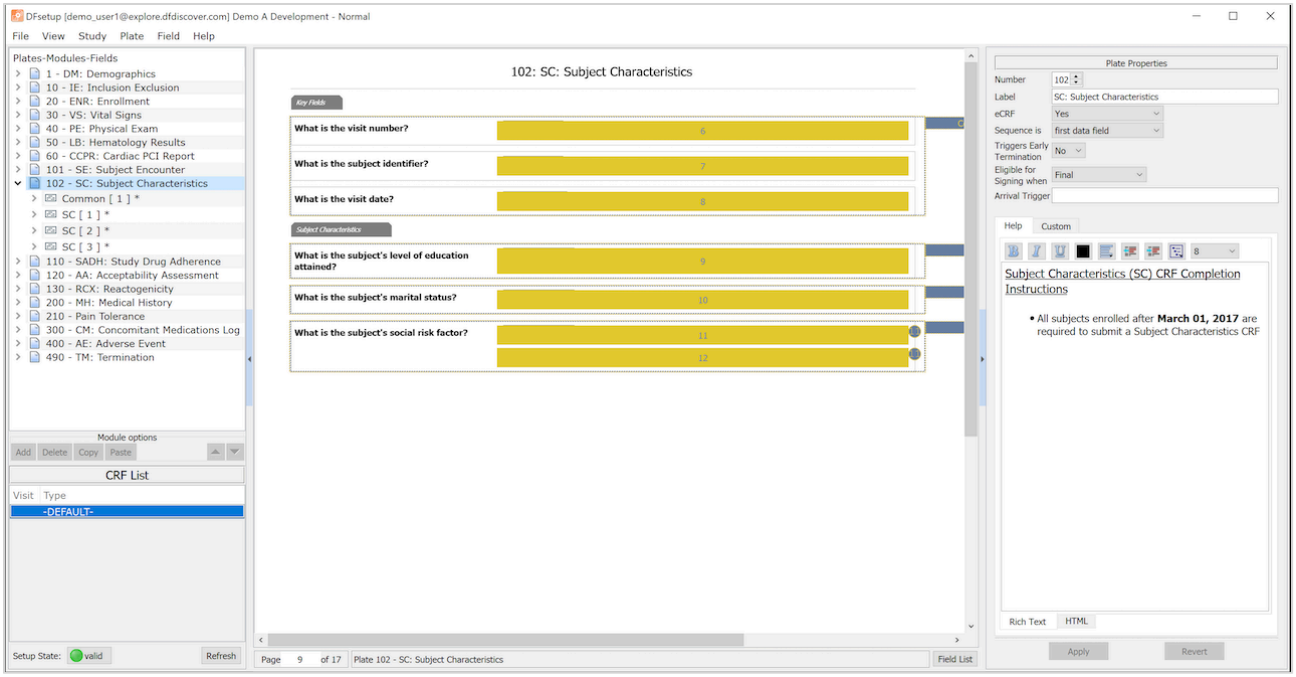
Electronic case report forms (eCRFs) are created within **DFsetup** without needing to import a PDF file containing the form background image. eCRFs do not contain barcodes and are therefore only suitable for entry via EDC.

Select **Study** > **Add eCRF Plate**. Assign the new plate a unique plate number.

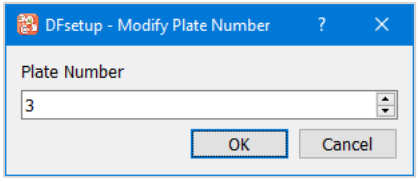


To add fields to an eCRF, begin by choosing the needed module(s) and adding them to the plate. Select **Plate** > **Add Module...**. From the **Add Module** dialog, select the modules to be added to the current plate. Click OK to add the selected modules.

Drag fields or modules to the eCRF window. Fields are numbered according to the order they are added to the eCRF. When adding entire modules to an eCRF fields are numbered according to the order that they were created in the module. Fields can be reordered by selecting **Field** > **Order**. On eCRF plates, field layout matches the field order. Reordering fields will cause the field layout to be updated on the eCRF; the field layout does not change with a traditional paper CRF-based plate.

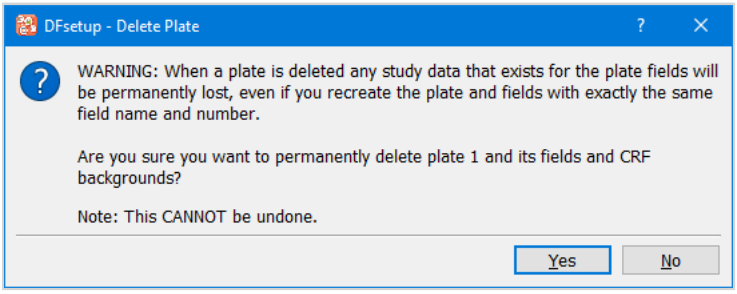


6.6. Modify Plate Number



This can be used to correct plate numbers, but be careful not to create a conflict between the plate number that appears on faxed/scanned CRF pages and the plate number you specify here. When a plate's number is modified, any study data existing for the original plate number is permanently lost. A warning dialog asks you for confirmation before proceeding with the operation.

6.7. Delete Plate



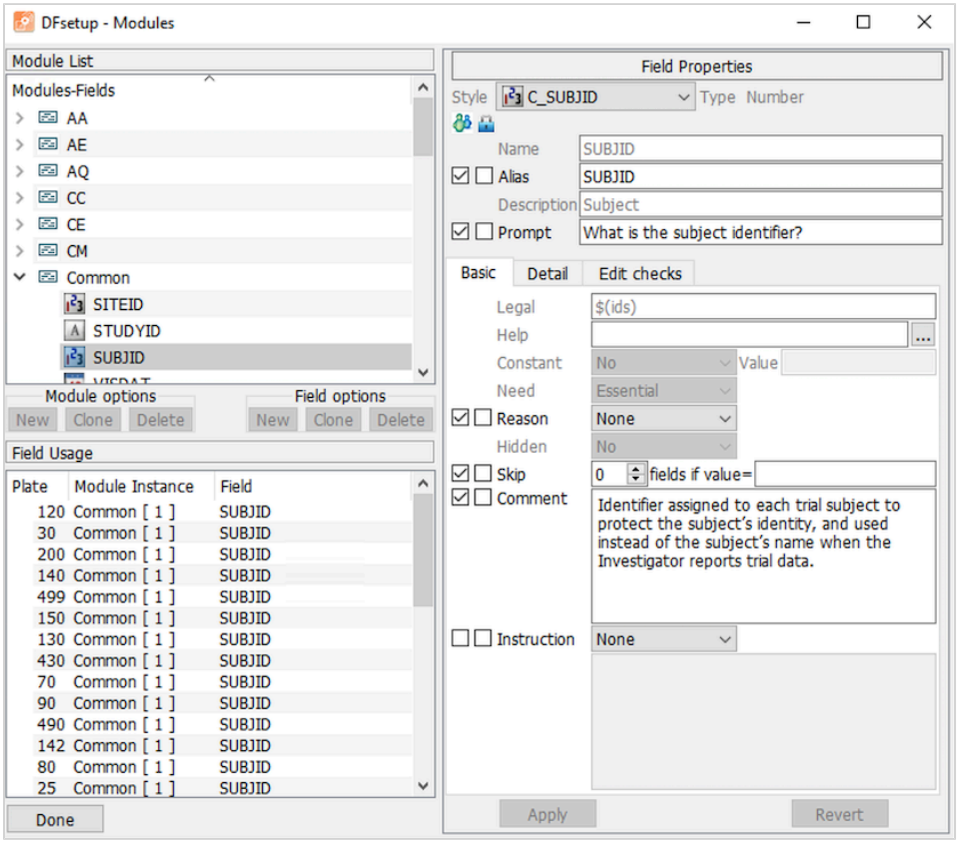
Select 'Yes' to permanently remove the current plate from the study setup, including all of its field definitions and background CRF images. Style definitions are not deleted even if they are no longer in use by any field. When a plate is deleted, any study data, queries and reasons existing for the plate are permanently lost, so exercise caution when performing this operation.

Chapter 7. View Menu

This chapter describes the functions located under the **View** menu.

7.1. Modules

Field definitions are added to modules using this dialog. A list of modules and the fields contained in them appear on the left side of the dialog. The individual module and field properties appear on the right.



Module List. The module list contains the module and field definitions for your study database. Use this part of the modules dialog to manage your modules and the fields they contain.

Module Properties. Module properties include the module name and a short description of what the module is for.

Module Options. Module options allow you to create a new module and clone or delete an existing module.

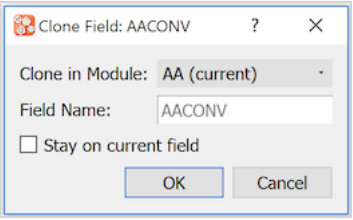
- New: Create a new module without any fields. Fields are then added and managed using the Field options buttons.
- Clone: Create a copy of an existing module including all of its fields. The module name provided for the cloned module must be unique. Field names within the cloned module remain the same as those in the original module provided that the Global Field Setting **Field Name must be unique across all module definitions** is disabled (unchecked). If this setting is checked, during the clone, field names will be modified to maintain uniqueness by appending an underscore and number to the original field. Field names and other field properties may be edited in the Field Properties window.
- Delete: Delete an existing module and all of its fields. If the module and its fields are defined on one or more plates, the module cannot be deleted. Module usage can be determined by selecting the module from the list and examining the Module Usage window.

Field Properties. The field properties area of the modules dialog is where you define each field in your database.

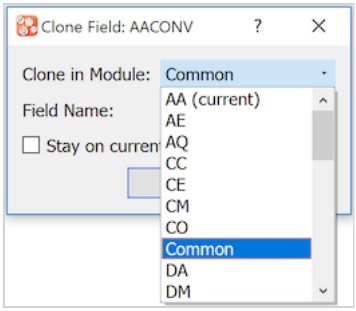
Field Options. Field options allow you to create a new field and clone or delete an existing field.

- New: Create a new field within the selected module. Provide a style and a data type. Enter all properties that are common to all planned instances of the field. Leave instance-specific properties unlocked or unspecified.

- Clone: Create a clone of a field. Complete the **Clone Field** dialog.



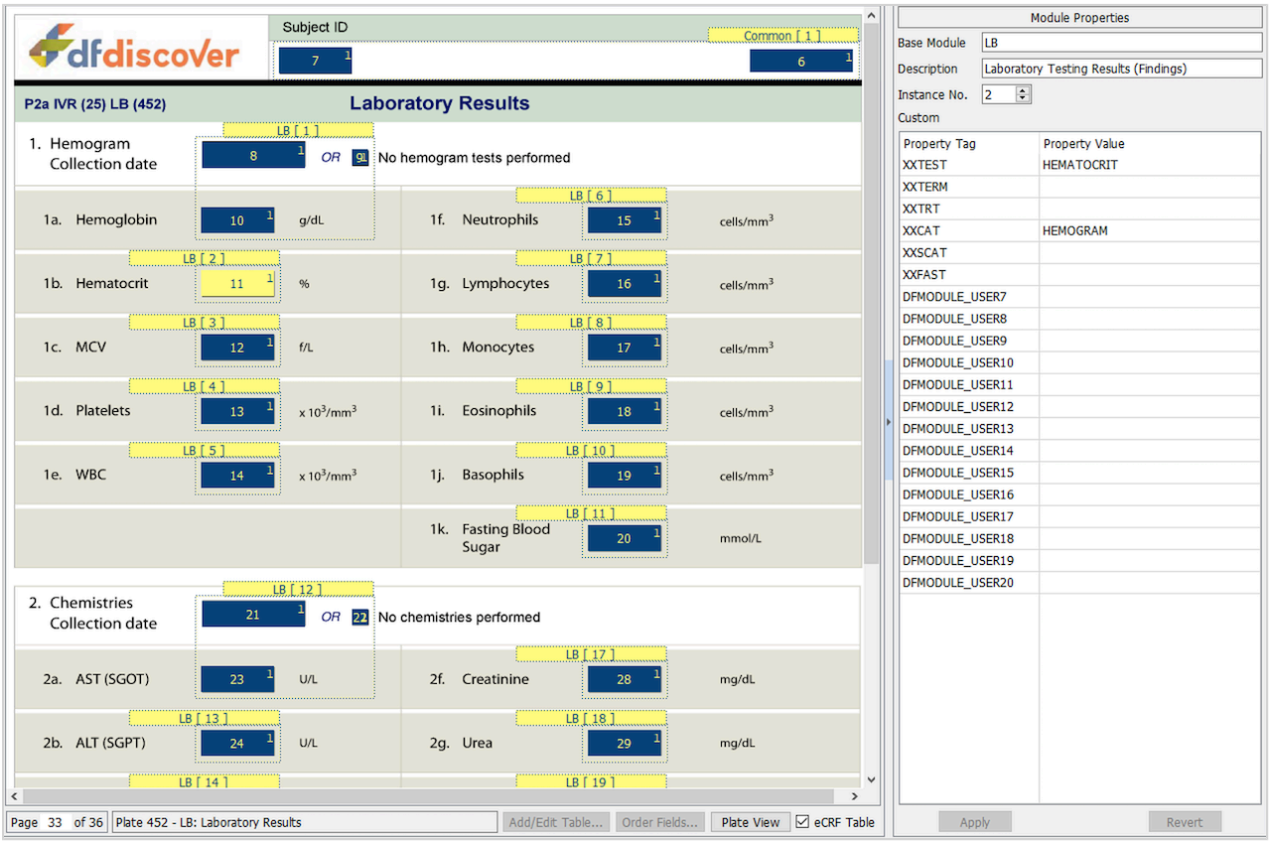
Fields can be cloned within the current module (default) and can also be cloned to a different module. For the latter, choose the destination module from the **Clone in Module** drop-down list.



The cloned field requires a name that is unique within the destination module. Specify a **Field Name**. After the field is cloned, the default behavior is to switch the focus to the new field in the destination module. Mark **Stay on current field** to override this behavior and leave the focus on the current field in the current module.

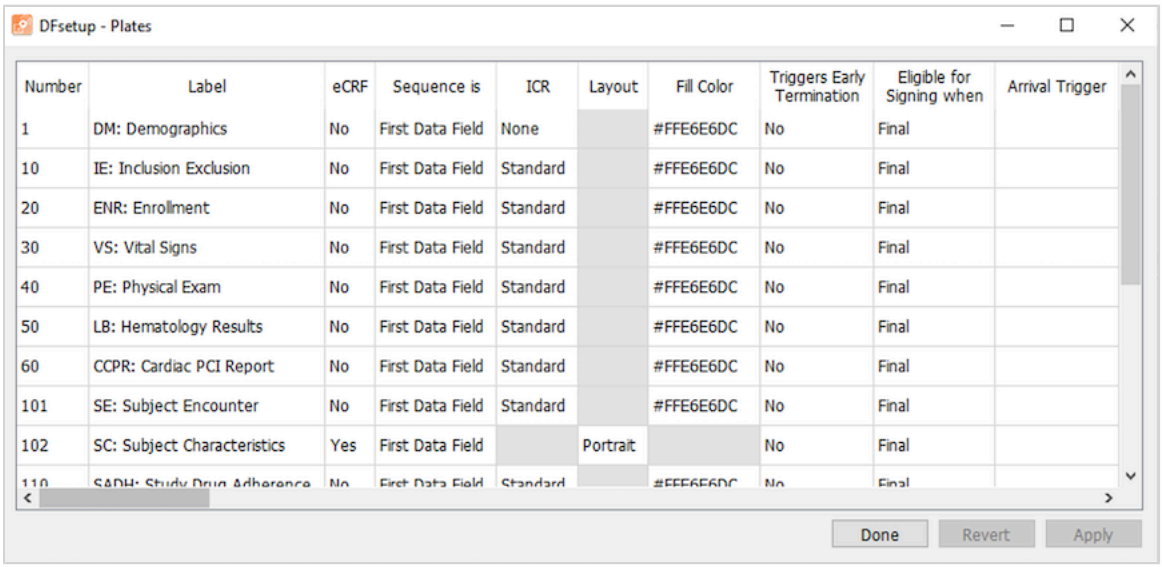
- Delete: Delete an existing field. If the selected field is in use on one or more plates, it cannot be deleted. Field usage can be determined by selecting a field from the list and examining the Field Usage window. As fields are mapped to plates, all fields and the module they belong to are listed in this part of the dialog.

Custom Properties. Once a module instance is applied to a plate, custom properties can be specified for that module. Up to 20 custom properties can be specified for each module instance. Property tags are assigned using Custom property tags. Each property value may contain up to 500 characters.



7.2. Plates

After adding eCRF plates or importing blank CRF images (Import CRFs), use the **Plates** dialog to enter a descriptive label and other properties for each unique plate in the study database.



Recent edits are shown in bold and, when the focus is on the cell, also have a red arrow that can be used to undo the change. Click Apply to save the changes and Done to dismiss the **Plates** dialog. A confirmation dialog appears if you attempt to quit without saving changes.

Label. A descriptive label not exceeding 100 characters is required for each plate. This label is used in Query Reports to identify missing pages, and as the data statement label in SAS® job files created by **DFsas**.

eCRF. eCRF plates do not contain barcodes and are therefore only suitable for entry via EDC.

Sequence. By clicking this cell 2 options are available to indicate whether the key field to identify the visit field number if predefined (or in the barcode), or is in the first user-defined data field of the plate.

If a plate is never repeated, and thus will appear only once in the study database with a single visit number, the visit number can be predefined or included in the CRF barcode. If the plate may be repeated, different visit numbers will be required to distinguish the different records. In this case it is typical to design the CRF page with the visit number as the first data field (as either a choice or numeric field) that is selected by whoever is completing the form (for EDC studies) or completed by whoever submits the form (for paper-based studies).

ICR. This is only relevant to CRF plates if pages will be faxed or emailed to the DFdiscover server. DFdiscover ICR can be used to read the data fields on a faxed CRF page and create an initial data record for review by data management staff. There are 3 options:

- none - do not perform ICR on any of the fields on the current plate
- standard - perform ICR on new faxed pages, but if the page is a refax discard all ICR'ed values when the existing data record is brought forward for comparison with the new faxed page.
- merge - perform ICR on new faxed pages, and retain the ICR'ed values for any blank fields in the existing data record when it is brought forward for comparison with the new faxed page, except for those blank fields which are supported by a reason in the study database.

The DFdiscover ICR algorithms can read check boxes (in check and choice fields), numbers (in numeric and date fields), 3-character months (in dates that use this format), plus and minus signs (in signed numeric fields) and visual analog scales.

The ICR algorithm uses the legal range specifications for each field to determine whether it read a legal value. If the value is illegal the field is left blank. Choice fields for which more than one box was checked are also left blank as this is always an illegal response.

Fill Color. Fill Color defined at the plate level is inherited by fields as default color. This color is used to paint/fill the area between the data entry widget and the CRF background color. When using color and higher resolution data collection CRF images, the appearance is greatly improved by also *filling* areas around data entry widgets in a color sensitive manner. Ensuring that *Fill Color* is set to match the surrounding area of data entry widget on CRF background will improve overall visual experience. If Fill color is not defined (i.e. blank) at the plate level, the color specified in **Study** > **Global Settings** is used.

Layout. The Layout property applies to CRFs without backgrounds in **DFexplore**. This property has 2 options:

- Portrait - Default appearance for eCRFs. When printed or saved to PDF, eCRFs are displayed as 8.5 x 11-inch pages.
- Landscape - eCRF has a greater width to allow for instructions on the right of fields or for wider tables. When printed or saved to PDF, eCRFs are displayed as 11 x 8.5-inch pages.

The default layout for eCRFs is Portrait.

Triggers Early Termination. There are various ways to signal the termination of subject follow-up for each cycle defined in the study visit map. One way is to design a special CRF form that is used by the clinical sites to record the date and reason that follow-up is terminating early for a specified cycle for a specified subject. The visit number on the early termination plate must be included within a cycle in the study visit map. This identifies which cycle is terminating early. See Visit Map for additional information on cycles and cycle termination.

Eligible for Signing when. This is relevant for plates where e-signatures fields are defined. If e-signature fields are defined for this plate, the subject record will be signed with user's signature upon save based on this selection. There are 2 options:

- Final - When record status is changed to Final.
- Final or Incomplete - When record status is changed to either Final or Incomplete.

Plate Arrival Trigger. Plate arrival triggers are used to run specified program on the DFdiscover server on arrival of a new record for a specified plate. For example a trigger might be used to send someone an email message on arrival of a serious adverse event form. Plate arrival triggers execute on the following events:

- when a barcoded CRF page arrives provided the barcode can be read,
- when a page is identified and sent to the study using the Image Router function,
- when data records are imported to the new record queue using **DFimport.rpc** with the `-n` option, unless explicitly suppressed by also including the `-s` option.

> **Important**
>
> If an image cannot be identified on arrival and thus ends up in **DFrouter**, the plate arrival trigger will be delayed until the page has been identified using **DFexplore**.

Plate arrival triggers are UNIX shell programs that may take arguments. Plate arrival triggers have the following requirements:

- They must be "installed" in either the study `ecbin` or DFdiscover `ecbin` directory - this "registers" them to DFdiscover. If the program is registered in both locations the study level takes precedence.
- The program name cannot contain any of the following characters:

```
/ space | ` $ ; & *
```

- The program arguments cannot contain any of the following characters:

```
| ` $ ; & *
```

Two arguments are automatically added to the end of the command line, in the following order:

1. full path to the image file
2. the data record created by ICR or imported

Plate arrival triggers are executed immediately following the ICR step for both incoming documents and pages identified using **DFexplore**, and the ICR results are passed to the program. However, when a plate is imported to the new record queue ICR does not occur, and the imported data record is passed to the program instead.

As an example, suppose we want to send a notification to the study adverse event monitors (Jack and Jill) whenever a new adverse event form arrives (plate 200) asking them to review it within 24 hours. In addition we want to notify the study data manager (John) so that he can check that the review occurred on time. The plate arrival trigger for plate 200 might look like this:

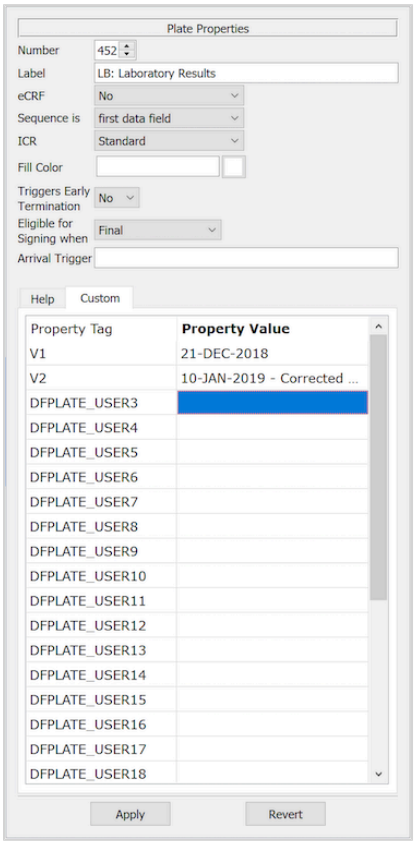
```
AEnotification.shx "jack@company.com jill@company.com" "john@company.com"
```

The script, stored in the study `ecbin` directory that does the work, might look something like this:

```
#!/bin/sh
# AEnotification.shx "jack@company.com jill@company.com" "john@company.com"
# send email to SAE monitors when an adverse event arrives

TO=$1
CC=$2
CRF=$3
DATA=$4
SUBJECT="New adverse event for study XYZ"

/usr/bin/mailx -s "$SUBJECT" -r "$CC" -c "$CC" "$TO" << END
A new adverse event report (plate 200) for study XYZ
arrived today: `date`
The ICR results are:
$DATA

Please review this form within 24 hours.
Thank you.
END
```

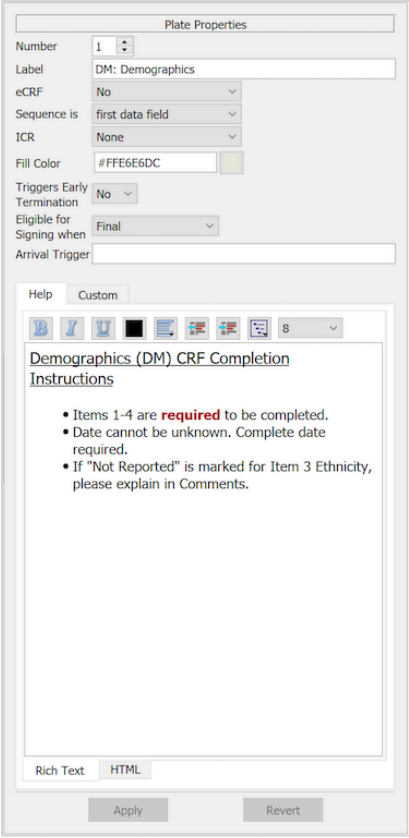
Custom Properties. Up to 20 custom properties can be specified for each plate. Property tags are assigned using Custom property tags. Each property value may contain up to 500 characters.

Figure 7.1. Plate Properties - Custom properties



You can also modify these plate properties using the **Plate Properties** dialog.

Figure 7.2. Plate Properties dialog



This dialog may also be used to specify an optional plate level help message. Each Help message can be formatted using the Rich Text editor, or HTML code can be entered or pasted using the HTML editor. [1] In **DFexplore**, this help message is available to users when they select **Help** > **Plate Help**.

7.3. Styles

Default Styles. When defining a data field you start by selecting a style that has the desired data properties. DFdiscover includes the following predefined styles:

- SimpleNumber - for numeric fields
- SimpleDate - for dates
- SimpleTime - for times
- SimpleDateYYYY - for dates with 4-digit years
- SimpleString - for text fields
- SimpleChoice - for fields with 2 or more mutually exclusive check boxes
- SimpleCheck - for fields with a single check box

[1] DFcollect is not able to consistently render Rich Text Format (RTF) with all Google API versions. If help is required in each of **DFexplore**, DFcollect and DFweb, format the message as HTML.

- SimpleVAS - for visual analog scales

These styles should be reviewed and modified as needed. For example, the date format defined in the SimpleDate style should be changed to match the date format used in the study. The DFdiscover default styles cannot be deleted.

Visit Date Usage. Use must be set to Visit Date for each date that is relevant to subject visit scheduling. This includes the date of each baseline, scheduled follow-up, and termination or abort visit (including date of death if recorded). The Visit Date use may also be assigned to optional visit dates. The visit date fields on the study CRFs identify the date on which the visit occurred. Thus, there should be only one field with the visit date property set for each visit. If the visit date use is assigned on more than one plate within a visit, it is expected that all of these date fields will have the same value.

eSignature Usage. An eSignature module is included by default for all studies. The module has five fields - `SigName`, `SigDate`, `SigTime`, `SigReasonChoice` and `SigReasonString`. Respectively each have their Use property set to `eSignature Name`, `eSignature Date`, `eSignature Time`, `eSignature Reason` and `eSignature Reason`. See eSignature Module and 21 CFR Part 11 Compliance for additional information.

Study Specific Styles. Study specific styles should also be defined for data field types that appear repeatedly in the study definition, e.g. Subject ID, Yes/No questions, and any repeating block of questions, e.g. medical history items, medication usage, etc. You can define as many study specific styles as needed. Each style is identified by a unique style name comprised of letters, digits, spaces and the '_' character. Each style name must begin with a letter or digit, and cannot exceed 15 characters in length.

Styles can include both fixed properties which are locked in the style and cannot be changed when defining a data field that uses the style, and default properties which are not locked in the style and can be changed when defining a data field. Styles thus serve both to simplify the definition of individual data fields, and more importantly to enforce consistency among data fields that share some of the same properties: coding, legal ranges, edit checks, etc.

Ideally you should define the styles you will need before starting to define individual data fields, but styles can be created as the need arises. Be sure to consider the required properties of all of the data fields that will use each new style you create so you can decide which properties to lock in the style.

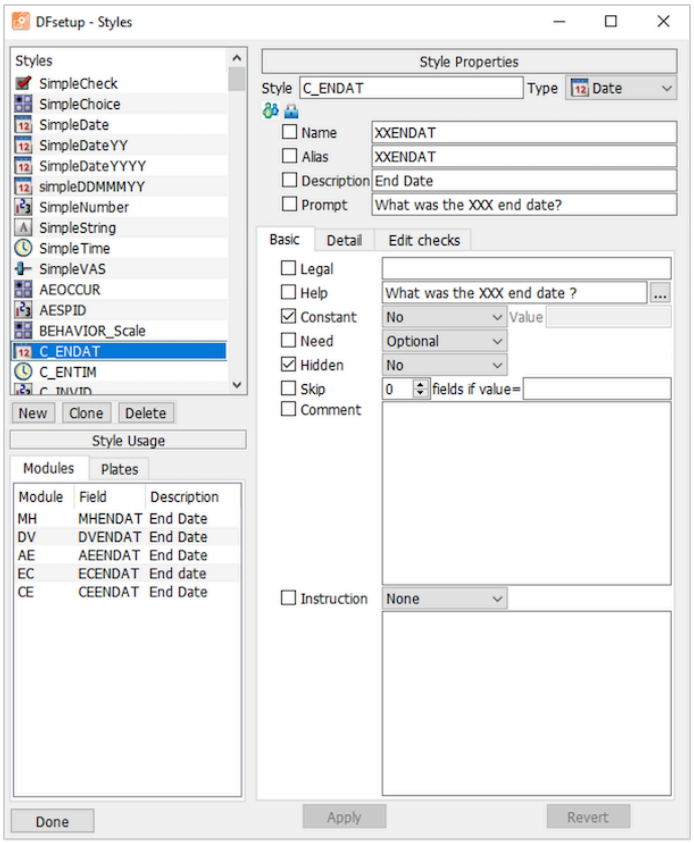
Modifying Styles. A style definition can be changed at any time. Properties that were previously defined at the field level can be over-ridden by specifying and locking a value in the style. When this is done all fields that use the style will inherit the newly locked style property.

Styles can also be changed by unlocking a previously locked property. When this is done the data fields that use the style will continue to hold the property's previous value. No changes will occur unless the user decides to edit the now unlocked property for some or all of these data fields.

> ## Warning
>
> Styles can be changed at any time. Once study data has been entered into the database, be careful not to change anything that would conflict with data that has already been collected. Examples of properties that should not be changed include:
>
> - the label on codes that already appear in the database if this would change the meaning of existing data values
> - the field storage length if values already collected would not fit in the new storage length
> - the format of numeric and date fields if this would make existing data values invalid
> - names that have been used in edit checks, unless you are willing to also change the edit checks

Styles Dialog. This dialog is used to define and manage all study styles:

Figure 7.3. View > Styles



The components of this dialog include:

- Styles: list of currently defined styles
- Style Properties: data field properties for the selected style, C_ENDAT in this example
- Checked Properties: are locked in the style and cannot be changed when defining a data field
- Style Usage: data fields that use the selected style
- New: define a new style
- Clone: define a new style starting with a copy of the selected style
- Delete: delete the selected style
- Apply: save changes to the selected style, requires **File** > **Save** to become permanent
- Revert: undoes changes made to the style, but only those which have not yet been applied
- Done: dismiss the dialog

7.3.1. Common Properties

The data field properties displayed in the right panel depend on the selected field type (*check, choice, date, time, number, string, visual analog scale*). We will start by describing the common properties and then describe those that are specific to each data type.

Style. Each style is identified by a name consisting of letters, digits, underscores and spaces, at most 15 characters in length and beginning with a letter. A dialog requests the new style name whenever you select New or Clone. Existing style names can be modified in the Style Properties panel, provided the style name remains unique.

Type. This property identifies the underlying data type from the list of available data types:

- choice - a field consisting of up to 98 choices, only one of which may be selected. Each choice has a code and a label, and a special code and label for cases in which no box was selected. The codes are stored in the study database and labels are stored in the study schema.
- check - a field consisting of a single box which may be checked or unchecked. A code and label are defined for each of these 2 states. The code is stored in the study database and the label in the study schema.
- number - a field consisting of digits, with or without embedded delimiters (e.g. decimals, commas, etc.) and with or without a leading plus or minus sign.
- date - a field consisting of a day, month and year part in any order and with or without delimiters, as specified in a format specification.
- time - a field containing hours and minutes, or hours, minutes and seconds.
- string - a text field with specified screen display and maximum storage lengths.
- VAS - a visual analog scale for measuring subjective characteristics where a continuous scale is more appropriate than a discrete scale. It is typically represented by a horizontal line, representing a dimension (e.g. pain) with labels at either end (e.g. none, excruciating) which research subjects use to mark their position along the dimension.

For more details on each supported data type see Data Types. The properties panel changes on selecting the field type to display the properties appropriate to each type, as described in Type Specific Properties.

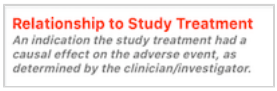
Name. Each data field is required to have a name. The name must be unique within all module definitions for the study, but it does not have to be unique across the entire study. For instance, it would be common for a field name to repeat across instances of a module definition. The name must be at most 80 characters in length, begin with a letter and consist of only letters, digits, and underscores. The name is case sensitive, hence `MED1`, `Med1` and `med1` each define a unique field name. This name is used in the edit check language and by **DFsas** when creating SAS® data sets.

Alias. The alias of a data field is a string that uniquely identifies the field across the entire study. The alias property shares the same limits as the name property with respect to length and character set. Each data field must have an alias that is different from all other aliases used to define data fields in the study. An alias may also contain meta-words (a name may not).

Most aliases are defined at the module or field level, not in the style, with one exception. An alias can be defined in the style for fields which appears only once on any given plate, like subject ID and initials, by using the meta-word for plate number like this: `PID_$(plate)`. This evaluates to `PID_001` on plate 1, `PID_002` on plate 2, etc. and will thus be a unique alias.

Description. A description, up to 40 characters long, must be specified for each data field. It is used to identify data fields in Query Reports and as the field label in SAS® data sets created by **DFsas**, and thus must uniquely identify each data field. Descriptions can be fixed in the style if they include meta-words, like `$(plate)` which make them unique, or a default description can be included and then modified at the data field level, to reduce typing and provide an initial template. Most often the description is unlocked and blank in the style.

Prompt. A prompt, up to 256 characters long, may optionally be specified for each data field. It is used to display the question text or item label for the data field on eCRFs (plates without a CRF background image) and in **DFcollect**, if it differs from the field description. The prompt is required for the first field in a set of grouped fields. The prompt is formatted as bold text by default, but simple HTML may be used for additional formatting, such as using italics, changing the font color, or adding line breaks, spacing, or symbols. For example, a prompt defined as `<font color="red">Relationship to Study Treatment</font><br><font color="grey"><small><i>An indication the study treatment had a causal effect on the adverse event, as determined by the clinician/investigator.</i></small></font>` will appear on the eCRF as:



Hover over the Prompt field in the Field Properties panel to preview its appearance on the eCRF.

Legal. An optional comma delimited list of legal values and ranges can be specified, using a dash (-) or tilde (~) for inclusive range specifications, as illustrated in the examples below:

- 0,3-8,20,22,30-39 for a numeric field
- -5.0~-2.5,2.5~5.0 for a numeric field with negative values
- 01/01/06-31/06/07, 01/09/07-today for a date field
- ABC,XYZ,A B,"A,B" for a string field
- `$(choices)` for check and choice fields

If a legal value includes commas it must be enclosed in double quotes. For example quotes are needed here: "JAN 01,2008"~today for a date with format 'mmm dd,yyyy', but not here: JAN 01 2008~today for a date with format 'mmm dd yyyy'.

Quotes are not required when the legal value contains spaces as in the following example: Dr. A,Dr. B,Dr. C.

The meta-word `$(ids)` can be used as the legal specification for the subject ID field. This indicates that only subject ID numbers specified in the sites database are legal, and is the recommended way to specify legal subject IDs. The actual subject ID numbers are determined by examining the range of subject IDs for each site from the site definition.

DFdiscover does not prevent the entry of illegal values but highlights the field with the illegal color (red) if an illegal value is entered. If the Legal value property is not specified the field will only turn red if Need is specified as required or essential and the field is left blank. If metadata (a query or reason) is attached to a data field the metadata color (blue, orange or green) takes precedence over the illegal value color.

If there are any illegal fields on a data record **DFexplore** prevents the user from saving the record with status 'Final', unless the user has provided a reason explaining each illegal value, or replied to a query on the illegal values. This rule applies at all workflow levels, including level 7.

Data queries classified as requiring 'correction' (not 'clarification'), and which have query categories 'illegal' or 'missing' are automatically resolved when the data field is corrected according to the following rules:

- Queries classified with category 'illegal' are automatically resolved if one of the values specified in the Legal property is entered into the data field. If the Legal property is unspecified these queries will not auto-resolve. This applies to all field types. For example, a blank choice field which has an illegal value query will only auto-resolve on selecting one of the choice options if the Legal property contains the `$(choices)` specification.
- If the Legal property is specified, a query classified with category 'missing' will be automatically resolved when one of the specified legal values is entered into the data field. If the Legal property is not specified, any entry will resolve a 'missing' value query.

Help. An optional help message can be displayed at the bottom of the **DFexplore** window or in the metadata editor panel when the focus rests on the data field. In **DFexplore** users can also see this help message from **Help** > **Field Help**. Help messages can include meta-words (see Meta-Words). For example, the legal value list can be included in the help message as in 'Legal values are: `$(legal)`'. Help message can be either formatted using Rich Text editor or HTML code can be entered or pasted using the HTML editor. The tool button in front of Help property can be used to launch the help editor window.

Constant. If a field is preset to a constant value, the Constant property is set to yes and the constant value is given. The value must adhere to the rules for the field - field type, format, store length.

Need. One of 3 values must be specified for this property:

- **Optional.** for fields that may be completed with a data value or missing value code. Optional fields are legal when left blank.
- **Required.** for fields that may be completed with a data value or missing value code, but are considered illegal when blank.
- **Essential.** for fields that must be completed with a data value. Essential fields do not accept missing value codes. They are considered illegal if blank or if they contain a missing value code (which can arise if a previously optional or required field is changed to essential).

Key fields (subject and visit numbers) are always essential, regardless of how the need property is specified.

None of the need options prevent users from entering illegal values, but entering an illegal value does prevent the user from saving the data record with status Final. As for illegal data values, fields that are illegal because they are required or essential but blank also prevent the data record from being saved with status Final.

Reason. The reason level (1-7) identifies the minimum workflow level at which changing the data field requires a reason. For example, if the reason level is set to 3, a user can change the value without specifying a reason if the data record is currently below level 3, but a reason must be provided if the record is at level 3 or higher. If the reason level is set to anything between 1-7, an additional option 'Only if changing a non-blank value' becomes available. If this option is checked, a reason for data change will only be required if the record exists at or above the specified level and the field's value was not blank prior to being changed. By default, this option is unchecked which means that the field's original value will be ignored - a reason for data change will be required for any change to the field if the record exists at or above the specified level.

The reason property set in the style or the data field definition is only used if the global setting (see Global Settings) is set to 'per field'. A global setting of 'never' is equivalent to specifying 'none' for all data fields, and a global setting of 'always' is equivalent to specifying level 1 for all data fields.

In **DFexplore** the reason dialog automatically appears when a reason is required. The dialog displays the reason, if any, for the current value, a text widget in which to enter a reason for the new value, and buttons to either record the new reason or revert to the previous value.

This property is optional. If 'none' is selected a reason is not required for data changes, with one exception. If a field already has a reason associated with its current value, then changing the value requires the entry of a new reason, regardless of the setting for this property at field, style or global levels.

Hidden. Data fields can be hidden (data entry widget not visible) or masked (data entry widget is visible but unavailable, and no data can be entered). By default, data fields are not hidden. If a data field is classified as hidden or masked users require permission to see it in **DFexplore**. Permission to see hidden or masked fields is set in the study roles defined in **DFadmin** and may differ by plate and visit, but not by level. A field cannot be hidden or masked at some workflow levels and visible at others.

Hidden or masked fields can be located anywhere on the plate, and are used for fields that only certain users need to use, for example:

- dictionary coding fields not present on the study CRFs
- fields for a reviewer's initials and date signifying completion of some task
- a date stamp completed by a batch edit check indicating the investigator payment run in which the assessment was included.

This field property only affects whether users can access the hidden or masked fields in the **DFexplore** Data and List views. It has no effect in **DFbatch**. Edit checks on hidden or masked fields a user cannot see will not be triggered in **DFexplore**. This applies to all edit checks on hidden or masked fields: plate entry and exit edit checks, as well as field entry and exit

edit checks. While hidden or masked fields are skipped and their edit checks ignored, they can be read and set by other edit checks both interactively and in batch.

This field property has no effect on DFexport.rpc or **DFsas**. Thus users with permission to create and run **DFsas** jobs in **DFexplore** will be able to create and export SAS® data sets containing hidden or masked fields.

Display, Store and Widget Lengths. The Display length, which must be specified for string, numeric and date fields, determines the number of characters that will be displayed on one line of the data field in **DFexplore**, while the Store length determines the number of characters that can be saved in the database, and also whether the data field needs to expand vertically to allow data reaching the store length to be entered.

The length of the data field widget drawn on the screen determines the region of the CRF that will be covered by the data field in **DFexplore**. The widget length may exceed the display length, and often does when clicking boxed data fields. This allows the boxes to be completely covered in the **DFexplore** data entry window. Thus, data field widgets will not automatically shrink if you draw them longer than required by the display length.

Clicking Adjust to Fit changes the length of the data field widget to match the specified display length. The adjustment is made to the right edge of the widget, with the restriction that widget length cannot exceed the right edge of the data entry screen. If matching the specified display length would exceed this boundary, the widget increases to the right as far as possible and the display length is then decreased to match the new widget length.

Remember that any text and graphics that lie beneath the data field widgets will be hidden on the **DFexplore** data entry screen. This is what you want when the field is designed using data entry boxes, as in CRFs designed for faxing and ICR, so Adjust to Fit should not be used on such data fields.

Fill Color. Fill color is used to paint/fill the difference area between the data entry widget and CRF background color. Particularly with color and higher resolution data collection CRF images, the appearance is greatly improved by also *filling* areas around data entry widgets in a color sensitive manner. Ensuring that *Fill Color* is set to match the surrounding area of data entry widget on CRF background will improve overall visual experience. Fill color if not defined (i.e. blank) at the field level, the color specified in Plate properties is used.

Fill color has 4 components: alpha (opacity), red, green and blue and is expresed as a hex number with the format #AARRGGBB. Click the square color swatch to select the color (and opacity) from one of several palettes or use the color dropper to sample from any part of the CRF. Use opacity to "mix-in" fill color with underlying color if needed.

Units. This is a string value recording what units are used for the current field values.

Skip. An optional rule may be specified to skip one or more fields following the current field when the current field has one or more specified values. The rule has 2 parts as shown in the style dialog above: the number of fields to be skipped and the data value or values that trigger the skip.

To indicate that more than one value can trigger the skip enter a comma-delimited list of values and ranges, e.g. 1-3,5,7.

Two meta-words can be used in skip specifications. Specify `$(blank)` to trigger the skip when the current field is blank, `!$(blank)` when it is not blank, `$(legal)` when it contains one of the values specified in the legal values property, and `!$(legal)` when it does not contain one of the legal values.

The `$(legal)` meta-word cannot be combined with other values and only refers to the values specified in the legal values property. If this property is blank the skip rule will never be met.

The `$(blank)` meta-word can be combined with other values, except `$(legal)`. For example, if the current field is a choice field and you want to skip fields if none of the choice boxes was selected (i.e. the field is blank) or if one of the boxes corresponding to choice codes 1-3 was selected, the skip specification would be: `$(blank)`,1-3.

The Skip rule is evaluated on a forward exit from the field using the tab or return key in **DFexplore**. It is not evaluated on a backward exit using Shift-Tab or Shift-Return, or on a direct move to another field using the mouse.

If the rule is met the focus jumps directly to the data field following the number of fields being skipped. The fields are skipped regardless of whether they are blank or contain data values, and are not changed in any way by being skipped. Any field entry or field exit edit checks that might exist on the fields being skipped will also be skipped since these fields never get the focus.

More complicated rules and/or different skip behavior can be implemented using edit checks.

Edit checks. Edit checks are user defined programs used to implement data consistency checks, lookup tables, computed values, and many other types of programmed behavior (see *Programmer Guide, Edit checks*).

Edit checks can be triggered in **DFexplore** when the plate first appears on screen (plate enter), when the user moves into a data field (field enter) or leaves a data field (field exit), or when the user saves the data record (plate exit). All edit checks are attached to data fields by entering the edit check name and parameters (if any) in one or more of these 4 edit check widgets. It is perfectly legal for an edit check to be triggered more than once, for example on field exit and again on plate exit. It is also legal to trigger more than one edit check on the same event by entering a list of edit checks separated by commas and/or spaces. If more than one edit check is entered in any of the 4 edit check widgets they will be executed in order from left to right, on the designated plate or field enter or exit event.

Procedure 7.1. Edit check Execution

Edit checks are processed in the following order:

1. *On Plate Entry:* all fields are examined in field number order. Any plate entry edit checks are executed before moving to the next field.
2. *On Field Entry:* by keyboard or mouse, forward or backward, any field entry edit checks are executed.
3. *On Field Exit:* by keyboard or mouse, to move to any other field or to the Save buttons, any field exit checks are executed.
4. *On Plate Exit:* all fields are again examined in field number order. Any plate exit edit checks are executed before moving to the next field.

> **Important**
>
> - Field enter and field exit edit checks only fire if the user visits the field, and thus will not be triggered on fields the user skips. This applies to manual skipping via the mouse and scroll bar, and to automated skipping via the Skip field property and edit check `dfmoveto` instructions.
> - Edit checks attached to hidden fields are not executed in **DFexplore** if the user does not have permission to see the hidden fields. This applies to all 4 types: plate and field, entry and exit.

Edit checks can also be executed in batch mode, see *Programmer Guide, Batch Edit checks*. Batch mode simulates a user retrieving each specified data record, tabbing through all data fields, and then saving the record. Thus the order of edit check execution proceeds just as it does interactively, but with the following differences:

- edit checks on hidden fields always run in batch mode regardless of who is running the batch,
- `dfaccess` statements are ignored during batch processing,
- `dfmoveto` statements are ignored during batch processing thus all fields are traversed.

Edit check programs can be entered, checked and published using the **View** > **Edit Checks** dialog described in Edit checks.

Comments. Internal documentation for the field can be stored in the comments property. Comments are only visible when editing a field definition; they are not displayed, nor accessible, anywhere else. A comment has a maximum length of 1500 characters.

Instruction. Additional text related to the field may be defined in the Instruction property for display on CRFs without a background.

The instructions are displayed on eCRFs in one of five positions:

1. *Above the prompt:* In a separate row above the field, aligned with the field prompt.
2. *Above the field:* Aligned with the leftmost boundary of the field widget.
3. *Below the field:* Aligned with the leftmost boundary of the field widget.
4. *Below the prompt:* In a separate row below the field, aligned with the field prompt.
5. *Right of field:* After the the field widget in the same row.

Instructions are not displayed on CRFs with backgrounds or for fields that are included in a table. In **DFweb** on larger screens (desktop), the QRM buttons are shifted to make space for instructions right of field. In **DFweb** on smaller screens (tablet and mobile), instructions right of field are displayed below the field. Given screen size limitations in **DFcollect**, instructions only appear above or below the prompt text, in the area above the data field and QRM buttons. Instructions defined above the field will appear above the prompt text, while instructions defined below or right of field will appear below the prompt text.

Instructions are formatted as plain text by default, but simple HTML may be used for formatting, such as using bold and italics, changing the font color, or adding line breaks, spacing, or symbols. The same HTML formatting allowed in the Prompt property is also allowed in the Instruction property.

Custom Properties. Custom properties may be specified once the field is applied to a plate. Up to 20 custom properties can be specified for each field. Property tags are assigned using Custom property tags. Each property value may contain up to 500 characters.

7.3.2. Type Specific Properties

7.3.2.1. Specific Properties for Check & Choice Fields

Coding. Check fields consist of a single box, that is either checked or not checked, while choice fields have 2 or more boxes, at most one of which may be checked. In both data types, either the field has no response (no box is checked) or a single box is checked. A numeric code and descriptive text label are required for each box and defines the meaning of that box being checked. Additionally, a numeric code and text label are required for the "no response" case, where no box is checked. Within a field definition, each numeric code must be a unique integer value, in the range `0-65535` (inclusive). Text labels have a maximum length of 32 characters and are also generally unique, but are not required to be. Only the numeric code is stored in the database (and hence one needs to be aware of the implications of changing a numeric code once a study has started to receive data), while the text label appears in reports and is for readability purposes only.

	Code	Label
Check Off	0	Unchecked
Check On	1	Checked

- A numeric code and text label must be specified for each of the 2 possible states: the box is checked or not checked.

	Code	Label
No Box	0	blank
Box 1	1	MILD
Box 2	2	MODERATE
Box 3	3	SEVERE

- A numeric code and text label must be specified for each choice box and for the case in which no box is checked.

- Box 1, Box 2, etc. corresponds to the order in which the boxes were checked in the CRF window.

Use. Choice fields can have Use set to **Standard** or **eSignature Reason**. **eSignature Reason** is set and locked for the `SigReasonChoice` field in the eSignature module.

Legal. The meta-word `$(choices)` may be used in the legal specification for choice fields. It evaluates to all specified codes, including the code for no choice. However, if need is specified as 'Required' or 'Essential' the field will still be flagged as illegal if no choice is selected.

It is also possible to specify the codes directly, e.g. 0-4 for the choice field and 0-1 for the check field in the examples above. But remember to include the 'No Box' code as a legal value for 'Optional' fields, otherwise they too will be flagged as illegal when blank.

7.3.2.2. Specific Properties for Choice Fields

Show As. The choice items for a choice field can be presented in one of several ways in **DFexplore**, DFweb and DFcollect.

The overall appearance is controlled by the settings of the **Show As** property, which may be

- **Choices.**

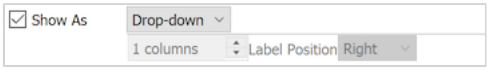
  

  When **Choices** is chosen, the additional **Columns** counter and **Label Position** properties are available.

  The # of **Columns** specifies the number of columns used to present the choices. The default is 3. If there are fewer choices than columns, the additional columns are presented but are blank. If there are more choices than columns, rows are added as needed. For example, if there are 9 choices and 3 columns, additional second and third rows are added, each also with 3 choices. If there are 5 choices and 3 columns, a second row is added and the third column of the second row is unused and blank.

  The **Label Position** specifies where the label appears relative to the choice. Options are None, Right, Top, Left and Bottom, and the default is Right. To create a table of responses, where each row asks a different question but the response choices are always the same, specify Top as the label position for the first question and None for the following questions.

- **Drop-down.**

  When **Drop-down** is chosen, the **Columns** counter and **Label Position** properties are not available as they are not relevant. A drop-down saves space and is able to accommodate more choice responses without requiring a larger screen area. The operating system may truncate labels that are too long to display in the drop-down, and it may also arrange choices into a second column if space is restricted.

Not all of the presentation combinations are implemented in all of the tools.

- An eCRF in **DFexplore** does implement all of the features.
- A paper CRF in **DFexplore** ignores all of the features (because the paper has the choices on it) unless the choice field is defined by one rectangular area, in which the case the choices are presented as a drop-down.
- An eCRF in DFweb honors the distinction between **Choices** and **Drop-down**. However the # of columns is always 1 and the label position is always Right (the actual setup settings are ignored).
- A paper CRF in DFweb ignores all of the features (because the paper has the choices on it).
- DFcollect honors the distinction between **Choices** and **Drop-down**. However the # of columns is always 1 and the label position is always Right (the actual setup settings are ignored).

- For backwards compatibility, when implementing older setups that do not have these properties, if the number of choice responses is 3 or less, the choices are presentation as 1 column of radio buttons; if there are 4 more choice responses, they are presented as a drop-down.

7.3.2.3. Specific Properties for Numeric Fields

Coding. It is possible to associate labels with specific numeric values, as illustrated below, although this is not commonly used.

It is not necessary to specify labels for all possible numeric values, but labels are mandatory for the codes present in the coding table. If display mode is set to labels (e.g. in **DFexplore** List View) and a label does not exist the numeric value is displayed instead.

Code	Label
10	Visit 10
20	Visit 20
30	Visit 30
40	Visit 40

- To associate labels with numeric values enter the numeric value in the Code column and a text label in the Label column.

Display and Store. For numeric fields, the limit for the subject ID number is 15 digits at maximum. The limit for any other numeric field is 10 digits (including the leading signs and the decimal points) at maximum. the display and store length must be the same. You cannot store more digits than are displayed on screen.

If a format is specified the store and display length must equal the format length. **DFsetup** will display a warning and will not allow a field definition to be saved if a mismatch occurs.

Format. An optional format may be specified for numeric fields comprised of:

- an 'n' for each digit in the number
- a decimal or other delimiters
- fixed components, e.g. a leading digit
- a leading 'S' if a sign (+/-) is required
- a leading 's' if a sign (+/-) is optional

Table 7.1. Number Format Examples

Format	Description
none	If no format is specified any positive or negative integer or decimal value that fits in the display length may be entered. Any legal range specification e.g. -5.5~+5.5 may be specified. The display and store length must allow for signs and decimals.
nnn	This format allows any integer between 0 and 999. The legal range specification (if any) may only include positive values, e.g. 80-230. The display and store length must be 3.
nn.n	This format allows any number between 0 and 99.9. The legal range (if any) may only include positive values, e.g. 5.0-69.1. The display and store length must be 4.
snn	This format allows any integer between -99 and 999. If a faxed CRF is used the data field must have 3 boxes with the first box reserved for an optional plus or minus sign. when the sign is provided, ICR will not read the content in the first box. The legal range (if any) may include negative values, e.g. -25~+25. The display and store length must be 3.

Format	Description
Snn	This format allows any integer between -99 and +99. If a faxed CRF is used, the data field must have 3 boxes with the first box reserved for a required plus or minus sign. The plus sign is required for positive values. The legal range (if any) may include negative values, e.g. -25~+25. The display and store length must be 3.
nnnnnnnnnnnnnn	This special format (Subject ID number field in its maximum length) allows integers between 0 and 281474976710655. The legal range specification (if any) may only include non-negative values, e.g. 0-222222222222222. The display and store length must be 15.
nnnnnnnnn	This special format (numeric field other than Subject ID number in its maximum length) allows integers between -2147483647 and 2147483647. The legal range (if any) may only include positive value, e.g. 1111111111-1222222222. The display and store length must be 10.

If an integer or decimal format is specified, e.g. 'nnn' or 'nn.nn', and the user does not enter all of the required digits **DFexplore** will auto-complete the value on field exit by adding leading and/or trailing zeros where needed to match the specified format. For example, 22 is converted to 022 to match format nnn, 2.2 is converted to 02.20 to match format nn.nn. This does not occur if no format is specified, or a format is specified that contains anything other than 'n's or a single decimal.

Important

The '|' character is used as the field delimiter in data records and thus cannot be included in data fields or in formats. **DFsetup** and **DFexplore** both block any attempt to use it.

Single Rectangular Box. Most numeric fields are defined using a separate box for each digit, but it is also possible to define a numeric field consisting of a rectangular box containing pre-printed or hand-written digits. To define such fields click inside the box, select a numeric style, and set the store and display lengths equal to the number of digits the field will contain.

When users complete such fields on a paper CRF all of the specified digits must be included, otherwise the ICR will fail and leave the field blank. Users should be instructed to enter leading zeros, leave a clear gap between each digit, and print large without touching the top and bottom of the box.

For pre-printed numbers ICR accuracy is best for 18 pt Avant Garde-Book with a spread of 15-20% to clearly separate the individual digits. The rectangular box used for these fields should be 1/3 inch high and wide enough to hold all of the digits without crowding. Whether pre-printed or hand-written, digits should be positioned mid-way between the upper and lower edges of the box and clearly separated from each other, as shown in the example below.

These fields should not have a format statement. Delimiters and decimals are not allowed.

The 2 types of numeric fields are demonstrated in the following examples.

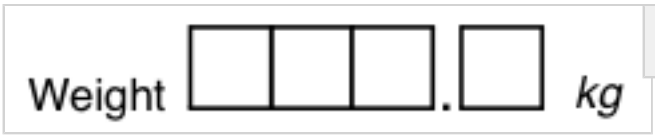
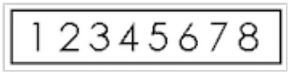
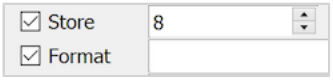
	For This CRF Field	*Length & Format*
1.		
2.		

Table 7.2. Numeric Field Examples

Format	Length	+ or - Sign	CRF Field Design	Value Range	Note
nnn.n	5	unsigned	[][][].[]	0.0 to 999.9	

Format	Length	+ or - Sign	CRF Field Design	Value Range	Note
snn.n	5	optional	□□□.□	-99.9 to 999.9	
Snn.n	5	required	□□□.□	-99.9 to +99.9	
2nn.n	5	unsigned	2□□.□	200.0 to 299.9	
none	4	optional	rectangular box	-999 to 9999	
nnnnnnnnnnnnnn	15	unsigned	□□□□□□□□ □□□□□□□□	0 to 281474976710655	For Subject ID number field only
nnnnnnnnn	10	unsigned	□□□□□□□□□□	0 to 2147483647	
Snnnnnnnnn	10	Required	□□□□□□□□□□	-999999999 to 999999999	
Snnnnn.nnn	10	Required	□□□□□□□□□□	-99999.999 to 99999.999	
snnnnnnnnn	10	optional	□□□□□□□□□□	-999999999 to 2147483647	
Snnnnnnnnnn	11	Required	□□□□□□□□□□□	-2147483648 to +2147483647	
none	10	optional	□□□□□□□□□□	-999999999 to 2147483647	

7.3.2.4. Specific Properties for String Fields

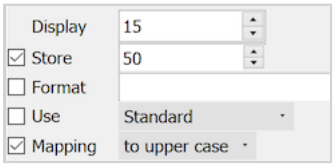
Display and Store Length. Because the width of computer screens is limited, the length of text fields displayed on screen is often set shorter than the maximum that can be entered and stored in the study database. Multi-line text fields can be created by holding the Control key (or Command key on Macs) while dragging out a rectangular box for the field. However, the space available on many CRF pages is only sufficient for a one-line text field. In such cases the height of the text field automatically expands into a multi-line text box when the user enters the field, and returns to its display size when the user leaves the field.

Format. Specifying a format is rare but possible for string fields. String formats use '^' as a place-holder for characters that can be entered by the user. All other parts of the format are considered fixed. For example, 'Model #^^^' could be used for a string field that must start with 'Model #' followed by up to 3 other characters.

If a format is specified its length must match the display and store length, with one exception. If the format ends in a single '^' character, as in 'Model: ^', the display and store length are honored and text can be entered from the position of the '^' character to the end of the store length.

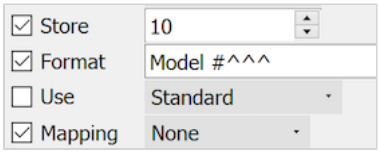
Use. String fields can have Use set to **Standard**, **eSignature Name** or **eSignature Reason**. **eSignature Name** is set and locked for the `SigName` field in the eSignature module. **eSignature Reason** is set and locked for the `SigReasonString` field in the eSignature module.

Mapping. Mapping can be used to automatically set all characters to upper- or lowercase as the user types.

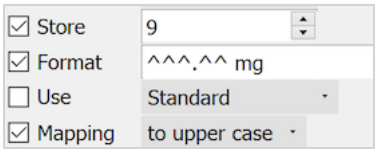


- This field can store a maximum of 50 characters in the study database.

- In the **DFexplore** data window the field is only 15 characters long but when the user moves into the field it automatically expands to display all 50 characters.
- All characters are automatically converted to uppercase as they are typed.



- This field begins with 'Model #' followed by a maximum of 3 characters.
- Examples: 'Model #123', 'Model #A-4', 'Model #7'



- Values entered into this field will be displayed with the full format only after the user enters all 5 digits.
- Examples: '325.00 mg', '082.50 mg', '000.25 mg'

7.3.2.5. Specific Properties for Time Fields

Format. A time format can be one of two choices - hh:mm or hh:mm:ss if seconds are also to be recorded by this field.

Use. Use will typically be set to **Standard**. If the time is the `SigTime` field of the eSignature module, then Use is locked to **eSignature Time**.

7.3.2.6. Specific Properties for Date Fields

Display and Store. Display and store length is always the same and depends on the date format.

Format. A date format is comprised of:

- 'dd' or 'DD' for day
- 'mm', 'MM', 'mmm', or 'MMM' for month
- 'yy', 'YY', 'yyyy', or 'YYYY' for year
- and may include optional spaces and/or delimiters between the parts

A date format is created by arranging these components to match the date format used in the study definition. DFdiscover supports day, month and year components arranged in any order and separated by any delimiter or no delimiter and thus can support all commonly used date formats.

A 3 character-month component indicates that English date acronyms will be used: Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec.

If lowercase is used the specified component can be filled with zeros to indicate that the value is unknown, while if uppercase is used the component cannot be unknown and zeros are not allowed. For example: if format is mmm/dd/YYYY then DEC/00/2008 can be entered for an unknown day in December 2008, and 000/00/2008 can be entered for an unknown day in 2008. It is not reasonable for the day to be known but the month to be unknown, nor for the day and/or month to be known but the year to be unknown, thus such date formats, and partial dates are not considered valid.

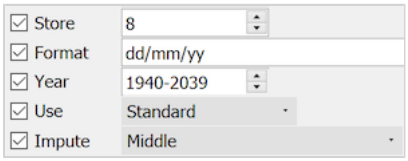
Delimiters between the components are optional and can include any character, including spaces. The following are all unusual but perfectly legal date formats: yymmdd, yy-dd-mm, yyyy: mmm.dd, Y=yy M=mm D=dd. However, before getting too creative you should consider what's legal in any other software you plan to use (like SAS).

Year. This property is used to specify the century for 2-digit years. If the box at the end of this property is checked the value specified in **Study** > **Global Settings** is used.

Use. A date can have three settings for the Use property: **Standard**, **Visit Date** and **eSignature Date**. In most cases Use is set to **Standard**, with the exception of two special ones: **Visit Date** indicates that this date field is to be used for scheduling, and **eSignature Date** is for a date field that is used to store the date that a record was e-signed (this use is generally locked in the `SigDate` field in the eSignature module).

Impute. When a partial date is entered by filling the day or day and month components with zeros a rule may be specified for imputing a valid date for use in edit checks, scheduling calculations, and when exporting the data. This property does not change the storage of partial dates, which always contain the zeros entered by the user. Thus it remains possible to export partial values without imputation even when imputation is specified. The imputation options include the following:

- Never: no imputation is performed, partial dates are invalid.
- Beginning: a missing day is imputed as the 1st day of the month, and a missing day and month is imputed as the 1st day of the year.
- Middle: a missing day is imputed as the 15th of the month, and a missing day and month is imputed as July 1st.
- End: a missing day is imputed as the last day of the month, and a missing day and month is imputed as the last day of the year.



- This date format is comprised of day, month, then year, with each component separated by a '/', e.g. 01/07/66 for July 1, 1966.
- The format uses 2-digit years which are interpreted as being in the range 1940-2039.
- Missing day and month components are allowed and imputed to the middle of the month or year whenever a valid value is required.

7.3.2.7. Specific Properties for Visual Analog Scales

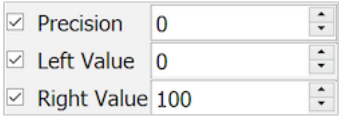
The only type specific property for visual analog scales is coding, which is comprised of 3 specifications:

Precision. the number of decimal places in the scale value.

Left Value. the minimum value, corresponding to a mark at the left end of the scale.

Right Value. the maximum value, corresponding to a mark at the right end of the scale.

Positive and negative integer values are allowed, thus a scale can be defined with a range of: -1 to 1, -10 to 10, -100 to 0, 0 to 7, etc.



- This example illustrates the most common scoring method, an integer in the range 0-100.

7.3.3. Edit Coding Table

7.3.3.1. Basic Editing

Basic editing of a code or label is started by single clicking the corresponding cell. Standard text editing shortcuts and context menu are accessible.

Validation. Entered data is validated, and the following errors are highlighted:

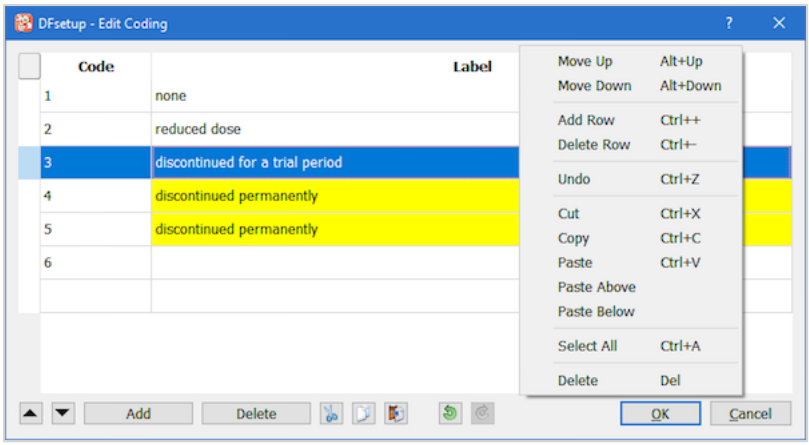
- Codes out of the valid range 0-65535 or empty codes. If not corrected, when the cell with such a code loses focus, the code with the label is erased.
- Duplicate codes are permitted during editing but the coding table may not be saved with duplicate codes.
- Duplicate labels, in particular empty duplicate labels, are permitted and no empty labels are allowed for numeric field codes.

Coding Table Menu. Additional functionality for coding tables is available from the coding table menu ( ) located at the upper-left corner of the coding table. The menu has three actions:

- **Edit Coding...** display the advanced table editor, described in the following section.
- **Remove Rows Without Labels** erase all entries with empty labels.
- **Clear** remove all table entries.

7.3.3.2. Advanced Editing

Selecting **Edit Coding...** displays the **Edit Coding** dialog. The dialog consists of a table with the data from the coding table and a set of tools for advanced editing. The basic editing of a cell can be activated by a double-click. For convenience of editing, if a code is deleted and the cell loses focus, the label is not erased. The single click action is reserved for selection or deselection of a cell or several cells with mouse gestures and optional **Ctrl** key.



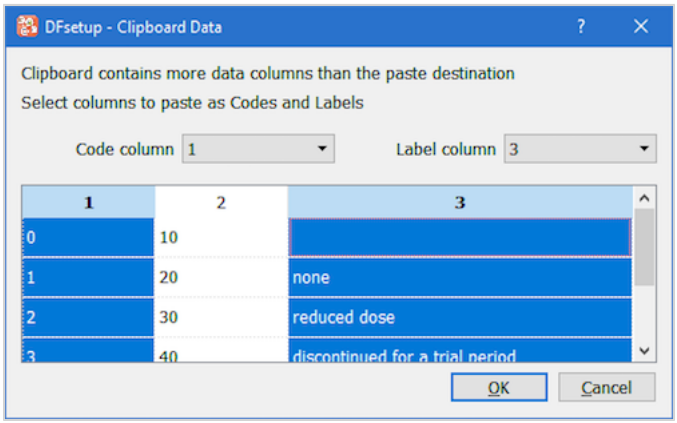
- : Move whole rows of the selected cells up (i.e. only code-label pairs may be moved), enabled when a cell or several cells are selected.
- : Move whole rows of the selected cells down, enabled with selection.

- Add: Add empty rows above the selected cells, enabled when a cell or several cells are selected contiguously; the number of rows with selected cells control the number of empty rows added.
- Delete: Remove rows from the table, enabled when whole rows are selected.
- and are enabled with selection.
- : The action depends on the selection made and the clipboard data pasted. The data is expected to be in the table format, i.e. cells are separated by a Tab symbol, each row ending with a line break. If no selection is made, the clipboard data will be appended to the existing data starting from the code column. If selection exists, the clipboard data will overwrite the existing data starting from the first selected cell, a cell with the lowest (row,column) index. In a situation when the clipboard contains a single column of data, and the first selected cell is a code, the data will be pasted as codes (if valid); if first selected is a label cell, the data will be pasted as labels. If the clipboard data and the destination do not match, the Clipboard Data assistant automatically opens, see the next sub-section.
- and are used to reverse (undo/redo) operations.

Selection dependent context menu and shortcuts are also available.

7.3.3.3. Clipboard Data Assistant

The assistant allows you to choose columns to be pasted when the clipboard data does not match the destination. Drop-down lists and mouse gestures may be used to select the columns.



7.4. Edit checks

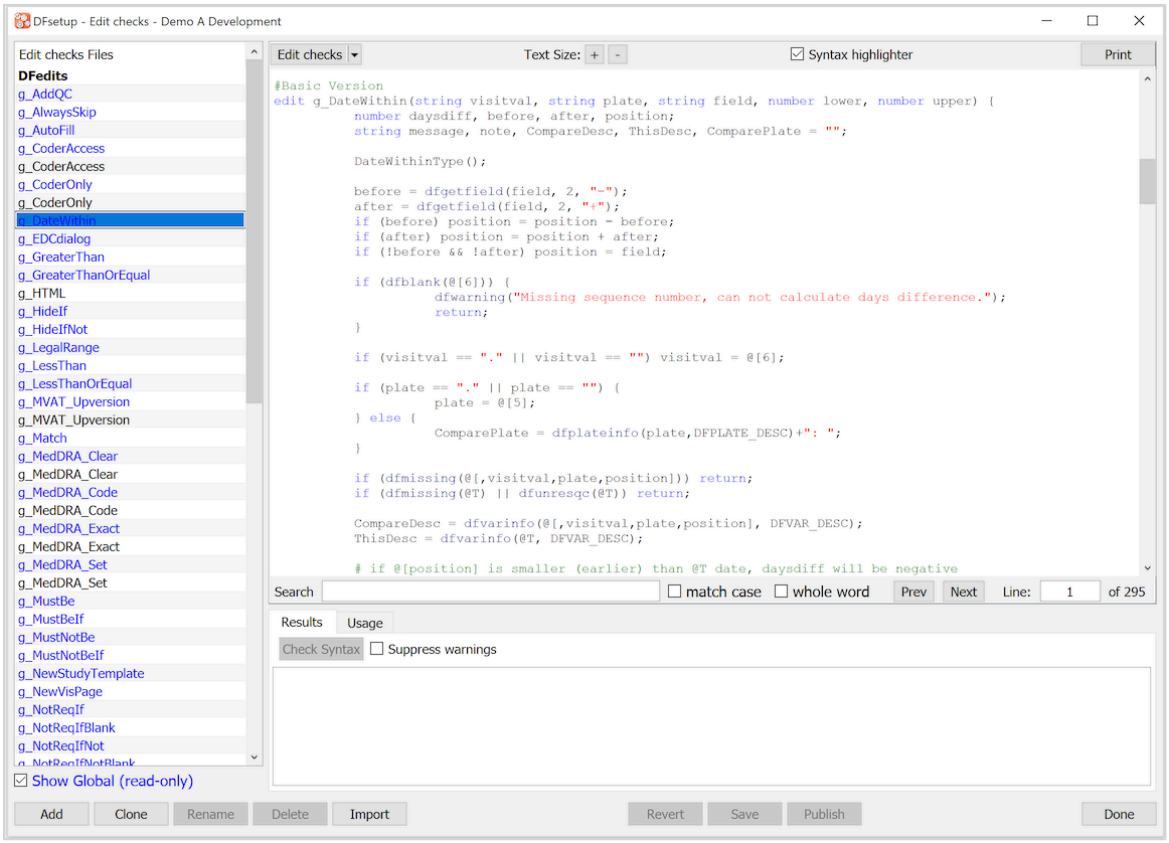
The Edit checks dialog is used to enter, verify and publish edit checks for use during data review.

In the edit check dialog, illustrated below, edit check source files are listed in the left panel. Edit checks specific to the current study appear in black, while global edit checks, which are available to all studies, appear in blue. Global edit checks can only be defined, modified, cloned and deleted by a DFdiscover administrator, but they can be viewed by study administrators and regular users who have been granted permission to use the edit checks editor in **DFsetup**.

The special source file, **DFedits**, appears at the top of the list. It must be present - without it, it is not possible to publish edit checks. All other edit check source files must be referenced in DFedits using `# include` directives.

Study and global edit check source files are stored in the `STUDY_DIR` and /opt/dfdiscover `ecsrc` directories respectively on the DFdiscover server. When a source file is selected it is retrieved from the server and displayed in the text edit window where it can be modified and saved back to the server.

For documentation on edit check programming see *Programmer Guide, Edit checks*.



The buttons at the bottom of the dialog are used to manage the edit check source files. These buttons include:

- Add: create a new empty edit check file.
- Clone: create a copy of the selected edit check file.
- Rename: rename the selected edit check file.
- Delete: delete the selected edit check file.
- Import: import a copy of an edit check file from another study.
- Revert: reload last saved version.
- Save: commit changes made to an edit check file to the server. As for all configuration dialogs, saves are immediate.
- Publish: make edit checks available to **DFexplore**. Confirm that there are no syntax errors before publishing.
- Done: dismiss the dialog.

When a source file is selected, the code is displayed in the main window where it can be modified, searched and printed. If **Syntax highlighter** is checked, the code is colored: green for comments, blue for reserved words in the edit check language, red for quoted strings, and black for everything else.

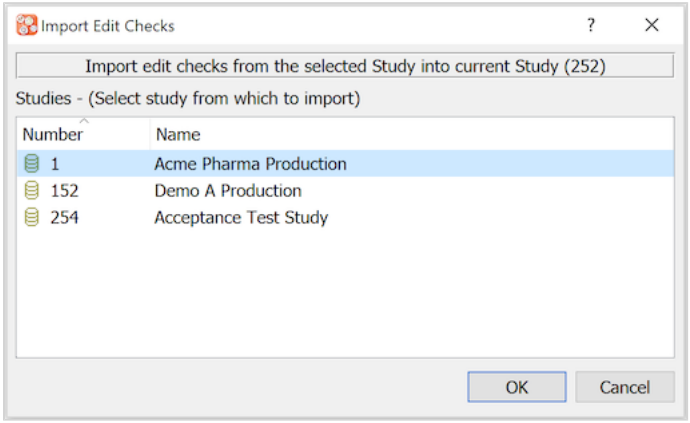
To find an edit check or function within a source file that contains many edit checks and functions, select the edit check or function name from the **Edit checks** drop-down. The window is scrolled so that the match is positioned at the top.

To find all of the data fields that use the current edit check, switch to the **Usage** view and then click Usage. Each of the data fields that uses the edit check is displayed in the scrolling list. Double-clicking any of the rows highlights the referenced plate and data field in the main application window.

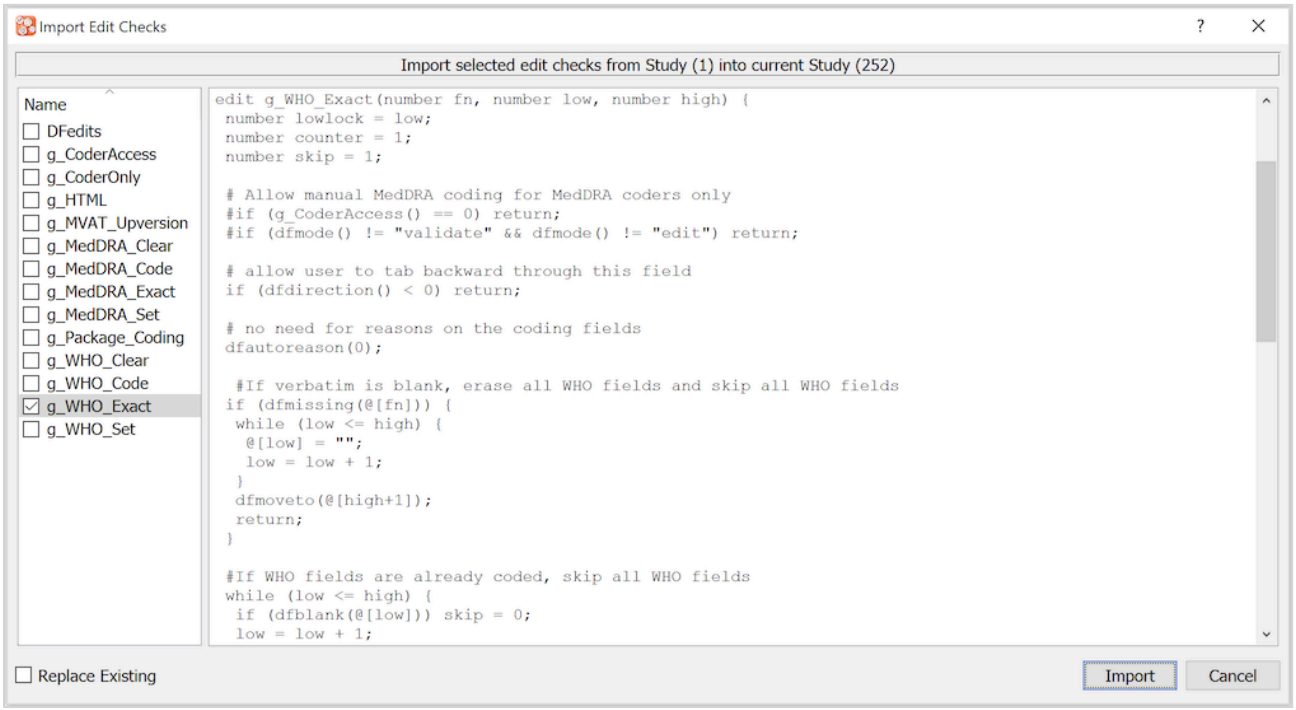
Any edit check source file can be checked for syntax errors by selecting it and clicking Check Syntax. Errors and warnings are presented in the **Results** view. When checking a source file other than DFedits, references to edit checks or functions that are in other source files will be flagged with warnings. Check **Suppress warnings** to prevent this behavior, which may make it easier to locate errors.

Users with 'DFdiscover-Developer' permission are able to reload published edit checks while working in **DFexplore**. This is particularly useful when testing edit checks. Users without 'DFdiscover-Developer' permission get the most recently published version of the edit checks when they login to **DFexplore**.

Clicking Import displays a dialog of other studies from which edit checks might be imported.



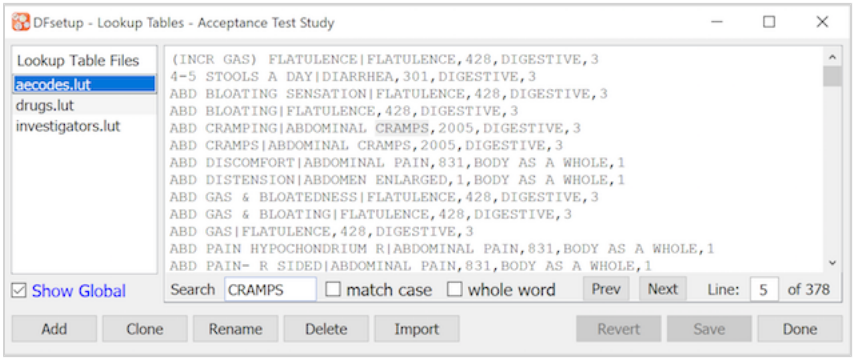
When a source study is selected the edit check source files for that study are listed in a separate dialog. Any of these files can then be selected, reviewed and copied to the current study. Check **Replace Existing** to overwrite any edit checks with matching names in the current study.



7.5. Lookup Tables

Study lookup tables can be created and modified using the dialog illustrated below. All lookup tables registered in the study and DFdiscover lut directories are listed in the left panel. When a lookup table is selected it is retrieved from the DFdiscover

server and displayed in the text edit window on the right. Only study level lookup tables can be modified and saved. Lookup tables stored at the DFdiscover level are displayed in blue. These files must be managed by a DFdiscover administrator.



- Add: create a new empty lookup table
- Clone: create a copy of the selected lookup table
- Rename: rename the selected lookup table
- Delete: delete the selected lookup table
- Import: copy a lookup table from another study
- Revert: reload last saved version
- Save: save changes to the selected lookup table
- Done: put the dialog away

All of the actions performed by the buttons have permanent effects on the contents of the study lookup table directory on the DFdiscover server. If you delete a lookup table by mistake you will have to recover it from your most recent server backup.

Lookup tables are used by edit checks which can retrieve an entry matching a specified search string or display the entire lookup table to allow users to make a selection during data entry.

Lookup tables are stored in the study lookup table directory (`study`/lut) as plain text files structured in one of 3 ways:

1. **2 Field Lookup Tables**

   Each row in these lookup tables has 2 fields separated by a '|' character. The first field is what the edit check searches for and the second field is what it returns. In the example illustrated below, if the edit check function `dflookup` was asked to search for 'CHF' it would find and return the string 'congestive heart disease'.

   ```
   CAD|coronary artery disease
   CHF|congestive heart failure
   ```

2. **Single Field Lookup Tables**

   In lookup tables that do not contain the '|' field delimiter the text entry in each row serves as both the search and return value. Examples include lookup tables that contain the names of investigators, medications or procedures that are valid values for a particular data field.

3. **Multi-Field Lookup Tables**

   A flexible, multi-field lookup table structure can be utilized in DFdiscover.

```
#N:LL Term|Pref Term|SOC Term|LL Code
#L:Low Level Term|Preferred Term|System Organ Class|Low Level Code
#F:1|1-4
Abdomen enlarged|Abdominal distension|Gastrointestinal disorders|10000045
Abdomen mimicking acute|Acute abdomen|Gastrointestinal disorders|10000047
Abdominal cramp|Abdominal pain|Gastrointestinal disorders|10000056
```

These lookup tables contain 3 header rows that define the structure of the table.

- The first row begins with '#N:' followed by a short column name for each field. This appears above the scrolling list of entries in the lookup dialog displayed by an edit check.
- The second row begins with '#L:' followed by a longer descriptive label for each field. This appears in the top section of the lookup dialog where the field values for the current row are displayed, and where the user can indicate which fields to search.
- The third row begins with "#F:" followed by 2 fields: the 1st lists the fields to be searched by `dflookup` for a match and the 2nd lists the fields to be returned when a match is found or selected by the user. If multiple return fields are specified they are returned to the edit check as a '|' delimited string, which can be parsed using `dfgetfield`.

The return value can also be parsed in older lookup tables. In the example shown below, if `dflookup` is given the search string 'CHF' it will return 'congestive heart disease, heart', which could then be split into 2 separate strings: 'congestive heart disease' and 'heart' using `dfgetfield`.

```
CAD|coronary artery disease, heart
CHF|congestive heart failure, heart
KF|kidney failure, kidney
```

In addition to lookup tables used by edit checks, lookup tables can also be created for standardized queries and reasons, which can be selected when creating a new query or reason in **DFexplore**. These must be 2 field lookup tables consisting of a short acronym followed by the query or reason:

```
DATE|This is not a valid date. Please correct or clarify.
ELIG|This subject does not appear to be eligible. Please correct or clarify.
REQ|This value is required and cannot be left blank.
```

Lookup tables are assigned an arbitrary descriptive file name when they are created and a second typically shorter name by which they are called in edit check function `dflookup`. This pairing of names is done in a separate configuration file, the Lookup Tables Map. A lookup table that is not registered in this map will not be available to edit checks.

7.6. General Dialog Controls

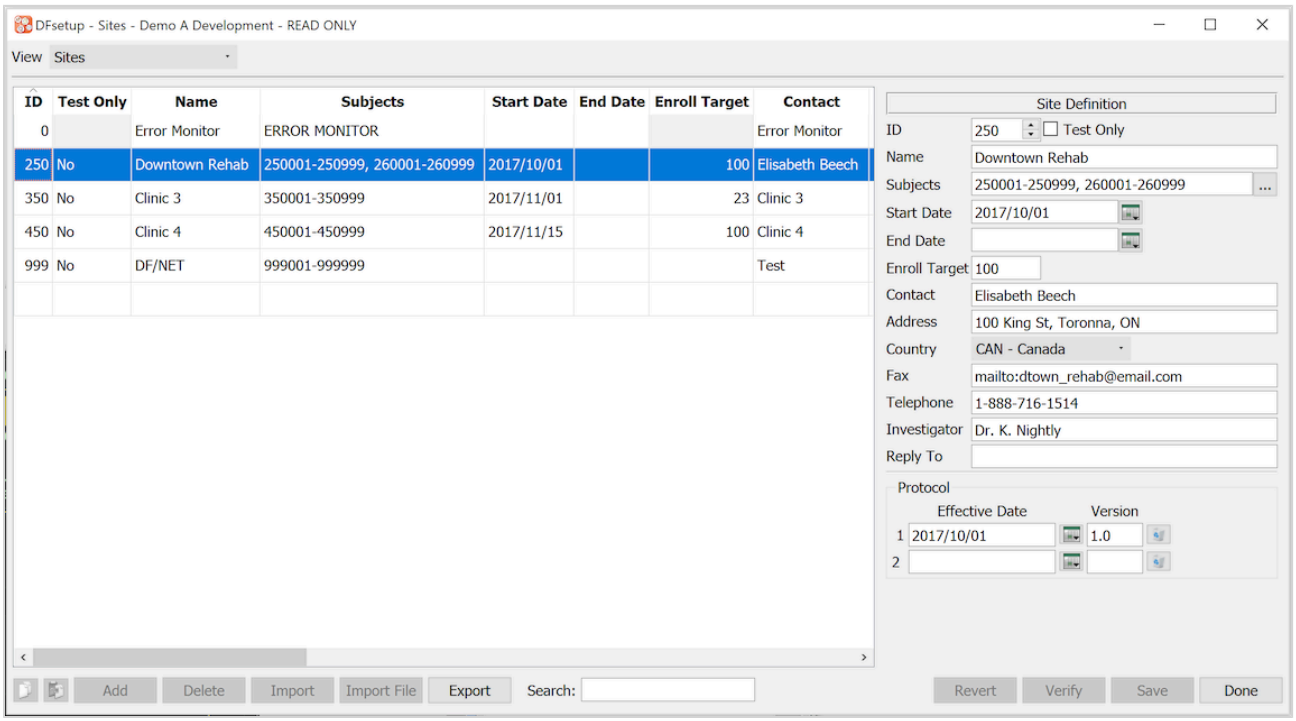
The study configuration dialogs described in the following sections have the same controls:

- **View**: select a study configuration dialog
- **Copy**: copies the selected row ()
- **Paste**: insert previously copied row above the current row ()
- Add: open a new blank row below the current row
- Delete: delete current row

- Import: replace all rows with sites from another study
- Import File: add or replace rows with site info from a file
- **Search**: case insensitive string match on all fields
- Revert: reload last saved version
- Verify: check last saved version
- Save: save to DFdiscover server
- Done: quit the dialog

7.7. Sites

The Sites dialog is used to enter contact information and subject IDs for each clinical site participating in the study. Associating subject IDs with sites is a critical component of this dialog.



Click Save to commit all changes to the DFdiscover server. Saves can be performed site by site or delayed until all changes have been completed. However, save is not possible if any of the required fields are left blank for any site.

The verification report checks the current version of the sites database stored on the DFdiscover server; ensure that the most recent changes are saved before clicking Verify.

At least 1 site must be specified for each study. If no sites are specified **DFexplore** will automatically assign all subjects to site number 0.

The specifications for each site can be changed at any time during the study, including while data entry is occurring in **DFexplore**. If a subject ID is moved from one site to another while someone is doing data entry in the old site, the subject will continue to be available to that user until the study is closed. Thus it is possible that the subject may appear in **DFexplore**

under both the old and new sites for a short period of time. However, record locking will continue to ensure that only one user at a time can enter or modify the subject's data.

When a subject ID is moved from one site to another, any outstanding queries will be included in the next Query Report sent to the new site, and all queries (regardless of status and which site created them) will continue to be visible when the subject is viewed in **DFexplore**. However, the site field in query records is only updated if the query is included in a Query Report for the new site, or is modified and saved in **DFexplore**.

7.7.1. Site Specifications

The components of each site specification include:

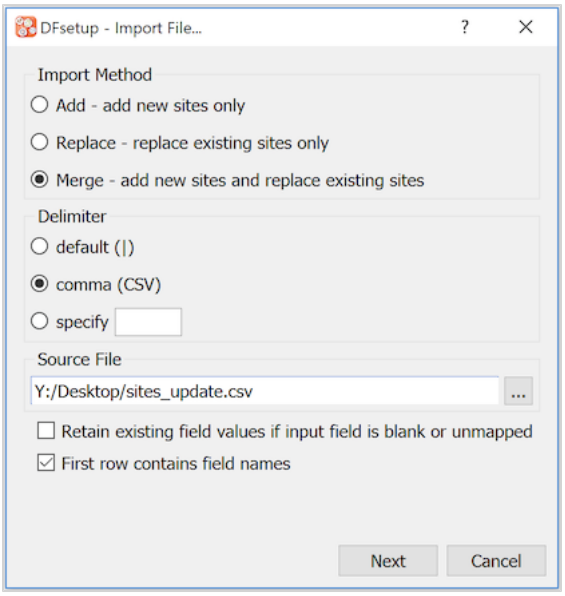
- **ID [required].** numeric value in the range 0-21460 inclusive, which uniquely identifies each clinical site participating in the study.
- **Test Only.** if checked, data for this site is omitted from data exports when the -X option is used. This applies to **DFexport.rpc**, **DFexport**, **DFsas** and **DFsqlload**.
- **Name [required].** name of the site, clinic, hospital or institution.
- **Subjects [required].** comma delimited list of all subject IDs and ranges belonging to the site. Each subject can belong to one and only one site.
- **Start Date.** the date, in yyyy/mm/dd format, for when the site is expected to start, or has started, enrolling subjects.
- **End Date.** the date, in yyyy/mm/dd format, for when the site is expected to stop, or has stopped, enrolling subjects.
- **Enroll Target.** number of subjects that the site is expected to enroll.
- **Contact [required].** the name of the primary contact to whom Query Reports will be addressed.
- **Address.** postal/street address.
- **Country.** 3-character code from the ISO 3166-1 list of approved country codes. The categorical setting can be used for grouping in reports.
- **Fax.** space-delimited list of fax numbers and/or email addresses, used by **DF_QCfax** to deliver DFdiscover Query Reports. The field may be left blank for sites that are not meant to receive Query Reports. If fax numbers are specified, each fax number must contain all of the digits required to dial the destination fax machine (i.e. long distance code, country code, area code - as needed) and may be formatted to aid readability e.g. (789)-123-4567. Each email address must start with the prefix **mailto:** followed by the complete email address, e.g. mailto:jack@somewhere.com. Fax numbers and email addresses cannot contain spaces.
- **Telephone.** telephone number of the contact person.
- **Investigator.** the name of the principal investigator at the site.
- **Reply To.** email address of the person at the study central office who will receive any replies to Query Reports that were sent to the site by email. This value appears in the `From:` address of emails that deliver Query Reports to the email addresses specified in the Primary Fax list. This provides a real email address that the site can respond to and is also the address that will receive email transmission failure notifications. If a Query Report is sent via email and this field is blank, reply emails from the site, or email transmission failures, will be delivered to the DFdiscover problem mail recipient defined by the DFdiscover administrator in **DFadmin**.
- **Protocol.** up to 5 protocol version and effective date pairs. The **Effective Date** is the first date on which the **Version** is in place at the site. Multiple **Effective Date** are expected to be in chronological order. The values are accessible via edit check functions and are implicitly used in the `dfprotocol` function.

Error Monitor

A special entry must be included in the sites database for an error monitor who is typically someone at the study coordinating site. This special site is identified by the string **ERROR MONITOR** in the Subjects field. Any data records with subject IDs that do not belong to one of the clinical sites are automatically assigned to the error monitor site. This might occur for example if the sites database is modified after subjects have been entered. Only one error monitor site can be defined for each study. By default site number zero is used for this special site, but this can be changed to any valid site number.

Import File

Site information can be imported into the sites dialog from a text file by using Import File. The file may be in DFdiscover, CSV or another delimited format, and may contain only some or all of the required fields, in any order. When this option is selected the following dialog appears.



When using 'Replace' or 'Merge', if the input file lacks some of the required fields, or you wish to ignore some of the input fields and keep existing values, select 'Retain existing field values'. When Next is selected a dialog will appear that can be used to identify how fields in the input file map to fields in the sites dialog. When the mapping has been completed select Import to load the records into the sites dialog. Any new records and modifications to existing fields will be shown in bold. Select Save to write the changes to the DFdiscover server, or Revert to undo the changes and reload the existing site records into the sites dialog.

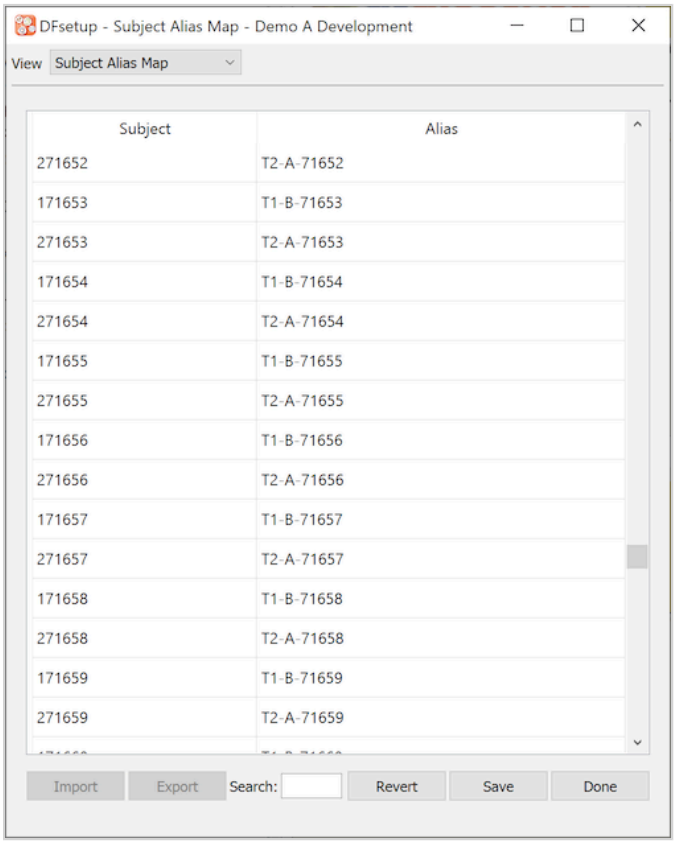
7.8. Subject Alias Map

This optional dialog is used for the specification of the subject alias map, the mapping of unique numeric subject identifier to unique subject alias.

Each alias must be unique, can be up to 30 UNICODE characters in length, and must start with a digit or letter (and not a punctuation character). The following symbols are not allowed in a subject alias:

| & * < > [ ] ? % / \ " ` # , ;

In order, these symbols are: space, vertical pipe, ampersand, asterisk, less than, greater than, open brace, close brace, question mark, percent, forward slash, backward slash, double quote, back quote, number sign (octothorpe), comma, and semicolon.



To define a new mapping, enter two values in a blank row of the table, specifying a unique numeric identifier for the **Subject** and a valid, unique subject alias for the **Alias**. Existing mappings can be re-defined by cicking on any cell and entering a new value, either id or alias.

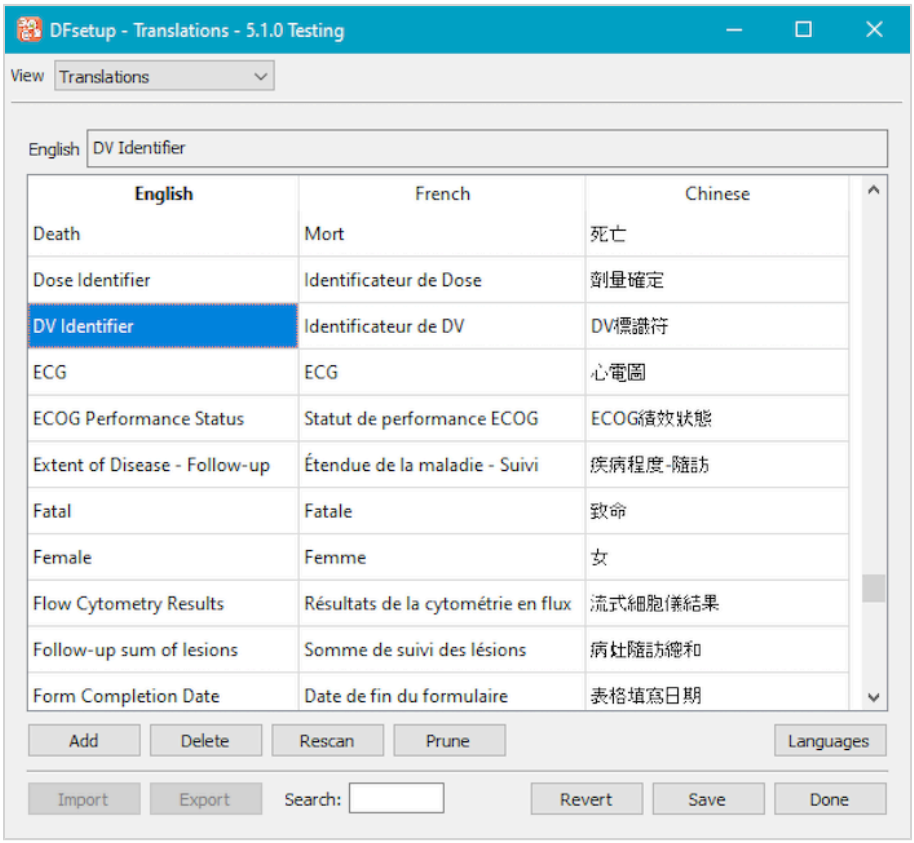
Each mapping row must define both a subject id and an alias. If only one is defined, the mapping cannot be saved. Any blank rows in the mapping are automatically discarded during save.

It is also possible to import an externally-defined csv file (comma-delimited) or a plain text file (|-delimited) to define the mapping. Each import clears the existing mapping and defines a new mapping. It is not possible to incrementally add to an existing mapping by importing additional files - each import replaces all of the existing mapping.

7.9. Translations

This dialog is used to provide translations for text from a subset of field properties in setup. Text translations can be entered for the following properties of data fields: prompt, description and coding labels. These translations are presented in API clients that support this feature (such as DFengage) by first allowing the user to select from the available languages. If any translation is not available for user's language selection the default text (from the first column in the dialog) is returned.

In each row the first column contains words and phrases found in the following data field properties: prompt, description and coding labels. Translations for these phrases can be entered in the columns after the first, corresponding to each language. The phrases in the first column are unique. Each column header identifies the unique language with a label that was previously defined by clicking Languages and completing the dialog. Pipe (|) symbols are not legal and cannot be entered in any translation.



New rows can be inserted using Add or existing rows deleted using Delete. New rows can also be added by typing in the blank row after the last row. All blank rows are automatically discarded during save.

Click Rescan to fetch all phrases from setup's field Prompt, Description, and Coding Labels properties. The scan appends any new phrases. When this dialog is opened and no languages or translations exist, Rescan is triggered and populates the phrases in the first column (the default language). The phrases are always presented in alphabetic sort order after a scan.

Click Prune to remove any rows containing unused phrases. This action also deletes any blank rows.

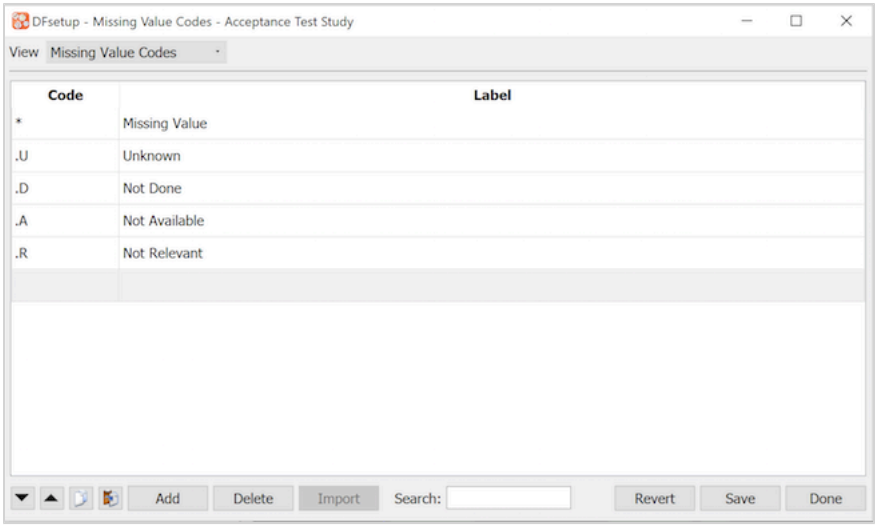
An externally-defined CSV, or |-delimited text, file can be imported. Each import clears the existing translations and starts a new view with the file contents. It is not possible to incrementally add to an existing translations by importing additional files - each import replaces all of the existing translations. The first row of the imported file is interpreted as the language names and is displayed as the column header in the translations view.

To manage languages, double-click the column header or click Languages. Language names must be unique, cannot start with a number, cannot contain special characters, and may be a maximum of 15 characters in length, To add a language click +. This starts a new column to be completed with translations of the phrases in the first column for that language. Click - to delete the highlighted language. This will also delete all translations for that language (the entire column) - be certain of this step before confirming.

7.10. Missing Value Codes

This dialog is used to enter missing value codes and labels. In **DFexplore** missing value codes can be assigned to any optional or required data field, and can be thought of as pre-approved reasons for missing data. **DFexplore** does not allow missing value code assignment to essential fields.

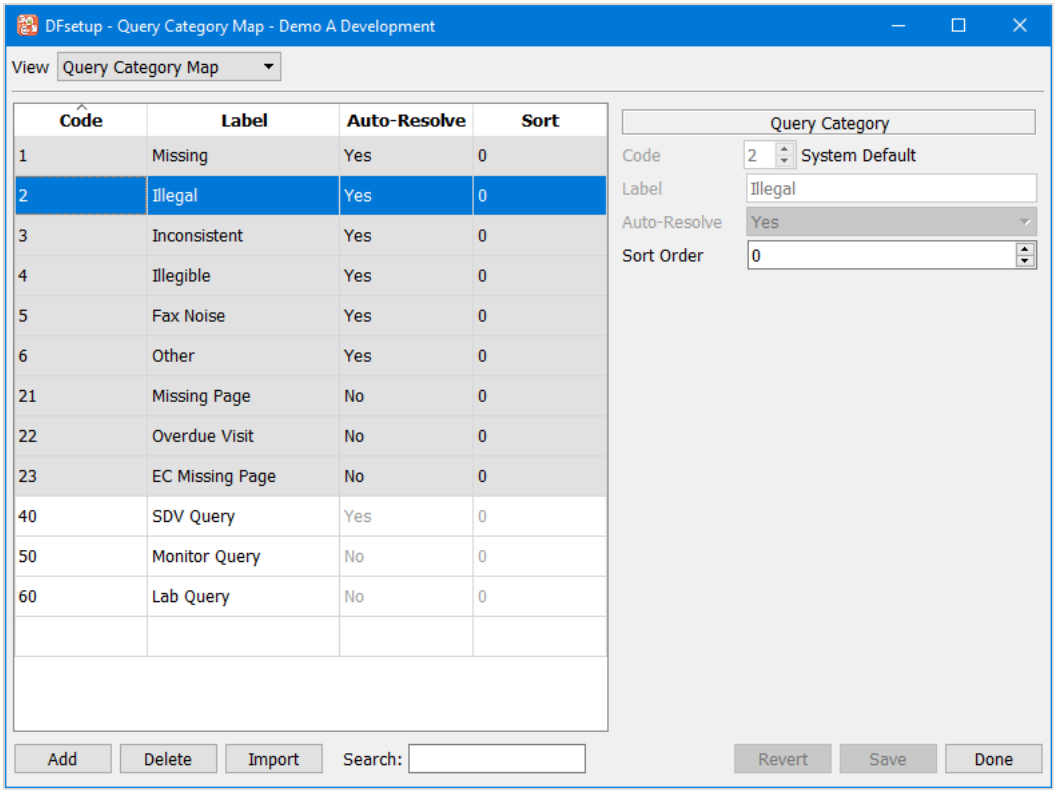
DFdiscover imposes no restrictions on the choice of missing value codes but all data fields use the same missing codes thus be careful not to define something as a missing code if it might be a valid entry for some data fields. Also consider restrictions imposed by other software you plan to use (like SAS®).



The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

7.11. Query Category Map

This dialog is used to design custom query categories for a study. These are in addition to the predefined categories: Missing, Illegal, Inconsistent, Illegible, Fax Noise, Other, Missing Page, Overdue Visit, and EC Missing Page. DFdiscover predefined query categories cannot be deleted.



Query categories are entered as follows:

1. **Open a New Row**

   Either select the last row of the table (always empty) or another row and click Add to open a new empty row below it.

2. **Code**

   A code value ranging from 30 to 99 (inclusive) must be entered in the 1st column of the table or in the Code section of the Query Category Map panel on the right. Codes do not need to be entered in sequential order. The same code value may not be used twice.

3. **Label**

   A Label must be entered in the 2nd column of the table or in the Label section on the right. This label will be used as the identifier for the category in the option list available when adding a query in **DFexplore**. This label must have 20 characters or less and the same label may not be used twice.

4. **Auto-Resolve**

   The auto-resolve property controls whether or not adding a reason to an external query of this type also changes the query status to 'pending'. Types may be set to be resolved automatically by selecting 'Yes' in the Auto-Resolve option list on the Query Category Map panel.

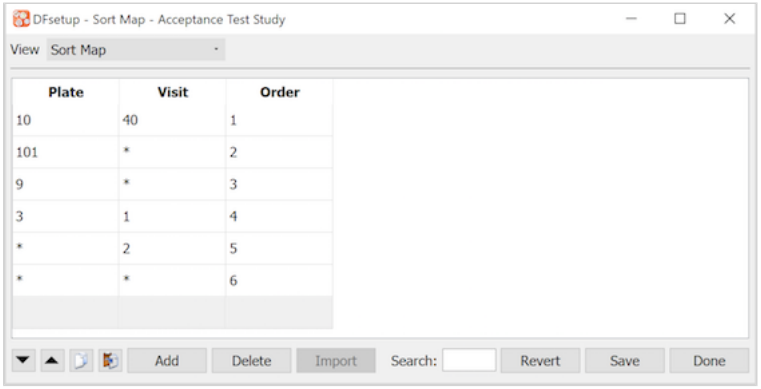
5. **Sort**

   The Sort option is used to control the order in which the categories will appear in the option list available when adding a query in **DFexplore**. Categories may be assigned any integer value between -2147483648 and 2147483647. They will be displayed in **DFexplore** in ascending by sort value, then by code value (types with higher positive values will be placed at the bottom of the option list, while categories with high negative values will be placed at the top of the list. 0 is the default sort value, as well as the value assigned to the predefined types. Sort values may only be entered in the Sort section of the Query Category Map panel.

A user-defined query category may be deleted; however, any query with the deleted category will be re-labeled with a default label of "Query category ##", where "##" is the code value of the deleted category. It is important to consider the impact caused by the loss of the category label when deleting user-defined query categories.

7.12. Sort Map

The Sort Map dialog is used to specify the order in which data queries are to appear within each subject in Query Reports.



In this example queries are printed:

- 1st for plate 10, visit 40
- then all queries on plate 101
- then all queries on plate 9
- then plate 3, visit 1
- then all other visit 2 plates
- then all other queries

When an asterisk '*' is used to signify all plates or visits, queries are printed in ascending numerical order for plates or visits within that sort map entry. Thus in the example queries on plates 101 and 9 will be printed in ascending order by visit number, with all plate 101 queries printed first.

The last entry is not required. It represents the default order that prints queries sorted in ascending numerical order by plate within visit.

The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

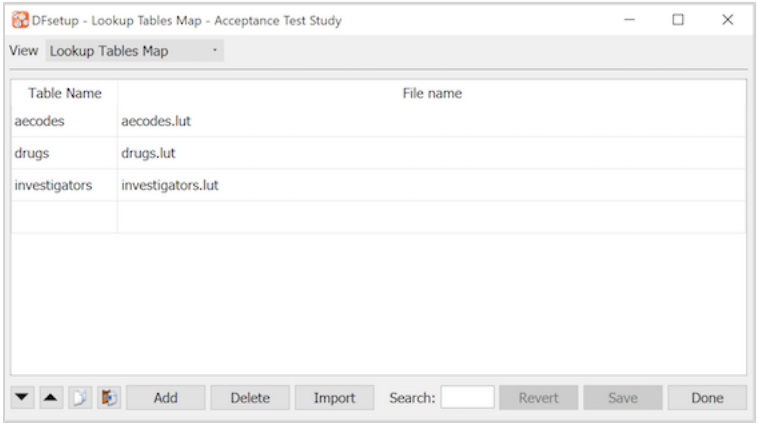
7.13. Lookup Tables Map

The dialog illustrated below, is used to enter a lookup table name by which each lookup file will be known to DFdiscover.

There are four (4) reserved lookup table names:

- `QC` for standard queries,
- `QCNOTE` for standard query note text,
- `QCREPLY` for standard query reply text, and
- `REASON` for standard reasons.

These names tell DFdiscover where to find these lookup tables when a user clicks the lookup buttons in the query and reason dialogs. All other names are for use in edit checks and can be created using any string of letters and numbers.



- A table name (left column) must be specified for each lookup table file (in the right column).

- The 'lut' file name extension shown in this example is not required.

The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

7.14. Visit Map

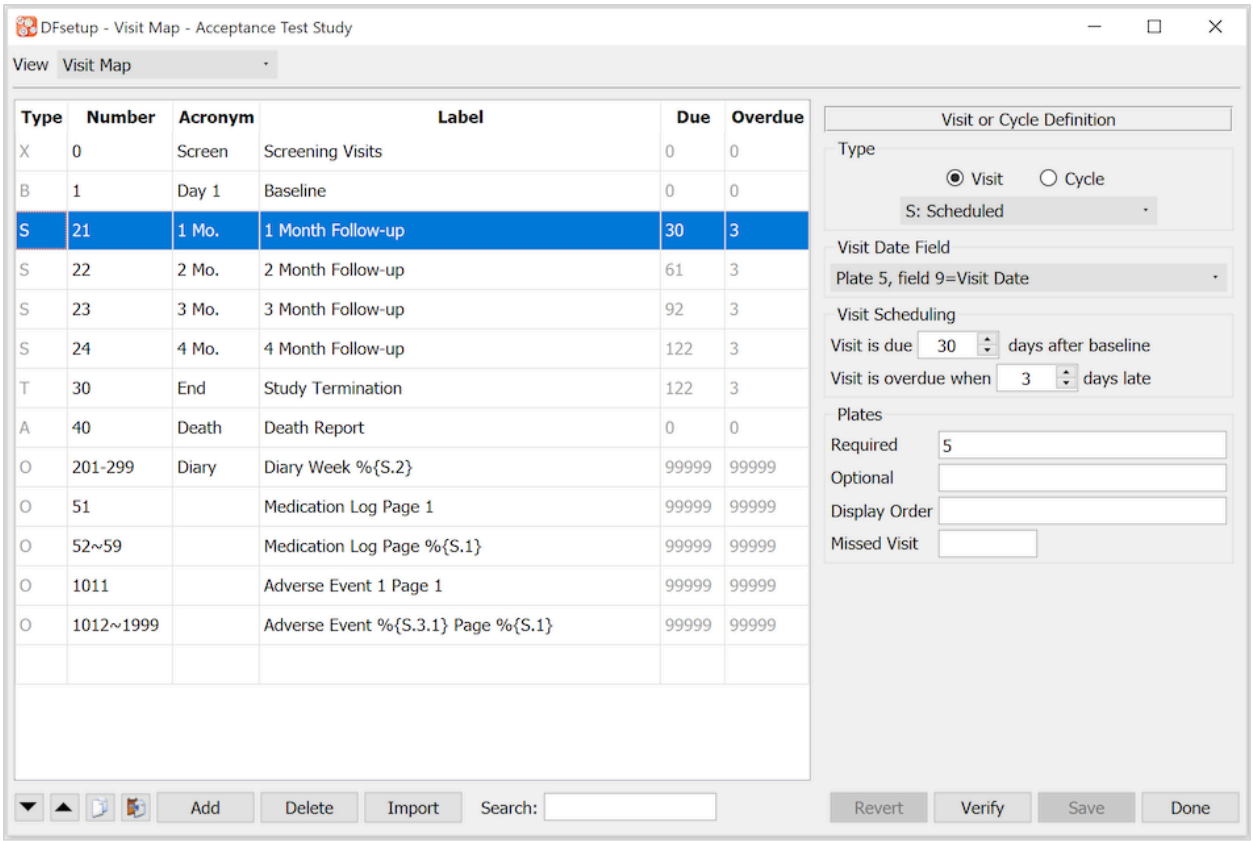
The Visit Map dialog is used to enter specifications describing the set of study CRFs and subject scheduling requirements. It must include all subject visits and repeated forms, and each visit must include a list of all CRF plates that are required or optional.

The visit map specifications are used to generate a blank CRF binder in **DFexplore** for each new subject and is thus required before data entry can begin. The order of visits in each subject binder corresponds to the order in which they are entered in the visit map. The order of pages within each visit corresponds to the Display Order specified for the plates of the visit. If no Display Order is specified, plates will be displayed in the default order of all required plates for the visit followed by all optional plates for the visit.

The study visit map is also used by report **DF_QCupdate** to calculate the follow-up schedule for each subject, and to generate data queries for missing pages and overdue visits.

A visit map can be simple, comprised of a few easily defined visits as illustrated below, or complex with multiple cycles and conditions that determine what is required and when. Detailed documentation of DFdiscover visit map features can be found in the appendix at Subject Visit Scheduling. It should be studied carefully and a visit map plan should be created before finalizing the study CRFs.

This section describes how to use **DFsetup** to enter and check your visit map specifications, it does not attempt to explain the different visit types or scheduling options, nor the rules that determine how they are used - for this see Subject Visit Scheduling.



The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

Visits are entered as follows:

1. **Open a New Row**

   Either select the last row of the table (always empty) or another row and click Add to open a new empty row below it. The order in which visits appear in the table must correspond to the order in which they are performed during the study. The location can be adjusted after the visit is defined by using the up and down arrows.

2. **Visit or Cycle**

   Select the Visit button at the top of the Visit or Cycle Definition panel.

3. **Visit Type**

   Select the appropriate visit type: screening, baseline, scheduled, optional, etc. from the option list below the Visit button. Each visit type has special meaning and behavior. The one letter acronym for the selected visit will appear in the 1st column of the visit map table.

4. **Visit Number**

   Enter the visit number in the 2nd column of the table. Some visits use a series of numbers to distinguish different reports, as illustrated for the medication change logs, adverse event logs and subject diaries in the example shown above. A visit number series can be entered using a dash (-) or a tilde (~). If a dash is used **DFexplore** adds a new blank page with the next visit number as soon as the last page is entered. If a tilde is used the visit number series does not appear in the **DFexplore** record navigation list. To enter these assessments the user must select **Assessment** > **Add New Assessment** and select the desired assessments from a dialog. This method is useful for assessments that occur rarely or that you do not want to display in the **DFexplore** navigation list. It can even be used for single assessments, like the death report shown in the example.

   It is legal to enter a comma-separated list of visit numbers and ranges. However, it is not possible to mix dash (-) and tilde (~) ranges in the same list. If a dash range appears anywhere in the list, any tildes will also be interpreted as though they were dashes.

5. **Visit Acronym**

   A short acronym can be entered in the 3rd column of the table for those visits that will appear in the subject status list of the standard DFdiscover Query Report. This value is used to identify the first, last and next scheduled visits.

6. **Visit Label**

   A visit label must be entered in the 4th column of the table. It is used in the subject binder in **DFexplore** and in overdue visit queries and thus needs to clearly identify each visit.

7. **Visit Date Field**

   If the visit is scheduled or has a date that signals termination of subject follow-up a visit date field must be specified. Select the appropriate visit date from the list of database fields that have been defined with the use property set to Visit Date.

8. **Visit Scheduling**

   If a visit is scheduled a scheduling method must be specified. Most visits are scheduled a specified number of days following the cycle baseline visit, but there are two exceptions.

Screening visits can be unscheduled or scheduled a specified number of days following the first screening visit - which itself cannot be scheduled.

The last cycle termination visit is sometimes defined as occurring within a specified date window using a type W visit, for which the scheduling options are illustrated in the following examples (which all assume that the study VisitDate format is yyyy/mm/dd).

- on a specific date: e.g. `on 2008/12/01` This schedules the final follow-up visit on Dec 1, 2008 for all subjects.
- before a specific date: e.g. `before 2008/12/01` This converts the last scheduled follow-up visit for each subject before Dec 1, 2008 into the termination visit.
- after a specific date: e.g. `after 2008/12/01` This converts the first scheduled follow-up after Dec 1, 2008 into the termination visit.
- between two specific dates: e.g. `between 2008/11/01~2008/11/30 .25` In this example all final follow-ups are scheduled for November 2008. The scaling factor (.25 in the above example) spreads the visits out over the termination interval. It should be set to the termination window width (days) divided by the usual inter-visit interval (days). In the above example a scaling factor of .25 would be appropriate for the 1 month termination window if the usual gap between follow-up visits is 4 months.

9. **Overdue Allowance**

   Visits are considered overdue when the overdue allowance has expired. Thus for example if a visit is scheduled to occur between day 100 and 110 (after baseline) it can be scheduled to occur on day 100 with a 10 day overdue allowance. The overdue allowance should be increased in a paper-based study to allow additional time for completion and faxing of data forms and level 1 data entry by central data management staff.

10. **Required Plates**

    A comma delimited list of plate numbers and ranges can be specified, e.g. 5,10-13,20,27-29. The order determines the order in which required plates will appear in the subject binders in **DFexplore**. If a Display Order is specified, plates will appear in the order corresponding to the Display Order.

11. **Optional Plates**

    A comma delimited list of plate numbers and ranges can be specified. The order determines the order in which these optional plates will appear in the **DFexplore** subject binders, where optional plates are identified by circular icons and listed below the required plates identified by square icons. Optional plates must be included in the visit map to make them available for data entry.

12. **Display Order**

    A comma-delimited list of required and optional plate numbers and ranges can be specified. The order of plates specified here will determine the order in which the plates will be displayed in the subject binder in **DFexplore**. Required and optional plates can be inter-mixed in the Display Order to customize the order of pages within each assessment. Not all required and optional plates for the visit need to be specified in Display Order. If Display Order does not contain a list of all required and optional plates, the plates listed in Display Order will come first, followed by all required plates excluded from Display Order, then all optional plates excluded from Display Order. If Display Order is empty, the default plate order (all required plates followed by all optional plates) will be used to create the subject binder.

13. **Missed Visit Plates**

    If a special missed visit plate has been defined to record reasons for missed visits it will be used when a users sets a visit as missed in **DFexplore**, if not that the default missed visit dialog will be used. Only 1 missed visit plate can be defined for a visit, but it is possible (but unusual) to have different missed visit plates for different visits.

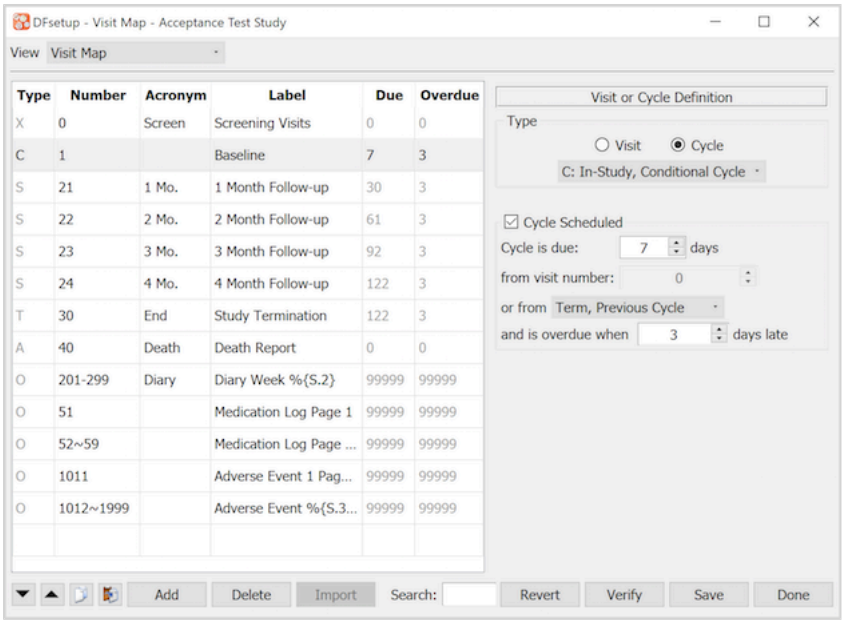
Cycles are containers for visits with a common baseline.

If a visit map includes only one baseline visit, cycles need not be defined. But if visits are grouped into sets each with their own baseline and scheduling then cycles must be used.

Cycles may be required, optional or conditional and may be scheduled in a number of ways. Different subjects may move through a different set of cycles during the course of the study based on conditions triggered by data values recorded in the study database.

This section describes how to use **DFsetup** to enter and check your cycle specifications, it does not attempt to explain how the different cycle options work or what they mean - for this see Subject Visit Scheduling.

The example below shows the definition of a conditional cycle for long-term follow-up after completion of the treatment cycle.



The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

Cycles are entered as follows:

1. **Open a New Row**

   Either select the last row of the table (always empty) or another row and click Add to open a new empty row below it. The cycle record must immediately precede the visits that comprise the cycle. The order in which cycles appear in the table must correspond to the order in which they are performed during the study. The location of the cycle record can be adjusted after the cycle is defined by using the up and down arrows.

2. **Visit or Cycle**

   Select the Cycle button at the top of the Visit or Cycle Definition panel.

3. **Cycle Type**

   Select the appropriate cycle type from the option list below the Cycle button. Each cycle type has special meaning and behavior. The one letter acronym 'C' will appear in the 1st column of the visit map table to identify cycle start records. The same acronym appears for all cycle types.

4. **Cycle Number**

   Enter the cycle number in the 2nd column of the table. Cycle number 0 must be used for all screening visits, cycle 1 for the first in-study cycle, and the other cycles must follow in sequential order.

5. **Cycle Acronym**

   This field is not used and cannot be defined.

6. **Cycle Label**

   A cycle label can be entered in the 4th column of the table.

7. **Cycle Scheduled**

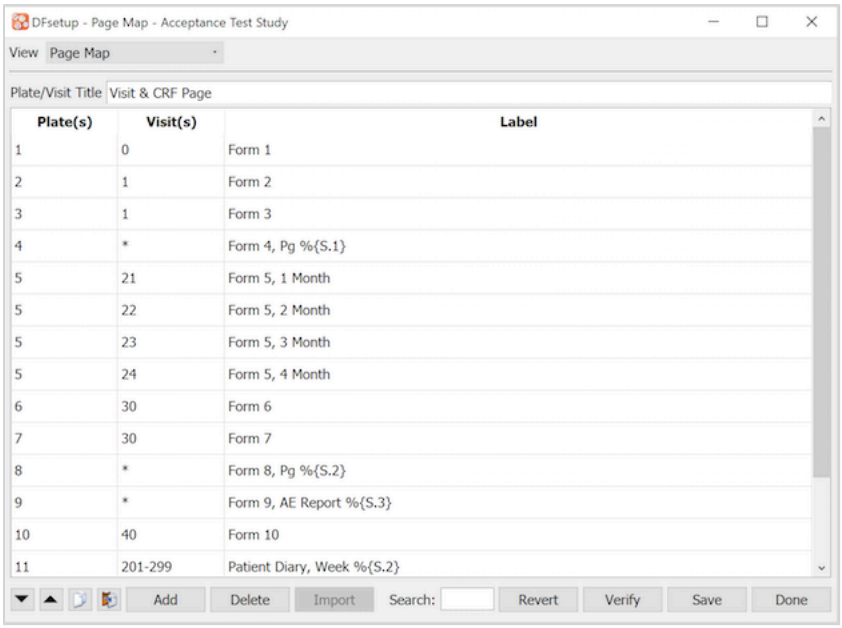
   Check this box if the cycle is scheduled and enter the scheduling method and overdue allowance. Cycles may be scheduled from a preceding visit, from the baseline of the first cycle or preceding cycle, from the visit date on which some condition occurred, or as shown in the example from the termination of the preceding cycle.

Verifying your work is highly recommended. The verification report checks the current version of the visit map and all of the conditional maps (see below) stored on the DFdiscover server; thus, Save changes before clicking Verify.

7.15. Page Map

Data queries are identified in **DFexplore** and Query Reports by: subject ID, visit number, plate number and field description. While subject ID and field description are always meaningful, in many studies the visit number and plate number are meaningless to the clinical sites and should be replaced by a more descriptive label. This is the purpose of the page map.

In addition to replacing visit and plate numbers in Query Reports, page map labels also appear in the **DFexplore** Queries and Data views where they replace the plate label in the queries list and in the subject binder navigation list respectively.



When creating Query Reports using **DF_QCreports**, if a page matches more than 1 entry in this table the first entry with the following specifications is used, where # stands for an explicit numeric value

- plate #, visit #

- plate #, visit *
- plate *, visit #
- plate *, visit *

In **DFexplore** the order of entries in the table is important. If a page matches more than 1 entry the first entry wins.

Page map specifications are optional. Any page which does not have a page map entry will be identified by its assessment and plate numbers.

The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

The Plate/Visit Title is optional, and is only relevant in the short version of Query Reports, where it appears as a title over the page map labels on the correction section of each report. If specified it should be no more than 17 characters long. If not specified the default title is 'PLATE SEQNO'

The plate and visit entries may contain a single value or range, a comma-delimited list of values and ranges, or an asterisk (*) to represent all possible values.

In the long version of Query Reports created by **DF_QCreports** page map labels have a maximum length of 32 characters. This is achieved when variable labels are set to their maximum length (40 characters) in **Study** > **Global Settings**. More compact Query Reports can be created by setting variable labels to 25 characters in **Study** > **Global Settings**, in which case page map labels are then limited to 17 characters. Page map labels will be truncated in Query Reports if they exceed these limits.

The above limits do not apply in **DFexplore** where page map labels may be more than 100 characters long, although practically they should be kept smaller.

The label entry may include any of the following substitutions. Remember to account for the maximum length of substituted values when computing final label lengths.

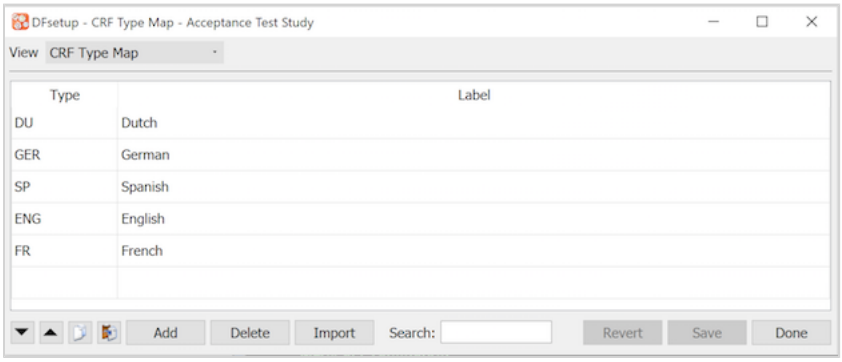
- **%P** is replaced by the plate number without zero padding.
- **%S** is replaced by the visit number without zero padding.
- **%{P.n}** is replaced by the last n digits of the 3-digit zero padded plate number.
- **%{n.P}** is replaced by the first n digits of the 3-digit zero padded plate number.
- **%{S.n}** is replaced by the last n digits of the zero padded visit number.
- **%{n.S}** is replaced by the first n digits of the zero padded visit number.
- **%{S.a.b}** is replaced by b characters of the zero padded visit number starting from character position a.
- **%n** is replaced by the value of field number n from the relevant data record. Only one field value substitution is allowed per label.
- **%n:d** this notation is for check and choice fields; field n is decoded and its label is used. Only one of these substitutions is allowed per label.

If the field used by Page Map is set to be Hidden or Masked in **DFsetup**, the label for the plate will be substituted with XXX.

If the field used by Page Map is set to be Hidden or Masked using `dfaccess` edit check function, the label with data value will still be displayed.

7.16. CRF Type Map

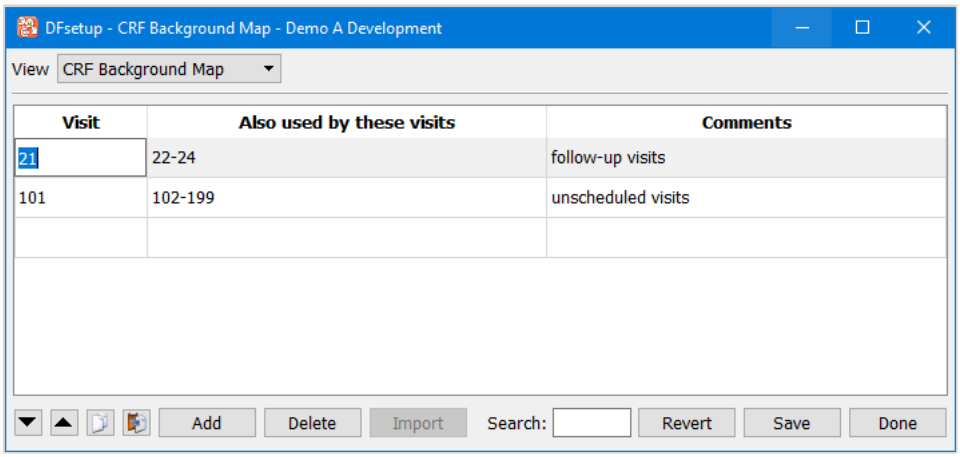
This dialog is used to create CRF types so that CRFs can be tagged by categorization (e.g. language). The CRF types must be defined in this map in order to be selected when importing CRF backgrounds of different types (refer to Import CRFs for details) and to be available as selection options in **DFexplore** Preferences (refer to *DFexplore User Guide, User Settings*) and in the `dfpref` edit check function (refer to *Programmer Guide, Edit checks*).



- Each CRF category must have a specified type name (max 8 characters) and label (max 20 characters).

7.17. CRF Background Map

When CRFs are imported, they may be tagged with a specific visit number (refer to Import CRFs for details). If a CRF tagged with a visit number is also used at other visits, use this table to link it to the additional visits.



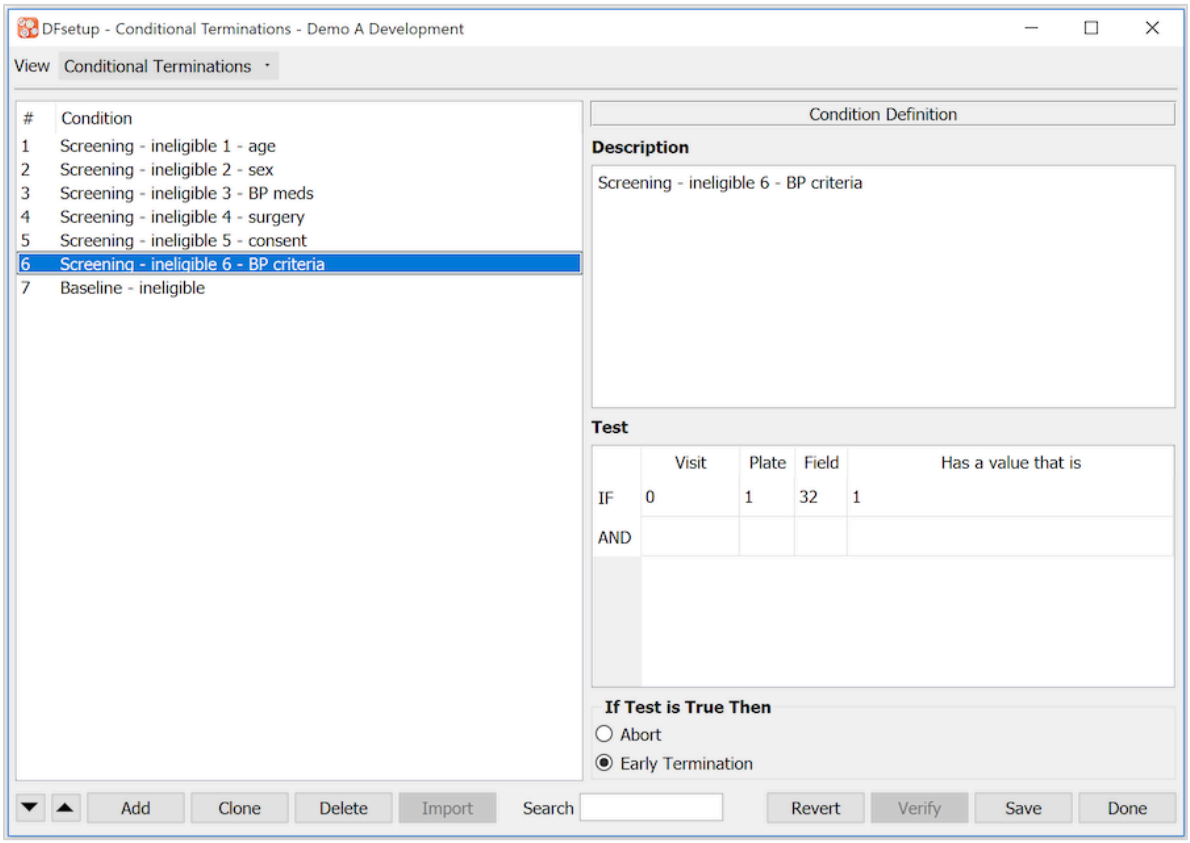
7.18. Conditional Terminations

The Conditional Terminations dialog is used to specify database conditions that terminate subject follow-up.

An early termination condition ends subject follow-up for the cycle that contains the visit (specified in the IF statement) that triggered the condition. An abort condition terminates all follow-ups for all cycles.

When a termination condition is met the VisitDate of the visit (specified in the IF statement) at which the condition was met becomes the termination date. When more than one termination condition is met the earliest VisitDate becomes the termination date. DFdiscover will not schedule visits beyond the termination date nor call such visits overdue.

This section shows how to use **DFsetup** to enter conditional termination specifications. For a detailed description of this and other visit scheduling features see Subject Visit Scheduling.



This example illustrates:

- Description:

  -for each condition the first line of the description is displayed in the Condition List.

- Test:

  -this test is true if field 32 on plate 1 at visit 0 equals 1.

- If Test is True Then:

  -subject follow-up terminates for the cycle containing visit 0

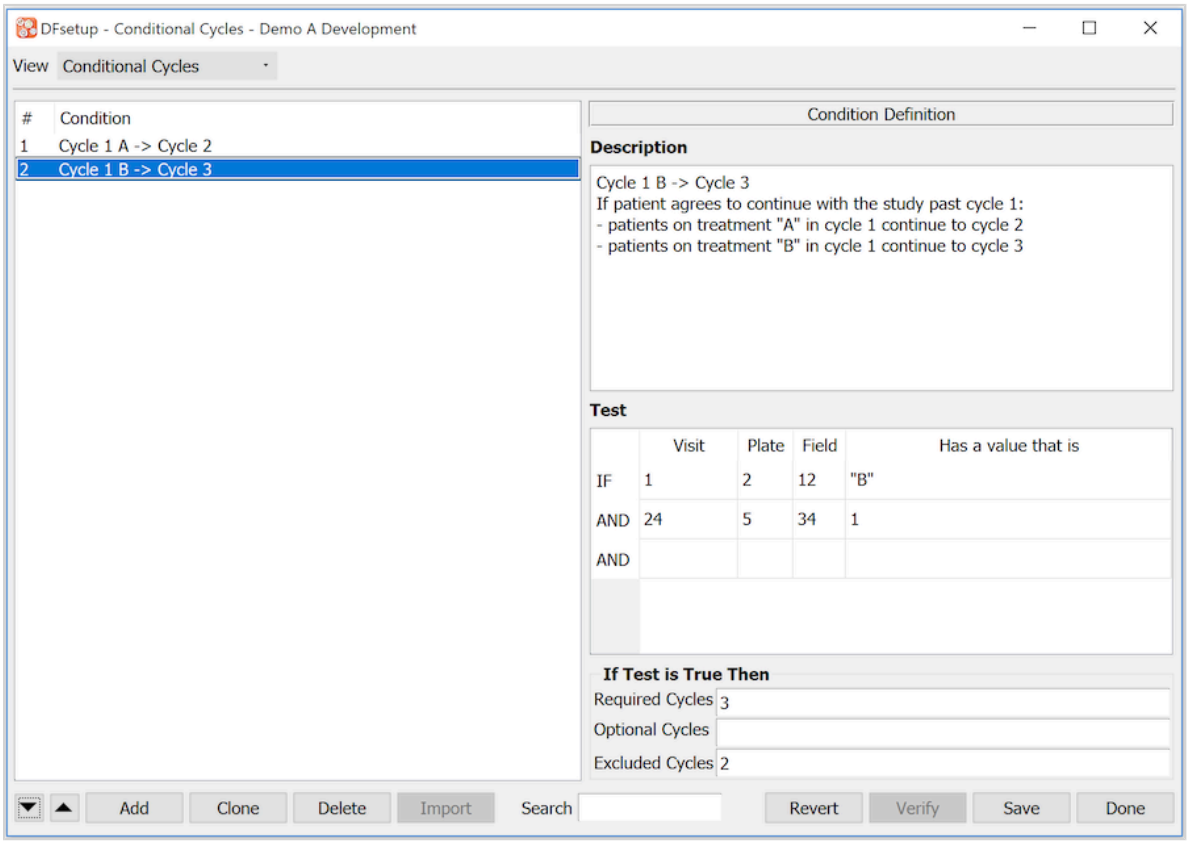
The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

All of the conditional maps perform database tests in the same way. The capabilities are described in the Conditional Tests section.

7.19. Conditional Cycles

The Conditional Cycles dialog is used to specify database conditions which can override the visit map specifications and make any cycle required, optional or excluded.

This section shows how to use **DFsetup** to enter conditional cycle specifications. For a detailed description of this and other visit scheduling features see Subject Visit Scheduling.



This example illustrates how conditional subject scheduling can be implemented when multiple visit cycles have been specified in the visit map.

- If the subject is in treatment group "B" (field 12 on plate 2 at visit 1 = "B")
- And the subject agrees to continue in the trial at the end of cycle 1 (field 34 on plate 5 at visit 24 = 1)
- Then the subject proceeds to the first visit in cycle 3, and is not expected to complete any of the visits in cycle 2.
- The date on which cycle 3 is expected to start is determined by the cycle scheduling method specified in the visit map.
- A comma-delimited list of cycle numbers and ranges can be specified if a condition effects more than one cycle.

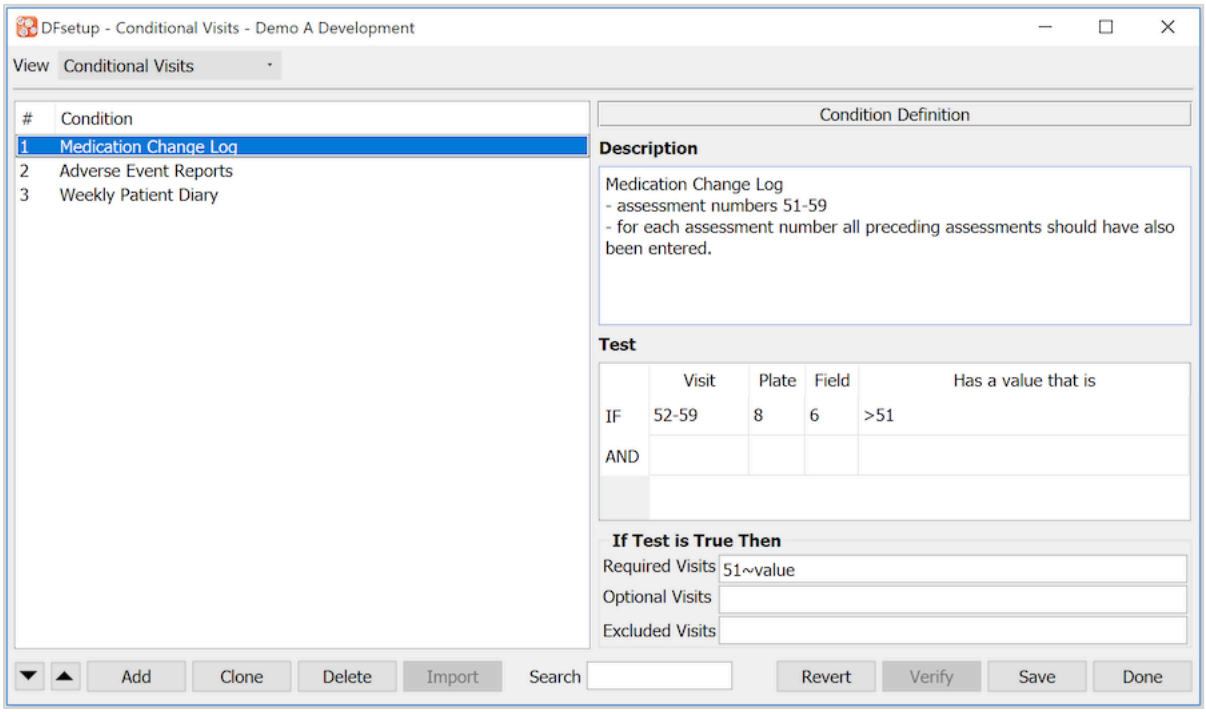
The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

All of the conditional maps perform database tests in the same way. The capabilities are described in the Conditional Tests section.

7.20. Conditional Visits

The Conditional Visits dialog is used to specify database conditions which can override the visit map specifications and make any visit required, optional or excluded.

This section shows how to use **DFsetup** to enter conditional visit specifications. For a detailed description of this and other visit scheduling features see Subject Visit Scheduling.



This example illustrates how the conditional visit map can be used to check for gaps in a sequential assessment number series.

- Medication log pages use assessment numbers 51-59 in order.
- If a medication log page is entered with a log number greater than 51 (field 6 on plate 8 at visit 52-59 is > 51)
- Then all lower numbered assessments in the series should also be in the database (required visits 51~value),
- where value is the value of the field specified in the IF statement that met the condition (i.e. field 6 the assessment number).

If a condition makes a visit required the scheduling specifications in the visit map (if any) determine when the visit is due and overdue.

The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

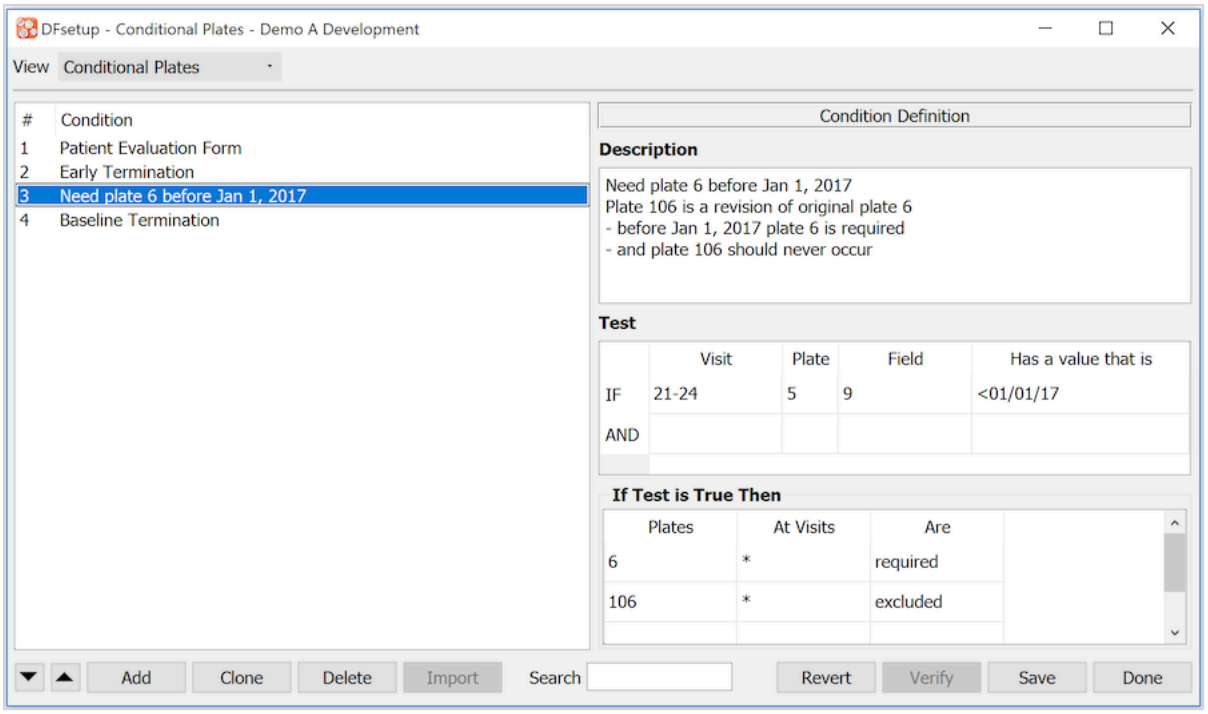
All of the conditional maps perform database tests in the same way. The capabilities are described in the Conditional Tests section.

7.21. Conditional Plates

The Conditional Plates dialog is used to specify database conditions which can override the visit map specifications and make any plate required, optional or excluded.

These conditions only take effect once the visit has occurred and at least one plate for the visit has been entered into the study database. If you need to specify a condition for a visit that has only one plate use the conditional visit map not the conditional plate map.

This section shows how to use **DFsetup** to enter conditional plate specifications. For a detailed description of this and other visit scheduling features see Subject Visit Scheduling.



This example illustrates how the conditional plate map can be used for CRF plate version control.

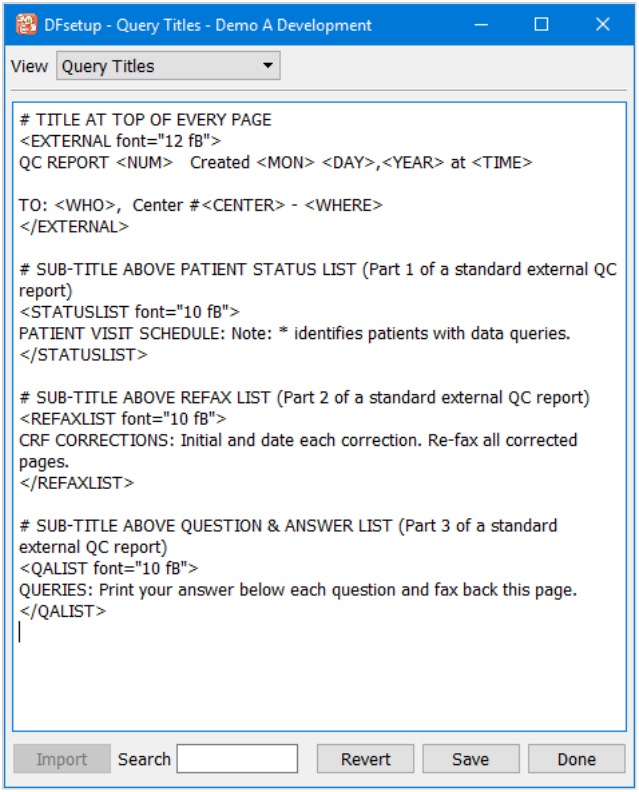
- The study started with plate 6 as one of the required plates at each follow-up. During the trial this plate was revised to create a new version, which was given plate number 106.
- The new version came into effect on Jan 1, 2017.
- 2 conditions are required for version control, one of which is shown here.
- For follow-up visits performed before Jan 1, 2017 (field 9 on plate 5 at visit 21-24 is <01/01/17)
  - plate 6 is required
  - plate 106 is not expected
- The Subject Scheduling > Subject Unexpected report (or the legacy DF_PTunexpected report) can be used to list any unexpected plates or visits that have been entered into the database.

The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

All of the conditional maps perform database tests in the same way. The capabilities are described in the Conditional Tests section.

7.22. Query Titles

The title at the top of each page and above each of the 3 sections of the Query Reports created by **DF_QCreports** can be customized as illustrated in the following example.



Query Titles can have any or all of the following parts:

- EXTERNAL - page title for external Query Reports
- INTERNAL - page title for internal Query Reports
- STATUSLIST - sub-title above subject status list
- REFAXLIST - sub-title above refax/correction queries
- QALIST - sub-title above clarification queries

The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

The following rules must be followed when defining Query Titles.

- Title specifications must be formatted exactly as illustrated in the example.
- The opening and closing tags for each section must appear on new lines by themselves with no leading or trailing space or text.
- No space is allowed within the opening and closing tag except before the word **font**.
- Anything entered outside of a tagged block is ignored.
- A # character may be used to indicate a comment line but # is not strictly needed, and has no special meaning inside a tagged block.
- The font value must be enclosed in double quotes. Limited font specifications may be included within the opening tag. The leading number sets the point size (10 or 12 are recommended). Available font types include:

fB for bold
fH for normal helvetica
fCW for constant width
The font specification is optional. Without it, **font="10 fCW"** is used for Query Report titles and **font="10 fB"** is used for the sub-titles for the three sections of the report.

The following variables may be included in the Query Report titles which appear at the top of each page, but not in the sub-titles above the subject status, refax and question & answer sections. The study name comes from the study configuration file maintained by **DFadmin**, clinical site information comes from the sites database, and date/time come from the system clock on the DFdiscover server.

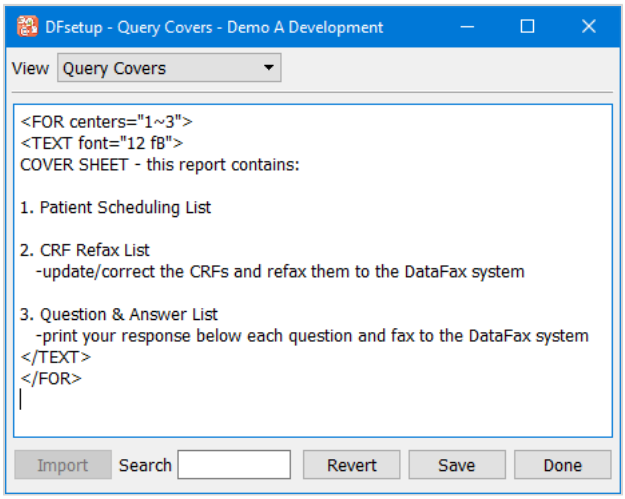
Table 7.3. Variables available for use in QCcovers, QCmessages, and QCtitles

Variable	Use in External	Use in Internal	Meaning
<STUDYNAME>	yes	yes	study name
<SITE> [a]	yes	no	site ID
<WHO>	yes	no	primary site contact
<WHERE>	yes	no	site affiliation
<MAIL>	yes	no	site mailing address
<PRIMEFAX> <FAX1>	yes	no	primary fax number (to which Query Reports are sent)
<FAX2>	yes	no	secondary fax number
<PHONE1>	yes	no	primary contact's phone number
<PI>	yes	no	principal investigator
<PHONE2>	yes	no	principal investigator's phone number
<REPLYTO>	yes	no	email address to which replies made to emailed Query Reports will be sent
<NUM>	yes	no	external report name composed of site ID, date (yymmdd), and page, e.g. 025-080115-01.
<NAME>	yes	yes	external report name composed of site ID and date only, e.g. 025-080115
<PAGE>	yes	yes	page number of Query Report
<DAY>	yes	yes	two-digit day of month
<MON>	yes	yes	three-character month of year
<YEAR>	yes	yes	four-digit year
<WKDAY>	yes	yes	three-character day of week
<TIME>	yes	yes	time of day (hh:mm:ss AM/PM), e.g. 01:12:22 PM
<DATE>	yes	yes	date (ddd mmm dd hh:mm:ss yyyy), e.g. Tue Jan 15 14:34:07 2018

[a]The variable <CENTER> is also supported for backwards compatibility

7.23. Query Covers

This dialog is used to specify cover sheets for external Query Reports as illustrated below.



- This example shows a simple cover sheet which will be used for external Query Reports created for sites 1-3.

The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

Each cover sheet begins with a **<FOR center="#list">** [2] tag which identifies the site(s) that will receive the cover sheet and ends with a **</FOR>** tag. The site number list is a comma-separated list of numbers and ranges.

The body of each cover sheet is defined in a **<TEXT>** block, nested within the FOR block. Variable substitutions and font specifications may be performed as previously described for Query Titles. Aside from variable substitution, all other text appears exactly as formatted within the TEXT block. Blank lines used to double space text will be preserved. The default font for cover sheet text is 10 point constant width.

It is possible for some sites to receive cover sheets while others do not, and for different sites to receive different cover sheets. Different cover sheets are created for different sites by adding a **<FOR>** block for each cover sheet.

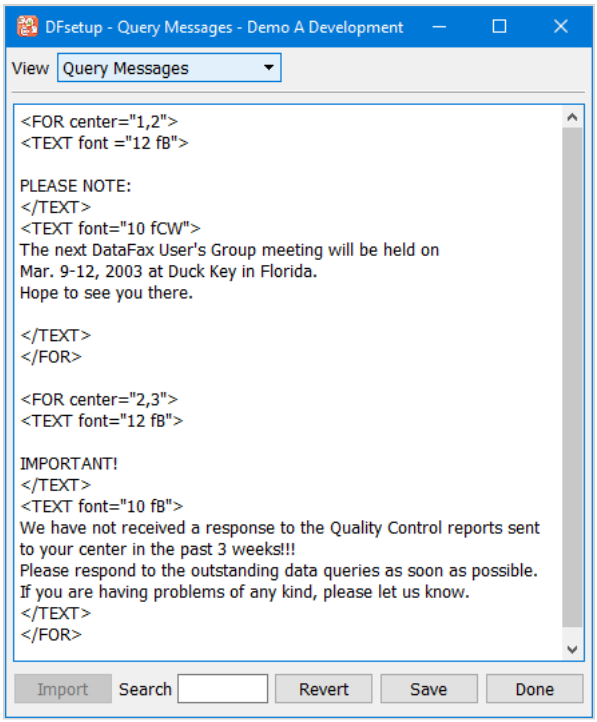
If a site number is included in the **<FOR>** block of more than one cover sheet the **<TEXT>** blocks are concatenated to form the Query Report cover sheet for that site.

Cover sheets can be modified at any time. The last set of specifications saved to the DFdiscover server will be used when external Query Reports are created by **DF_QCreports**.

7.24. Query Messages

The Query Messages dialog is used to add messages to external Query Report cover sheets for specified sites, as illustrated below.

[2] For backwards compatibility, **center** is still required in this context but the meaning of the value is the same as site.



- This example includes 2 messages, one for sites 1 and 2, and another for sites 2 and 3.
- Site 2 will receive both messages. Note the blank line at the end of the 1st message. This will separate the messages on the cover sheet.

The control buttons at the bottom of the window are the same for all study configuration dialogs. See General Dialog Controls for a description.

Query Messages have the same structure as Query Covers, consisting of a **<TEXT>** block nested within a **<FOR>** block, and support the same font and variable substitutions.

Any messages directed to a site are appended to the site's cover sheet. If no cover sheet has been defined for the site, the messages will be added to a blank cover sheet. If multiple cover sheets and multiple messages have been directed to the same site, the cover sheets are first concatenated followed by all of the messages, in the order in which they are defined in Query covers and Query Messages.

While messages could be added to the **<TEXT>** blocks defined within Query Covers, the use of a separate study configuration file for messages allows you to keep the static header in cover sheets separate from messages, whose content and relevance to different sites will likely change quite often during the course of the study.

Chapter 8. Subject Visit Scheduling

8.1. Introduction

Visit scheduling specifications are used to:

- schedule subject follow-up assessments
- identify overdue visits and missing CRF pages
- identify visits and CRF pages that are present but unexpected
- create subject and site visit and CRF tracking reports
- create Quality Control reports for the clinical sites containing subject scheduling information and outstanding data queries for overdue visits and missing pages.

The visit scheduling components are listed below.

- **Visit Dates.** Which data fields used to record the date of subject assessments and any termination events?
- **Visit Map.** What visits make up the full set of possibilities for all study subjects? What visit numbers are used to identify each visit? What is the chronological ordering of visits? Are visits grouped in cycles where each cycle starts with a new baseline visit? How much overdue allowance is permitted before an overdue visit query is created? What CRF plates are required and optional for each visit?
- **Conditional Cycle Map.** Are there database conditions that are used to decide whether some cycles are required, optional or unexpected for each subject?
- **Conditional Visit Map.** Are there database conditions that are used to decide whether some visits are required, optional or unexpected for each subject?
- **Conditional Plate Map.** Are there database conditions that are used to decide whether some CRF plates are required, optional or unexpected at specific visits?
- **Conditional Termination Map.** Are there database conditions that indicate that subject follow-up terminates at certain visits? Do such terminations stop all further follow-up, or just follow-up within the current cycle?
- **Early Termination Plates.** Does the study include one or more special CRFs that are submitted with an assessment to indicate that follow-up ends, for the current cycle, as of that assessment?
- **Missed Visit Plates.** Does the study include one or more special CRFs that are submitted by the clinical sites for each visit that was missed, so that DFdiscover will know not to send overdue visit queries?

Visit dates are identified by the special `Visit Date` attribute, and each of the other components listed above is specified using a dialog invoked from **DFsetup View**. A detailed description of each of these components follows.

8.2. Visit Dates

Visit date fields are used to record the day on which a visit occurred or subject follow-up was terminated. They are used for subject scheduling and must be specified as follows:

- All visit date fields must use the same date format and be defined using the special `Visit Date` attribute.
- A visit date field must appear on at least one CRF page for each baseline and scheduled follow-up visit. They may also appear on optional visits but are only required if the optional visit may become required when some condition is met as specified in the conditional visit map.

- A visit date field must exist for any event that terminates subject follow-up, e.g. final visit date, early termination date, date of death, etc. These dates are used to halt visit scheduling so that visits that would have been due after the termination date are not identified as overdue.
- The `Visit Date` attribute must not be used for unscheduled events like the onset dates of an adverse event.
- Each CRF page can have only one visit/assessment number (field 6) and thus can also have only one `Visit Date` field identifying when that visit occurred.

If you assign the `Visit Date` attribute to dates on more than one plate for the same visit make sure that you expect the same calendar date to be recorded in each of these fields. If DFdiscover finds inconsistencies between dates using the `Visit Date` attribute for the same visit, it will report the problem (when **DF_XXkeys** is run) and will generate a DFdiscover retrieval file that can be used to review the inconsistent cases.

While it is illegal to have different visit dates recorded for the same visit, it is legal to have the same visit date recorded for different visits. However, within the ordered list of study visits, defined within screening and in-study cycles in the study visit map, it is illegal for visit dates to be out of chronological order. Thus the same visit date should only appear, if at all, for adjacent visits.

8.3. Visit Map

The visit map is the key component of the subject scheduling specifications. This is where study visits are defined, grouped and ordered. Within every visit map there are 2 levels of organization, cycles which contain a set of related visits, and the visits themselves. In this section we'll first describe cycles and then visits.

8.4. Cycles

Cycles come in 3 flavors: screening cycles, in-study cycles and end cycles, and must appear in the visit map in that order. The screening and end cycles are optional, and no more than one of each may be defined in the visit map. In-study cycles contain most of the visits for a typical trial. Multiple in-study cycles may be defined, with some being required, optional and conditional and with different in-study cycles being used for different groups of subjects with different scheduling requirements, or to re-set the baseline for different phases within the same trial.

All cycles must be numbered consecutively within the visit map. The screening cycle must be defined as cycle 0, in-study cycles are must be numbered consecutively from 1, and the end cycle gets the next consecutive integer following the last in-study cycle. Each cycle can also be given a short descriptive label, which is used by some of the standard reports. Each of the 3 cycle types has different uses and follows different rules. We'll consider them in order.

8.4.1. The Screening Cycle

If used, the screening cycle must appear at the beginning of the visit map as cycle number 0. It may contain visit types X (screening) and E (early termination) only; no other visit types are allowed. The screening cycle will not have a scheduled start date because nothing precedes it.

Within a screening cycle all screening visits (X) are required, unless they are made optional or excluded by the conditional visit map. Each screening visit can be unscheduled, or scheduled relative to the first screening visit that serves as the baseline for the cycle. The final screening visit serves as the termination of the screening cycle, unless it is made optional or unexpected by the conditional visit map, in which case the last required or completed screening visit is used as the termination visit of the screening cycle.

The screening cycle can be brought to an early conclusion by defining one or more early termination visits (E) within the screening cycle, by designing an early termination plate that can be submitted at the appropriate screening visit, or by defining termination conditions for the screening cycle in the conditional termination map.

Although unusual, it is legal for a visit map to contain nothing but a screening cycle. This might be useful for a trial consisting of an ordered list of subject assessments, where each subject was allowed to complete the assessments at their own pace, with no specified scheduling targets. This could be done using a visit map having nothing but a screening cycle containing a list of unscheduled screening visits.

8.4.2. In-Study Cycles

In-study cycles sit between the screening cycle and end cycle, and are the most important part of the visit map. These are the cycles that contain the visits that record subject treatment and follow-up. A visit map may contain 1 or more in-study cycles, to accommodate different treatment and follow-up requirements for different subjects, or because treatment and/or follow-up is divided into different phases each with its own baseline visit for within-cycle visit scheduling.

In-study cycles may be defined as required, optional or conditional. A conditional cycle is unexpected unless triggered by a condition specified in the conditional cycle map (see Section 8.8, “Conditional Maps”). Required in-study cycles are required of all subjects unless the cycle is changed to optional or unexpected by the conditional cycle map. Optional in-study cycles may be completed or not at the discretion of the clinical site, unless the cycle is changed to required or unexpected by the conditional cycle map. Optional cycles become required once they start, i.e. visit scheduling and visit requirements become active within an optional cycle when one or more visits have been received.

A scheduling method should be specified for all in-study cycles, regardless of type. This is used to calculate the expected start date for required cycles, as well as for optional and conditional cycles (should they become required as a result of a conditional cycle map trigger). In-study cycles can be scheduled to begin a specified number of days following: the termination of the previous cycle, the baseline of the previous cycle, the baseline of the first in-study cycle, a specified visit number, or in the case of a cycle triggered by the conditional cycle map, the visit on which the condition was met. You will also be asked to specify an overdue allowance. When this time expires DFdiscover will generate an overdue visit query for the first required visit in the cycle if it has not yet arrived.

It is legal for any in-study cycle to consist of a single visit. If this is the case, the visit must be defined using the B (baseline) visit type, which then becomes both the baseline and termination visit for the cycle. This feature makes it possible to schedule visits relative to any other preceding visit in the visit map, rather than having all visits scheduled from a common baseline, as is the clinical trial norm. Such dynamic scheduling can be accomplished by scheduling each single visit cycle using any of the legal cycle scheduling methods described in Section 8.4.4, “Cycle Specifications”.

Most in-study cycles will consist of several visits, in which case they must include at least one baseline and one termination visit. A typical list of visit types for an in-study cycle would include:

- P - one or more pre-baseline visits
- B - a single baseline visit
- S - one or more scheduled follow-up visits
- T - a single planned termination visit
- O - one or more optional visits
- R - one or more assessments required on cycle termination

For a description of the visit types, what they mean and the rules that apply to each of them skip ahead to Section 8.5.2, “Visit Type”.

8.4.3. The End Cycle

The end cycle is so named simply because it comes at the end of the visit map. Its purpose is to serve as a container for any unscheduled visits that do not belong to a screening or in-study cycle. Because the end cycle contains no scheduled

visits it does not have a scheduling method or start date. The end cycle may contain 3 visit types: O (optional), R (required on termination), and A (abort).

This is where you should define most event reports, e.g. adverse events, outcome events, hospitalizations, change of medication, problem reports, etc. Such assessments are defined as type O visits (optional), but they can become required during the study, triggered by data collected at other visits, as we'll see below when we discuss the conditional visit map (see Section 8.8.2, “Conditional Visit Map”).

The end cycle can also contain type R visits (required on termination), used to collect CRFs that become due when all subject follow-up ends. This might be used for subject diary forms, or any other assessment that is not required until the subject has completed the study.

Type A visits (abort), which terminate all follow-ups for all cycles, should be placed in the end cycle.

8.4.4. Cycle Specifications

Each screening, in-study and end cycle begins with a cycle definition record containing 7 fields, which can be entered into the visit map using the **DFsetup** Visit map view.

1. **Cycle Number.**

   Cycles must be numbered sequentially, starting with 0 for the screening cycle, if present, and starting at 1 for the first in-study cycle.

2. **Cycle C.**

   The second field must contain the capital letter C. This distinguishes cycle records from visit records in the visit map.

3. **Cycle Label.**

   This field contains a descriptive label for the cycle. As with visit labels, the maximum length is 32 characters. The cycle label is used by some of the reports including **DF_SSvisitmap** and **DF_PTvisits**.

4. **Cycle Type.**

   This field contains a single letter that identifies the cycle type.

   - S - screening cycle
   - C - conditional in-study cycle
   - O - optional in-study cycle
   - R - required in-study cycle
   - E - end cycle

   Screening cycles are required, if present, unless made optional or unexpected by the conditional cycle map. A required in-study cycle is required for all subjects in the study, unless it is changed to optional or unexpected by the conditional cycle map. An optional in-study cycle may be completed or not, unless changed to required or unexpected by the conditional cycle map. Optional cycles are automatically changed to required on the arrival of any of the cycle visits. A conditional in-study cycle is unexpected unless changed to required or optional by the conditional cycle map. The end cycle has no requirement itself and cannot be acted upon by the conditional cycle map, because it is just a container for unscheduled visits that do not belong to any particular cycle. However, visits within the end cycle may become required through the conditional visit map. For example, a condition set on an adverse event report number recorded on a follow-up form could be used to trigger a requirement for the corresponding adverse event assessment report.

5. **Cycle Due Day.**

   The cycle due day identifies when the first required visit in the cycle becomes due. It is interpreted using the cycle scheduling method specified in the 7th and last field of the cycle definition record. This field is only relevant for in-study cycles.

6. **Cycle Overdue Allowance.**

   This field is only relevant for in-study cycles. Required cycles are called overdue when the scheduled due date plus overdue allowance lies in the past.

7. **Cycle Scheduling Method.**

   This field is only relevant for in-study cycles. Legal cycle scheduling methods include:

   - visit# - from a specified preceding visit
   - S - from the baseline visit of 1st required or completed cycle
   - C - from the visit specified in the conditional cycle map IF statement that made the cycle required
   - B - from the baseline visit of the last required or completed cycle
   - T - from the termination visit of the last required or completed cycle
   - N - not scheduled

   In many trials all cycle scheduling is relative to the same initial baseline visit. In this case the visit# method should be used. However, if the first in-study cycle is different for different subjects, the S method can be used to schedule all follow-ups relative to the baseline visit of the first in-study cycle completed for each subject.

   In those cases where a cycle is conditional upon the occurrence of some event it will often be a requirement that the conditional cycle be scheduled to begin a specified number of days after the occurrence of that event. The C scheduling method was designed for this purpose.

   The B and T methods were designed for scheduling relative to the last cycle. When scheduling the next cycle using the T method before the current cycle has terminated, the expected termination date of the current cycle is used. This may of course turn out to be wrong, but **DF_QCupdate** will correct the scheduling once the actual termination date becomes known.

   This approach, of scheduling cycles and visits from an expected date when the actual date needed for scheduling is unknown, is widely used by **DF_QCupdate**. It is in fact the only way that a complete schedule can be generated in the absence of the required dates. In addition to required cycles, **DF_QCupdate** also schedules optional cycles, even though they may end up being skipped. **DF_QCupdate** does not schedule any cycle or visit that is currently considered unexpected. This includes conditional cycles that have not yet been set to 'required' by the conditional cycle map.

8.5. Visits

The visits that comprise each cycle are entered into the visit map using the **DFsetup** Visit map view. In this section we'll explain each of the attributes required to complete a visit specification. These include:

Visit Number an identification number in the range 0-65535

Visit Type 1 of 12 types represented by the acronyms: XPBOSTWFEARr

Visit Label a descriptive label (32 char max) used in overdue visit queries

Due Day	# of days following baseline when the visit becomes due
Overdue Allowance	# of days past due day when the visit becomes overdue
Visit Date	location of the visit date field in the database
CRF Plates	list of required and optional plates, and the missed visit notification plate
Display Order	list of required and optional plates in the order that they are to appear in the **DFexplore** subject binders.

8.5.1. Visit Number

Each CRF page in a DFdiscover study is uniquely identified by 3 keys: a CRF plate number, a visit number and the subject ID.

Visit numbers (also referred to as assessment and sequence numbers) serve 2 purposes. For scheduled and optional visits they distinguish among different visit dates or times at which subjects were seen in the clinic, lab, home, etc. and for numbered event reports they serve as the report number, to distinguish, for example, one adverse event report from all others.

Each visit number must be an integer in the range 0-65535. Those less than 511 can appear in the barcode; otherwise, they must appear as the first data field at the top of each CRF page. If the visit number is greater than or equal to 511, and is fixed for the CRF page on which it appears, it can be pre-printed in the first data field so that it need not be entered by those recording subject data on the CRFs.

Although visit numbers are typically chosen sequentially this is not a requirement. Gaps can appear in the visit number list, and visit numbering need not correspond to the chronological order in which visits occur, as this is determined by the order in which they are listed in the visit map. Visit numbers can simply be viewed as an identification number that distinguishes one subject assessment from another.

Within each cycle the expected chronological order of the visits corresponds to the order in which they are listed in the visit map. Thus, for example, if a study uses visit numbers 1-50, and you discover that you need to add a new visit between visits 11 and 12, you could chose a number like 115, insert it in the visit map between visits 11 and 12, and specify an entry in `DFpage_map` to display it as 11.5 on quality control reports. This is the only way that visit numbers can appear to have decimal values. At the DFdiscover database level all visit numbers must be integers, but the CRFs can be designed to show decimals and the page map can convert the relevant integers to decimal values for the Query Reports sent to the clinical sites.

The selection of visit numbers should be done with care. They can be arbitrary but ideally they should be meaningful. For example, in a study consisting of up to 9 cycles, each with up to 99 visits, and with the possibility that each visit might need to be repeated, but no more than 9 times, visit numbers might be designed such that the first digit is the cycle number, the next 2 digits identify the visit number within the cycle, and the last digit is 0 for the first occurrence of the visit and 1-9 for repetitions. Thus, for example, visit number 4132 could be designed to represent the second repetition of visit 13 in cycle number 4.

For numbered reports, e.g. adverse event reports, hospitalization reports, medication change reports, etc., the visit number is more commonly called a sequence number. Since all visit/sequence numbers must be unique you cannot use the same value for both a visit and a sequence number. The recommended solution is to use a prefix number plus the report number to form sequence numbers that are different from any of the visit numbers used in the study. For example, to allow a maximum of 999 adverse event reports for each subject, in a study where all visit numbers were in the range 0-4999, we might use 5 + the adverse event report number to form sequence numbers in the range 5001-5999 for each adverse event report.

To make it easier for those completing the CRFs the visit number field can be displayed in separate parts on the CRF and combined by DFdiscover to compose the visit number key. In the first example described above the CRFs completed during the screening cycle might appear on the CRFs as:

```
Screening Visit #[ ][ ] repetition #[ ]
```

The cycle number, zero in this case, is not required. For visits completed during the in-study cycles the CRFs might read:

```
Phase #[ ] Visit #[ ][ ] repetition #[ ]
```

And for the adverse event forms the assessment number could look like this:

```
Adverse Event #[ ][ ][ ]
```

The prefix number 5, needed to put adverse event sequence numbers in the 5000 series, in the example above, can be specified as a constant when defining the sequence number for the adverse event plate(s) in **DFsetup**.

To make sure that investigators will be able to locate the specific CRF page related to each data query, the study page map (Page Map) should be used to convert visit and sequence numbers into meaningful labels. This becomes particularly important when visit numbers are comprised of several parts and sequence numbers use a fixed prefix, as in the above examples.

The visit map can accept a single visit number, and a range or range list of visit numbers in a single visit map entry. This greatly simplifies the specification of various types of optional assessments, such as adverse event reports, hospitalizations, and interim visits. In some cases, the distinction may need to be made between visit ranges in which visit numbers must be used in order (e.g. adverse events), and visit ranges in which visit numbering may include gaps. Two types of range syntax accommodate these behaviors:

1. `#-#` : delimiter is a dash, for visit numbers that must be used in order

2. `#~#` : delimiter is a tilde, for visit numbers that may include gaps

Visits specified with the `#-#` syntax will appear in order in the **DFexplore** record list. Visits with #~# will not appear in the **DFexplore** record list until they are manually added using the **Visit** > **Add New Visits** action.

> **Note**
>
> It is invalid to mix the `#-#` and #~# syntax within the same visit entry. If a range list contains both `-` and ~, the ~ is treated as a `-`.

8.5.2. Visit Type

DFdiscover recognizes 12 different visit types each denoted by a single letter key, one of: XPBOSTWFEARr. The rules governing the usage of each visit type are described below.

X: Screening Visit

Screening visits may only appear in the Screening cycle. All screening visits are required, unless changed to optional or excluded by the conditional visit map. Each screening visit may be scheduled or unscheduled. The first required screening visit serves as the baseline for scheduling within the Screening cycle and the last required screening visit serves as the cycle termination.

P: Scheduled Pre-Baseline Visit

One or more pre-baseline visits may be defined before the baseline visit in each in-study cycle. Pre-baseline visits are required and are scheduled relative to the cycle baseline visit (with a negative due day).

B: Baseline Visit

Baseline visits provide the initial date for all visit scheduling within each in-study cycle. Thus they must have a due day of zero. Each in-study cycle must have at least one baseline visit, but multiple baselines are allowed provided they appear together, following any pre-baseline visits (P) and preceding any scheduled follow-up visits (S). If multiple baseline visits are defined, the first one is required and all subsequent baseline visits are

optional, unless altered by the conditional visit map. For example, it is possible using the conditional visit map to require that the baseline be repeated if some condition is met. If more than one baseline visit is received in the same in-study cycle, the last one (per visit map order) is used for all within-cycle visit scheduling.

It is legal to define an in-study cycle with only one visit, provided that visit is defined as the cycle baseline. In such cases the baseline visit becomes both the baseline and termination visit for the cycle.

S: Scheduled Follow-up Visit

Scheduled follow-up visits are required visits scheduled a specified number of days following the cycle baseline (unless changed to optional or excluded by the conditional visit map). They can only be used in in-study cycles and must follow the cycle baseline visit and precede the cycle termination visit.

O: Optional Follow-up Visit

Optional visits are unscheduled visits. They may appear in in-study and end cycles. An overdue visit query can only be generated for an optional visit if it becomes required because of the conditional visit map. Optional visits do not have a scheduled due date in the visit map, and instead become required immediately if they become required through the conditional visit map; in effect their due date becomes the date of the visit on which the condition was met.

T: Cycle Termination Visit

The cycle termination visit type can only appear in in-study cycles. Each in-study cycle, that has a baseline visit and consists of more than this one visit, must have a termination visit scheduled a specified number of days following the baseline visit using visit type T, or occurring within a specified date window using visit type W. Only one T and W visit is allowed, but it is permitted to have both, in which case the T visit must precede the W visit. Arrival of either of these visits terminates the cycle in which they are defined, as of the date on which the visit occurred.

E: Early Termination Visit

Early termination visits may used in screening and in-study cycles (but not in the End cycle). These are optional assessments which terminate the cycle in which they arrive. As an example, an early termination assessment, containing 1 or more plates, might be completed when a treatment cycle is terminated early due to intolerable side effects.

A: Abort Visit

Abort visits terminate all follow-up for all cycles and should be placed in the end cycle. Examples include a death report or a subject discontinuation report.

r: Required By Time Of Next Visit

This visit type can only be used in in-study cycles. Use this visit type for those assessments which are linked to, and required upon arrival of, the next scheduled visit in the cycle. Type r visits become overdue when the next scheduled visit arrives or if that visit is itself overdue. These visits are only linked to the next scheduled visit, not to any subsequent scheduled visits. Thus if the next scheduled visit becomes unexpected because of a conditional visit map or termination event, any type r visits that immediately precede it will also be considered unexpected.

R: Required By Time of Termination Visit

Type R visits can only be used in in-study and end cycles. When specified in an in-study cycle they become required on termination of the cycle, and when specified in an end cycle they become required on termination of all study follow-up.

If the scheduled due date for a type R visit is zero, it becomes required immediately upon termination provided follow-up has at least reached the baseline visit. If the specified due day is greater than zero, the type R visit is required on termination provided the due day precedes the termination date. This can be useful for assessments like subject diaries, to be collected on termination, where you only expect to collect diaries that were scheduled for completion prior to termination.

When defined within an in-study cycle the due day is calculated from the cycle baseline, and when defined in an end cycle the due day is calculated from the baseline of the first cycle that was used for the subject.

F: Final Visit

This visit type exists for historical reasons. Although not really necessary it is retained for backward compatibility. The final visit type, if used, must appear as the first visit in the last in-study cycle. Thus only one such visit can appear in a visit map. It serves as both the baseline and termination visit of the cycle. It also acts like an abort visit in that its arrival terminates all follow-up in all cycles.

The only visit types that may follow a final visit within the last in-study cycle are: R (required on termination of the cycle - i.e. immediately) and O (optional).

W: Study Termination Window Visit

This visit type has significantly different semantics from the other visit types. It serves as an alternative way of terminating an in-study cycle.

When all subjects are followed for a fixed duration (e.g. 3 years) the visit map can use a type T termination visit, and the amount of time required to close out the trial will equal the amount of time required to enroll all of the study subjects. Other studies end on a specified date, resulting in different amounts of follow-up for each subject. In such studies, we need a different way of scheduling the termination visit. DFdiscover provides the following methods. Note that in each case the termination date or dates must be specified using the study `Visit Date` attribute.

- **on a specific date.** For example, `on 2006/12/01`. This method schedules the type W follow-up visit on the same day, Dec 1, 2006, for all subjects.
- **before a specific date.** For example, `before 2006/12/01`. The last scheduled follow-up visit for each subject that occurs before Dec 1, 2006 is replaced by the type W termination visit.
- **after a specific date.** For example, `after 2006/12/01`. The first scheduled follow-up after Dec 1, 2006 is replaced by the type W termination visit.
- **between two specific dates.** For example, `between 2006/11/01~2006/11/30 .25`. All type W follow-up visits are to be scheduled in November 2006. The scaling factor (.25 in the above example) spreads the visits out over the termination interval. It should be set to the termination window interval divided by the usual inter-visit interval. In the above example a scaling factor of .25 would be appropriate for the 1 month termination window interval if the usual interval between follow-up visits is 4 months.

The between method identifies the date of the first scheduled visit after the start of the interval and modifies this date using the scaling factor. For example, if the date of the next scheduled visit is 100 days after the start of the termination window and the scaling factor is .25, the date of the type W termination visit will be set to 25 days after the start of the termination window. If a scaling factor is too large, the final visit for some subjects may be beyond the end of the termination window. In such cases the final visit is scheduled for the final day of the termination window. For example, if the scaling factor is 1, those subjects whose next scheduled visit falls within the termination window will have the final visit scheduled on this date and all other subjects will have their final follow-up visit scheduled on the final day of the termination window.

Most trials will terminate with either a termination visit (T) or a termination window visit (W), but it is legal to include both. This allows the visit map to define some maximum

duration of subject follow-up using a termination visit (T), followed by a termination window visit (W) which brings the study to a close by scheduling a final visit for all subjects who have not yet reached the termination visit (T). A termination window visit (W) must be defined as the last scheduled visit in the cycle, and thus must follow the type T termination visit, if both are used.

As for all other scheduled visits a termination window visit (W) will only be scheduled if the cycle has not already terminated by some other method before it becomes due.

It is legal to have a type W visit defined in more than one in-study cycle. This may be necessary when different subjects follow different in-study cycles. It also allows a study with 2 or more phases to have different termination windows for each phase.

8.5.3. Visit Label

A short (32 character maximum) descriptive must be provided for each visit (e.g. Cycle 1 Baseline, Week 2 Follow-up, Diary Card Week 22, etc.). This label is used to describe overdue visits on quality control reports if the visit is flagged as overdue. Thus each visit must have a unique visit label.

When a visit list is defined (e.g. 5001-5999 for adverse event reports) the label must include some part of the visit number in order to create unique labels. This can be accomplished using the same visit number substitutions described for the study page map. The label for visits 5000-5999 in our example might be specified as:

```
Adverse Report #%{S.2.3}
```

which indicates that, starting at the 2nd digit, the next 3 digits are to be included in the label immediately following

```
Adverse Report #
```

This results in the last 3 digits of the visit number being used to produce visit labels as in the following examples:

```
Adverse Report #001
Adverse Report #002
```

8.5.4. Due Day

A visit due day, relative to the cycle baseline, must be specified for all scheduled visits. The due day must be negative for all pre-baseline visits (P), zero for all baseline visits (i.e. the first type X visit in the screening cycle, and all type B and F visits in in-study cycles), and positive for all scheduled follow-up visits (S, T). The due day can be greater than or equal to zero for type R visits. For screening visits (X) the due day must be zero if the visit is unscheduled and greater than zero if it is scheduled within the screening cycle. No due day is allowed for optional visits (O, E, A) or for visits linked to the next scheduled visit (r).

8.5.5. Overdue Allowance

How long do you want to wait before calling a visit overdue? It may be OK if a visit occurs a few days late, or you might want to allow a few days for completed assessments to be transmitted, and a few days for data entry staff to review incoming CRFs, particularly if the incoming volume is large, or staffing is light. For all of these reasons it is a good idea to specify a grace period, following each visit due date, during which overdue visit queries will not be created. If the overdue allowance is set to zero, and a visit has not arrived by its due date, an overdue visit query will be created the very next day (or to be more exact, the next time the **DF_QCupdate** program is run).

8.5.6. Visit Date Location

This specification identifies a date field, defined with the `Visit Date` attribute, which will be used to record the date on which the assessment occurred. This date must exist on one of the required plates for the visit. DFdiscover uses all dates

defined with the `Visit Date` attribute, so this is not a limiting specification. However, the date specified here has priority and will be used for subject scheduling if more then one `Visit Date` is available for any assessment.

8.5.7. CRF Plates

The CRF plates definition for each visit includes required plates, optional plates, the display order in which these plates are to be listed in the subject binders, and any plates that are submitted when a visit is missed. All relevant plates should be specified for each visit. This allows DFdiscover to generate a warning for any plates that arrive unexpectedly. Any plate not listed as required, optional or a missed visit plate will be flagged as unexpected and listed by report **DF_PTvisits**.

Plates in the required list are expected when the visit arrives, unless they have been made optional or unexpected by the conditional plate map. Similarly, plates in the optional list are not considered necessary unless they have been made required by the conditional map. Plates that are required, after conditional plate map actions have been accounted for, will be identified as missing if they have not arrived when other plates for that visit have been received. The type of plate(s) received (optional, required or unexpected) does not matter. Arrival of any plate with a given visit number indicates that the visit has occurred and thus that all plate requirements should be checked.

The check for required plates does not occur immediately. After all, if it did, DFdiscover would identify all but the first plate to arrive as missing as soon as the first plate was entered into the database. Instead, missing plates and overdue visits are identified when the **DF_QCupdate** report is run. This can be scheduled to execute automatically (using the UNIX cron process) or can be executed manually as deemed appropriate for the study.

If the clinical investigators know that a particular visit was missed and will never be made up they can notify the data management office (and DFdiscover) of this fact, and thereby avoid receiving an overdue visit query, by submitting a missed visit report. When DFdiscover receives a plate that has been classified as a missed visit plate for that visit, it considers the visit as completed even though none of the required plates have been received.

If this method of registering missed plates and visits is not used, it will be necessary to either record missed visits, when you learn of them, or require that investigators submit all CRF pages for all assessments, including those which are missed or have not been completed.

Required and optional plates and their display order are entered into the visit map using the **DFsetup** Visit map view, as a list of comma or space separated values and value ranges, like this `1-3,7,9,10-12,101`, or this `1-3 7 9 10-12 101`.

8.6. Display Order

Display Order allows you to customize page order within each assessment in the **DFexplore** subject binders. Required and optional plate numbers may be specified here in the order that you want them to appear in the subject binders.

8.7. Visit Map Examples

Example 8.1. A Simple Visit Map

The following example shows a visit map as it appears in the configuration file `DFvisit_map`. It has a screening cycle, one in-study cycle and an end cycle. Each cycle begins with a cycle definition record followed by one record for each of the cycle visits.

```
# 3 screening visits at 7 day intervals with a 2 day overdue allowance.
# The visit date is in field 10 of plate 1 for all 3 screening visits.
# Each screening visit has one required plate (plate 1), and
# no optional plates, and no missed visit plate
0|C|SCREENING|S|0|0|N
```

```
91|X|Screen #1|1|10|00|0|1||||
92|X|Screen #2|1|10|07|2|1||||
93|X|Screen #3|1|10|14|2|1||||

# There is just 1 in-study cycle, it is required and scheduled 7 days
# after termination of the screening cycle, with no overdue allowance.
# Plate 12 is a missed visit form which can be submitted at any visit in this cycle.
1|C|IN-STUDY VISITS|R|7|0|T
51|P|Pre-entry|3|10|-2|0|2,3||12||
0|B|Baseline|3|10|0|0|4-9|101,105|12||
2|r|Baseline Lab Test|||0|0|21-23||12||
3|S|Month 3|5|10|91|0|5||12||
31-39|O|Interim Visit 3.%{S.2.1}|5|10|5||12||
6|S|Month 6|5|10|183|0|5||12||
61-69|O|Interim Visit 6.%{S.2.1}|5|10|5||12||
9|S|Month 9|5|10|274|0|5||12||
91-99|O|Interim Visit 9.%{S.2.1}|5|10|5||12||
12|T|Month 12|5|10|365|0|5||12||
100|E|Early Term|71|10|0|0|71||12||
210|R|Clinical Eval|72|10|0|0|72||12||
211|R|Subject Eval|73|10|30|0|73||12||

# End cycle for unscheduled reports - death and adverse events
2|C|REPORTS|E|0|0|N
80|A|Death Report|9|10|0|0|99||||
101-199|O|AE Report #%{S.2.2}|||0|0|98||||
```

8.8. Conditional Maps

Conditions can be specified for cycles, visits, plates and terminations, to override the visit map specification when the condition is met. By specifying a database test in the appropriate conditional map, you can change any cycle, visit or plate to make it required, optional or unexpected; or indicate that subject follow-up terminates on a specified visit. Conditions can thus override the initial requirement for cycles, visits and plates that were specified in the visit map.

Conditions are entered using the conditional: cycles, visits, plates and termination dialogs under the View menu in **DFsetup**. The setup files created by these views are stored in the study `lib` directory and are named: `DFccycle_map`, `DFcvisit_map`, `DFcplate_map` and `DFcterm_map`. Each of these files follows the same simple structure.

Each condition begins with the definition of a database test, followed by a list of actions to be performed if the test is met. The actions relate to cycles, visits, plates or terminations, depending on which conditional map is used, but the way in which the database tests are defined is the same for all of the conditional maps, as follows:

- Lines beginning with a # are comments used to document the intent of each specification.
- Conditions are composed of IF and AND statements each comprised of 5 fields separated by `|`.
- Each condition begins with the keyword `IF` in the first field, followed by the visit, plate and field number of the database field to be tested. The 5th, and last field, is the data value or values that make the condition true.
- To evaluate a condition at multiple visits, a visit list comprised of one or more ranges and values separated by commas, e.g. 2,10-15,101-199,300 can be specified in the visit field. An asterisk * can be used to evaluate the condition at all visits.
- If the condition requires testing more than one data field, the IF statement may be followed by one or more AND statements, constructed just like the IF statement except that the first field contains the keyword AND. When AND statements are used the condition is not met unless all of the AND tests evaluate true.

- The visit field in an AND statement can contain either a specified visit number or an asterisk *. If the visit in the IF statement is also specified with an asterisk (*) then the AND condition is evaluated at the same visit that met the condition specified in the IF statement, otherwise such AND conditions are true if met at any visit.
- There is no OR statement; instead another condition can be specified. When multiple conditions are specified for the same event (cycle, visit, plate or termination) they are evaluated in order and in the case of conflicts the last condition wins.

Each condition is followed by a specification of the effect it has on cycles, visits, plates or terminations when the condition is met. These specifications are described in a separate section below for each of the 4 conditional maps, but first let's look at some example conditions.

Example 8.2. Example conditions

```
# Condition is true if visit 0, plate 1, field 13 equals 1
IF|0|1|13|1

# Condition is true if visit 0, plate 1, field 13 equals 1
# and visit 100, plate 20, field 25 is not blank
# and there is a valid date in visit 100, plate 20, field 26 that precedes 01/JAN/2004
# Note: date tests must be specified in the format used by the specified data field
IF|0|1|13|1
AND|100|20|25|!blank
AND|100|20|26|<01/JAN/2004

# Condition is true at visits 100 to 199 if plate 41, field 10 is greater than 2
# and at visit 1, plate 14, field 16 equals 1
# and at any visit plate 3, field 17 is less than 5
IF|100-199|41|10|>2
AND|1|14|16|1
AND|*|3|17|<5

# Condition is true at visit where plate 56, field 20 is not blank
# and at the same visit plate 56, field 21 is blank
# and at the same visit plate 55, field 16 has a value between 1 and 199 inclusive
IF|*|56|20|!blank
AND|*|56|21|blank
AND|*|55|16|1-199
```

See Section A.5, “Conditional Tests” for a description of the tests that may be specified in each IF and AND statement. The same tests can be used in all of the conditional maps.

8.8.1. Conditional Cycle Map

Each cycle defined in the visit map starts with a default requirement. The screening cycle, if present, is by default required for all subjects. Each in-study cycle is defined in the visit map as required, optional or conditional. An optional cycle is converted to required if any of the visits defined in the cycle arrive. A conditional cycle is unexpected unless it becomes required through a condition specified in the conditional cycle map. The end cycle, if present, is not required being just a container for unscheduled visits.

Using the conditional cycle map the requirement for screening and in-study cycles can be changed from their default visit map setting to any of: required, optional or excluded.

- **required.**

Cycles that become required are due on their scheduled start date. However, abort dates take precedence over cycle conditions. Thus, if there is an abort date for the subject that falls on or before the scheduled start date of the cycle, the cycle will be classified as unexpected and the visits in that cycle will not be scheduled or classified as overdue.

- **optional.**

  The possible due date for optional cycles is calculated by **DF_QCupdate** and is reported by **DF_PTvisits** but optional cycles are not required and may be skipped. However, once a visit from an optional cycle arrives the cycle changes to required, and visit scheduling and visit requirement rules become active for that cycle.

- **excluded.**

  If a condition indicates that a cycle is excluded all visits in that cycle become unexpected. If any visit arrives in an excluded cycle the visit will be flagged as unexpected and will be reported by **DF_PTunexpected**. The arrival of unexpected visits in an excluded cycle does not trigger visit scheduling within the cycle, i.e. all visits remain unexpected.

If there is more than one condition for a given cycle and the conditions have different actions, e.g. one making the cycle required while another makes it excluded, there is the potential for conflict, which will arise if 2 or more conditions having different actions are met. **DF_QCupdate** resolves such conflicts by applying the action specified for the last condition met. Thus it is important to consider the order in which conditions are specified in the conditional cycle map. If necessary conditions can be re-ordered within the Conditional Cycles view in **DFsetup**.

The following example illustrates how conditional cycle specifications are stored in `DFccycle_map`. It shows 5 conditions each followed by one or more action statements. Each action statement consists of 2 fields. The first field indicates the action to be applied when the condition is met, and the second field indicates the cycle or cycles to which the action is applied. If the first field contains a + (plus sign), the cycle(s) are required; a - (minus sign) indicates that they are unexpected, and a ~ (tilde) indicates that they are optional.

Example 8.3. A conditional cycle map with 5 conditions

```
# After screening visit 0 subjects proceed to:
# Cycle 1 if hypertensive, defined by: visit 0, plate 1, field 13 equals 1
# Cycle 2 if normotensive, defined by: visit 0, plate 1, field 13 equals 2
# Hypertensive subjects may then optionally proceed to cycle 3,
# but normotensive subjects never do
IF|0|1|13|1
+|1
-|2
~|3
IF|0|1|13|2
+|2
-|1,3

# Cycle 3 requirements depend on the final diastolic reading recorded at
# visit 19 of cycle 1, in field 22 of plate 9. Cycle 3:
# - is not to be done if then diastolic is less then 80
# ~ optional if the diastolic is 80-89
# + required if the diastolic is 90 or more
IF|19|9|22|<80
-|3
IF|19|9|22|80-89
~|3
IF|19|9|22|>89
```

```
+|3
```

In this example visit 19 only occurs in cycle 1, thus the last 3 conditions will only be applied to subjects who were hypertensive at screening as these are the only subjects who enter cycle 1. Note that the first condition indicates that cycle 3 is optional for hypertensive subjects and that this can be changed by conditions 3 and 5. Remember that conditions are applied in the order in which they are defined in the conditional map, and that if conflicts arise the last condition wins.

This example is a little over specified. For hypertensive subjects, the first condition makes cycle 3 optional, as does the 4th condition. Thus the 4th condition is not really necessary. While this will not lead to errors it will have an impact on processing time and thus in general should be avoided.

DFdiscover report **DF_ICvisitmap** checks the conditional cycle map for syntax errors, and **DF_SSvisitmap** can be used to produce a numbered list of the conditions, according to the order in which they are defined in `DFccycle_map`. Cycles effected by the conditional cycle map are indicated on the report produced by **DF_PTvisits** with a `CC#[rox]` tag, where # is the condition order number, and the suffix `r, o` or `x` indicates whether the cycle became required, optional or excluded respectively, when the condition was met.

8.8.2. Conditional Visit Map

Each visit defined in the visit map starts with a default requirement, which is either optional (visit types O, E, A) or required (all the rest). Using the conditional visit map the requirement for any visit can be changed to any of: required, optional or excluded.

- **required.**

  Visits that become required are due on their scheduled due date if they have one. This is not the case for optional visits (O, E, A), as these visits are not scheduled. Instead, optional visits become due immediately on being triggered by the conditional visit map, provided they are in a cycle that has already started. If the cycle has not yet started, these visits will become required as soon as any other visit in the cycle arrives.

  If the subject has an abort or cycle termination date that falls on or before the scheduled due date for the visit, the visit will be classified as unexpected and will not be scheduled or classified as overdue. However, cycle termination and abort dates have no effect on optional visits that become required via the conditional visit map. Such visits are considered overdue regardless of when the termination occurred.

  If the conditional cycle and visit maps conflict, which one wins? An important point to remember is that the conditional visit map can only make visits required, not cycles, and that cycle requirements override visit requirements. Thus, a visit made required by the conditional visit map does not automatically make its cycle required, and in fact if the cycle is unexpected, the visit will also be considered unexpected, despite the conditional visit map calling it required.

- **optional.**

  The conditional visit map may be used to change a required or unexpected visit to optional. If a previously required visit is made optional it will be scheduled using its visit map scheduling specifications, but it may be skipped and will not be called overdue if it does not occur.

- **excluded.**

  If a visit that has been excluded by a condition arrives, it will be flagged as unexpected and will be reported by **DF_PTunexpected**.

If there is more than one condition for a given visit and the conditions have different actions, e.g. one making the visit required while another makes it excluded, there is the potential for conflict, which will arise if 2 or more conditions having different actions are met. **DF_QCupdate** resolves such conflicts by applying the action specified for the last condition met. Thus it is important to consider the order in which conditions are specified in the conditional visit map. If necessary, conditions can be re-ordered by editing `DFccycle_map` directly.

Conditional visit map action statements have the same syntax as described above for the conditional cycle map, except of course that the second field contains a list of the visits (not cycles) effected by the action. When the first field contains a + (plus sign), the visit(s) are required; a - (minus sign) indicates that they are unexpected, and a ~ (tilde) indicates that they are optional.

Conditional visit map action statements support one feature not supported by any of the other conditional maps. This is the ability to use the key word `value`, which is interpreted as the value of the data field, tested by the IF statement, to define the list of visits effected by the condition. The following example illustrates how conditional visit specifications using this feature would appear in `DFcvisit_map`.

Example 8.4. Check for Expected AE Reports

```
# Adverse Event reports use visit numbers 5001-5999, where the
# last 3 digits correspond to the AE Report# recorded on AE forms.
# At each follow-up visit investigators are asked:
# Has the subject had any adverse events since the last visit?
# [ ]no  [ ]yes -> specify AE report#s: [ ][ ][ ] to [ ][ ][ ]
# The AE report numbers are recorded in fields 12 and 13 on plate 101.
IF|*|101|12|>0
+|5001~5000+value
IF|*|101|13|>0
+|5001~5000+value
```

This example shows how the conditional visit map can be used to make sure that all event reports are received. Typically event reports are numbered sequentially and the event number determines the DFdiscover visit number. In this example the visit number = 5000 + the AE report number. Consider the first condition. It will be true, if at any visit, field 12 on plate 101 contains a value of 1 or more. The following action statement indicates that when this is true, visits 5001 to 5000 + the value in field 12, are required. Thus if the value in field 12 is 7, visits 5001-5007 will become required, and DFdiscover will create an overdue visit query for any assessment in this range that is missing.

If investigators are instructed to complete both fields 12 and 13, even when only one adverse event has occurred, then only the second condition would be required, as field 13 must be at least as large as field 12, making the first conditional test unnecessary.

Note that when both tests are specified, as above, each visit in the first range will be tested again in the second range (e.g. if AE#s 6 to 7 are recorded, 5001-5006 is include in 5001-5007). So if AE 4 for example is missing does this mean that DFdiscover will generate 2 overdue visit queries for visit 5004? Of course the answer is no. The reason has to do with how DFdiscover processes this information. First visit requirements are established by evaluating the visit type and all conditional visit map tests. Remember that when multiple tests apply to the same case the last test wins. So there is only ever one result from any set of conditional specifications. Then if the visit has been determined to be required DFdiscover checks to see if it is overdue.

Example 8.5. Check for Missing AE Visit Numbers

This feature can also be used to ensure that all visits in a sequential series are received, by checking for all lower visit numbers, as illustrated in the following example.

```
# Adverse Event reports use visit numbers 5001-5999 in sequential order.
# These visit numbers, like all DFdiscover visit numbers, appear in field 6.
# AEs are reported on plate 101.
# If for example report 5012 has arrived, 5001-5012 should all be in the database
IF|*|101|6|>5001
+|5001~value
```

These 2 examples illustrate the 2 supported uses of the `value` keyword. These are the only supported uses. No other mathematical manipulations are allowed.

Use DFdiscover report **DF_ICvisitmap** to check the conditional visit map for syntax errors, and **DF_SSvisitmap** to produce a numbered list of the conditions, according to the order in which they are defined in `DFcvisit_map`. Visits effected by the conditional visit map are indicated on the report produced by **DF_PTvisits** with a `CV#[rox]` tag, where # is the condition order number, and the suffix `r, o` or x indicates whether the visit became required, optional or excluded respectively, when the condition was met.

8.8.3. Conditional Plate Map

For each visit, a list of required and optional plates is defined in the visit map. Think of this as the default plate requirements. Using the conditional plate map the requirement for any of these plates can be changed to required, optional or excluded.

Plate requirements for each visit are checked by **DF_QCupdate** if at least one plate is found for that visit in the study database. Each visit found in the database is checked regardless of whether the visit was required, optional or unexpected. Missing plate queries are created for any required plate that is missing, but queries are not created automatically for unexpected records. Instead these cases are written to the DFdiscover retrieval file and displayed by **DF_PTunexpected**. These tools should be used to review the unexpected cases so that the appropriate action can be taken. At one extreme unexpected records may arise because of a simple data entry error and at the other extreme they may identify a serious protocol violation that requires personal attention.

> **Important**
>
> may contain multiple entries for the same set of keys. This is because a plate may be rendered unexpected by a condition being met in `DFcplate_map` and `DFcvisit_map`. In **DFexplore**, an unexpected plate can only be retrieved once in a given set of records, regardless of the condition that made it unexpected.

If the conditional plate map contains multiple conditions affecting the same plates at the same visits it is possible for conflicts to occur. One condition might indicate that a plate is required while another indicates that it is optional or excluded at the same visits. **DF_QCupdate** evaluates conditions in the order in which they are defined in the conditional plate map, and for each action the last condition that is met wins. It is thus important to consider the order in which conditions are defined in the conditional plate map if different actions have different consequences for a given visit/plate combination.

In a conditional plate map, the action statements begin with either `+`, `-`, or `~` (for required, unexpected and optional respectively), followed by a visit number, a visit list or * (asterisk). This indicates the type of action and the visits to which the action is to be applied. The second field, following the `|`, is the list of plates affected by the condition at these visits.

An asterisk, in place of a visit number or list, in the first field of an action statement, indicates that the action is to be applied to the visit at which the condition was met. If the condition includes AND statements it is possible for the visits referred to in the IF and AND statements to differ. When this occurs the condition is classified as having been met at the visit referenced in the IF statement. Thus when using * to indicate that plates are required at the visit on which the condition was met make sure the appropriate visit is referenced in the IF statement.

Example 8.6. Conditional plate map with 2 conditions

```
# For follow-up visits 3-12 and 16 performed after Dec 31,2003
# a new form, plates 216-218, is required.
# This form did not exist prior to this date and thus would be unexpected
# for follow-up visits performed before Jan 1, 2004.
# The visit date is recorded in dd/mm/yy format in field 10 of plate 200
# at each of these follow-up visits.
IF|3-12,16|200|10|>31/12/03
+*|216-218
```

```
IF|3-12,16|200|10|<01/01/04
-*|216-218
```

The first entry in the example above specifies that, for visits 3-12 and 16, plates 216-218 are required if the condition is met at that visit; and the second condition indicates that, for visits 3-12 and 16, plates 216-218 are not expected when the second condition is met at that visit. If plate 200 is only used at visits 3-12 and 16, the above specification could also be entered as shown below using an asterisk instead of a visit list in the condition specification. The following conditional plate map entries would then be equivalent to the entries above.

```
# For all follow-up visits performed after Dec 31,2003
# a new form, plates 216-218, is required.
# This form did not exist prior to this date and thus would be unexpected
# for all follow-up visits performed before Jan 1, 2004.
# The visit date is recorded in dd/mm/yy format in field 10 of plate 200
# at each of these follow-up visits.
# Plate 200 is only used at follow-up visits (3-12 and 16).
IF|*|200|10|>31/12/03
+*|216-218
IF|*|200|10|<01/01/04
-*|216-218
```

Example 8.7. Conditional plate map specification with multiple actions

It is possible to indicate that a condition met at one visit has implications for many other visits, and also to indicate that the condition has more than one consequence. These features are illustrated in the following example in which a long or short version of a cardiovascular disease (CVD) evaluation form is to be used for subjects entering the trial with different CVD histories. The following specifications will make the appropriate plates required for each subject, will result in missing page queries if they do not arrive, and will also result in cases being put on the unexpected list if the wrong form is used for any subject.

```
# For subjects with a recent history of heart disease or stroke
# (field 22 on plate 15 at baseline visit 10 equals 2), the Long CVD Evaluation
# form (plates 60-66) is required at each follow-up (visits 20-80).
IF|10|15|22|2
+20-80|60-66
-20-80|67-68

# For subjects without a recent history of heart disease or stroke
# (field 22 on plate 15 at baseline visit 10 equals 1), the Quick CVD Evaluation
# form (plates 67-68) is required at each follow-up (visits 20-80).
IF|10|15|22|1
-20-80|60-66
+20-80|67-68
```

Use DFdiscover report **DF_ICvisitmap** to check the conditional plate map for syntax errors, and **DF_SSvisitmap** to produce a numbered list of the conditions, according to the order in which they are defined in `DFccycle_map`.

8.8.4. Conditional Termination Map

The conditional termination map is used to specify database conditions that indicate that follow-up has terminated, either for the current cycle, or for all cycles. Conditions are specified as for all other conditional maps, using IF and AND statements, and each condition is followed by a single action which can be either: A, to indicate that all follow-up is aborted when the condition is true, or E, to indicate early termination of the cycle in which the condition is met. Both of these uses are illustrated in the following examples.

Example 8.8. Conditional termination map with 3 conditions

```
# During screening subjects abort (do not proceed to any in-study cycles)
# if any of the 3 eligibility questions (fields 14-16 on plate 1 at visit 0)
# is answered no (which is coded 1)
IF|0|1|14|1
A
IF|0|1|15|1
A
IF|0|1|16|1
A

# The trial consists of several treatment cycles, each of which may end early.
# A question represented by field 33 on plate 20 at each follow-up visit asks:
# "Is the subject continuing with this course of treatment?"
# If the answer is no (code 1) the cycle terminates at that point
# and the next cycle is scheduled.
IF|*|20|33|1
E
```

When termination occurs the visit date of the visit at which termination was triggered becomes the abort or early cycle termination date, and is used like any other termination date to decide whether other visits are still required. Visits scheduled after a termination date are not required but those scheduled prior to termination are still expected and will be called overdue if they have not arrived.

Use DFdiscover report **DF_ICvisitmap** to check the conditional termination map for syntax errors, and **DF_SSvisitmap** to produce a numbered list of the conditions, according to the order in which they are defined in `DFcterm_map`. Visits at with a conditional termination has been triggered are indicated on the report produced by **DF_PTvisits** with a `CT#` tag, where # is the condition order number.

8.9. Early Termination Plates

In addition to defining visits which signal termination (T, W, E, A, and F), it is also possible to define plates that signal early termination. Such plates can be identified by selecting **Plate** > **Edit** in **DFsetup**. However, plates can only be marked as indicating early termination of the cycle with which they are associated. Thus they are like termination (T) and early termination (E) visits. They cannot be used to define an abort event, like final (F) and abort (A) visits.

Arrival of an early termination plate terminates the cycle that contains the visit number on the early termination plate. Since plates which signal early termination only terminate the cycle they are associated with, it would be inconsistent to include such plates in an abort (A) visit. It would also be inconsistent to specify an early termination plate in the required plate list for a visit, except perhaps for an early termination (E) visit.

8.10. Missed Visit Plates

In many clinical trials it is useful to design a special form that can be used by the clinical sites to report missed visits. Such forms can typically be a single CRF plate and need little more than a place to record the reason for the missed visit. The key to making this work is to make sure that the clinical sites enter the visit number of the missed visit in the visit key field at the top of the form before they submit it. This is how DFdiscover knows which visit was missed. Also, the plate number of this form must be registered in the visit map, as the missed visit plate, for each visit at which it is allowed to be used.

When a registered missed visit plate arrives DFdiscover will stop generating overdue visit and missing plate queries for that visit, and will remove any such queries that might already exist in the database.

If a missed visit plate arrives that has not been registered in the visit map as a missed visit plate for that visit, DFdiscover will mark the plate unexpected, and will not classify the visit as missed.

8.11. Implementation & Scheduling Rules

So far in this chapter we have described the separate components that comprise the subject visit scheduling specifications, and the rules that govern how each component works. In this final section we describe when and how these rules are implemented and some additional rules related to how they work together.

8.11.1. DF_QCupdate

Schedule View

Beginning with DFdiscover 2018 Version 5.1.0, implementation of user-defined schedule rules is handled in the **Schedule View** of **DFexplore**. That implementation is quicker to update, providing feedback about subject schedule, missing data and overdue status in a near real-time fashion.

DF_QCupdate will be deprecated in a future release and is provided for users so that they can transition their study schedule monitoring tasks to Schedule View.

A report that implements the scheduling rules is named **DF_QCupdate**. Each time it is executed it reads the current study setup files to identify the visit date fields and current visit scheduling requirements, and then updates each subject in the database with regard to: visit scheduling, overdue visit queries, missing plate queries, and the arrival of unexpected visits and CRF plates. If errors have been made in the scheduling specifications, or specifications need to be added, dropped or changed, simply modify the appropriate setup specifications and re-run **DF_QCupdate**

An obvious limitation of this reliance on **DF_QCupdate** is that information regarding subject scheduling is only up to date immediately after **DF_QCupdate** is executed. Subsequent changes to the study database will not be reflected until the next execution of **DF_QCupdate**. Thus it is important that **DF_QCupdate** be run before creating Query Reports for distribution to the clinical sites, or before creating internal reports related to visit scheduling if such reports need to be right up to date. One way to mitigate this limitation is to run **DF_QCupdate** in a UNIX cron job at specified times of the day.

8.11.2. Termination of Subject Follow-up

The screening and in-study cycles consist of an order sequence of visits that must be terminated either by reaching the planned termination visit within each cycle or by terminating the cycle early. Sometimes a clinical trial needs the ability to terminate a cycle early so another cycle can begin, or to terminate all follow-up because some event has occurred which withdraws the subject from the trial. DFdiscover recognizes a termination event in each of the following ways:

- **Visit Type (see Visit Type).**

  Arrival of any of the following visit types signals termination of follow-up as of the visit date for that visit, defined using the special DFdiscover `Visit Date` attribute. The first 3 visit types, T,W,and E, terminate follow-up only for the cycle in which they are defined. A screening visit, X, only terminates the screening cycle if it is the last required screening visit in the screening cycle. A baseline visit, B, only terminates the cycle in the special case where it is the only visit defined in the cycle. Arrival of either of the last 2 visit types, F and A, terminate all further follow-up for all cycles.

  - T - scheduled cycle termination
  - W - scheduled cycle termination within a calendar window
  - E - early cycle termination
  - X - the last required screening visit in the screening cycle

- B - a baseline visit if it is the only visit in an in-study cycle
- F - termination of the final in-study cycle, ends all follow-up
- A - abort all follow-up

- **Early Termination Plate (see Section 8.9, “Early Termination Plates”).**

  Using the plate dialog of **DFsetup** any plate can be classified as signaling early termination of the cycle in which it is received. This designation is typically used for special study forms designed to record cycle termination events like treatment failure or intolerable side effects. Termination is deemed to have occurred at the visit number recorded on the early termination plate, and the visit date for this visit, defined using the special DFdiscover `Visit Date` attribute, is used as the termination date.

- **Conditional Termination Events (see Section 8.8.4, “Conditional Termination Map”).**

  Using the conditional termination map, database conditions can be defined that indicate either that cycle follow-up terminates, or all follow-up terminates, as of the visit at which the specified condition was met. Again, the visit date for this visit, defined using the special DFdiscover `Visit Date` attribute, is used as the termination date.

If termination occurs at more than one visit within a cycle, the earliest termination date will prevail. Typically this will mean that the second termination visit is unexpected because it occurred after the cycle had already terminated.

If an abort date exists that precedes the scheduled start date for a cycle, the cycle termination date is set to the abort date, indicating that the cycle was terminated before it became due.

8.11.3. Effect of Early Termination on Visit Requirements

DF_QCupdate considers all termination events when deciding whether a visit is overdue. Two rules are applied, one relating to visit order as defined in the visit map, and the other relating to visit scheduling. These rules apply to all termination events regardless of how they arise (i.e. whether by visit type, an early termination plate, or a conditional termination map specification).

The first rule specifies that when a termination event occurs within a cycle, visits that follow the terminating visit in that cycle become unexpected. This rule applies to both scheduled and unscheduled visits up to and including the planned type T or W cycle termination visit. But, it does not apply to unscheduled visits (if any) that are positioned after the planned cycle termination visit. Such unscheduled visits, will not be called unexpected if they arrive after the cycle has terminated. Also, if these visits become required by a conditional visit map specification they will be called overdue if they have not arrived, even if the cycle has terminated.

The second rule, related to visit scheduling, states that when a termination event occurs, any scheduled visits that have not yet arrived are called overdue if they were due prior to the date of termination, but not if they were scheduled to occur on or after the termination date. For example, if the planned cycle termination visit is performed early, the cycle will terminate as of the date on which the visit was performed and any visits scheduled to occur prior to that date will no longer be required, and in fact will be flagged as unexpected should they arrive.

Visits that are reported as having occurred after a termination event will be flagged as unexpected in the DFdiscover retrieval file . This also applies to a second termination visit if it occurs after the subject has already terminated. All unexpected data records flagged by **DF_QCupdate**, including visits that occurred after termination, can be reviewed using **DFexplore** or **DF_PTunexpected**.

Optional visits defined in the end cycle are unscheduled and thus are not subject to the timing of termination events, including a final abort event. Consequently, if such visits become required because of a conditional visit map specification they are required regardless of the timing of termination events.

8.11.4. Identification of Missing Plates

For each subject, a missing plate query is created for each required plate that is missing for each visit found in the database. If the visit exists in the database, it's visit requirements, as defined in the visit map and conditional plate map will be applied. It does not matter whether the visit was required, optional or unexpected.

The trigger for missing plate checks is the existence of the visit in the database. Thus a conditional plate map specification that makes a plate required will only become active once the visit arrives, i.e. once at least one plate is entered into the database for that subject with that visit number.

Conditional plate specifications that make a plate required at some visit do not imply that the visit is also required. Instead they only indicate that the plate is required should the visit arrive. The conditional visit map must be used to specify conditions that make a visit required.

8.11.5. Conditional Cycles and Conditional Visits

The requirement for cycles and visits can be modified independently using the conditional cycle and conditional visit maps. Thus we need rules that define how cycle and visit requirements interact when they are in conflict. The basic rule is that cycle requirements win out over visit requirements. Making a visit required does not imply that the cycle is required. If a cycle is excluded by a conditional cycle map specification, all visits in that cycle become excluded regardless of any conditional visit map specifications.

Visit requirements within a required cycle kick in when the cycle becomes due, and visit requirements within an optional cycle kick in when the cycle starts. Essentially an optional cycle that starts becomes required on being used. Visit requirements never kick in for an unexpected cycle, because cycle requirements trump visit requirements. All visits in an unexpected cycle are automatically called unexpected as well.

Chapter 9. CRF Design Guidelines

9.1. Introduction

Each DFdiscover database requires a set of Case Report Forms (CRFs). CRFs can be created in a page design package like FrameMaker, InDesign, etc. or created directly in **DFsetup** as an eCRF. This section applies primarily to CRFs created using the former method, not the latter. The study CRFs are used in the following ways:

- When setting up a new study the CRFs are imported into **DFsetup** and used to define the size and location of each data field, as well as the field properties: name, description, legal range, etc. Each unique CRF page (aka a 'plate') corresponds to a database table, and becomes the background screen for data entry in **DFexplore**.
- At the clinical sites paper CRFs may be used as data collection forms on which subject data is recorded and then faxed to DFdiscover.
- Alternatively CRFs may be used as work sheets on which subject data is recorded prior to using **DFexplore** to enter the data directly into the study database over the internet.

When designing CRFs it is important to remember that they may serve both as paper data collection forms and as the background screens for data entry. For example, if you plan to include 'hidden fields' on a data entry screen, to be seen and used by designated staff for coding or other purposes, blank space must be provided on the corresponding CRF page where you want these fields to appear.

Faxed CRFs can be read by the DFdiscover intelligent character recognition (ICR) software to create an initial data record as soon as they arrive. Handwritten numbers, dates, check boxes and visual analog scales can be read by ICR but any handwritten text must be entered manually in **DFexplore**. ICR accuracy can be maximized by following the CRF guidelines described in this section.

9.2. CRF Plates

A plate is a single page data collection form having unique data field content and layout. Each plate corresponds to a study database table used to store the subject data records. Most studies will not use more than 30-50 plates, but some plates may be repeated because the same data is collected at different visits or for events that may occur more than once.

DFdiscover supports plates in portrait orientation only and expects all plates to be 8.3" to 8.5" wide. Length may vary anywhere from 4.75" to 11.69" (A4).

Each plate must be assigned a unique number, in the range 1-500. Unless otherwise specified in the Page Map configuration file, **DFexplore** displays CRF pages within each assessment in the subject binders sorted in ascending numerical order by plate number.

9.3. Barcodes

A barcode must appear at the top of any CRF pages that will be faxed. We recommend adding barcodes to all CRF pages including pages to be entered directly using **DFexplore**, as this allows the CRFs to be collected by fax should the need arise. A CRF barcode has the following components that can be created using **DFbarcode** described in *Barcode User Guide, Barcode User Guide*.

- A registration line, 7.5" long by .0625" wide, located .5" from the top of the page and .5" from the left edge of the page. It is used to locate the top of the page and de-skew it if needed. Although titles (e.g. study name) may be printed above

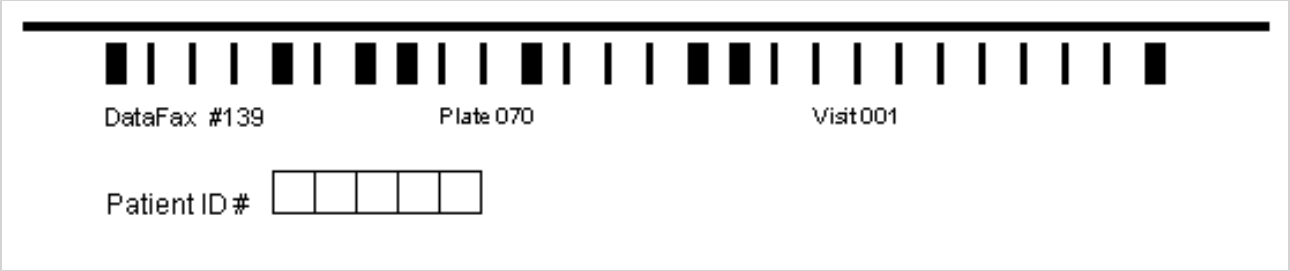
the registration line this area is not preserved on either the **DFexplore** data entry backgrounds or any faxed CRF pages received by DFdiscover.

- A 3 part barcode located below the registration line, which includes the study database number (1-255) and plate number (1-500), followed optionally by an assessment or visit number (0-511). A visit number can only appear in the barcode on plates collected at only one visit.

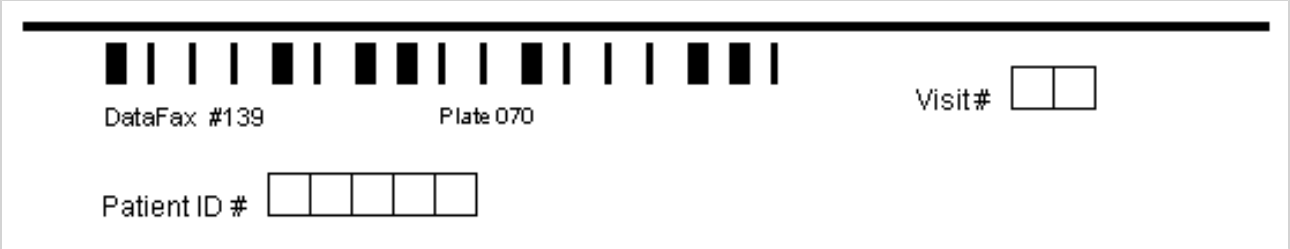
The following examples illustrate the registration line and barcode created by **DFbarcode**. For those interested in the details:

- The keys are binary numbers represented by 8 bars for the study number and 9 bars each for the plate and visit numbers.
- Each '1' in a binary number is represented by a black bar .12" wide, and each '0' is represented by a black bar .04" wide.
- Each bar is .25" high.
- The first bar of the study number is positioned with its upper-left corner exactly 0.5" right and 0.125" down from the upper-left corner of the top of the registration bar. The remaining bars are positioned from left to right with their left edges on 0.25" intervals.

Example 9.1. A barcode containing study, plate and visit numbers



Example 9.2. A barcode containing study and plate numbers only



The labels below each component of the barcode identify the key fields so they can be easily read if a page ends up in the unidentified image router and needs to be identified manually.

9.4. Key Fields

As can be inferred from the preceding description of CRF plates, each data record and each faxed CRF page in a DFdiscover study database is identified by 4 numeric keys: Study#, Plate#, Visit# and Subject ID.

9.4.1. Study Number

The study number identifies a DFdiscover database. It appears as the first value in a CRF barcode. Some research projects may require more than one database and thus have multiple study numbers. Study number 254 is used by the DFdiscover Acceptance Test Kit (ATK).

9.4.2. Plate Number

The plate number identifies each unique CRF page and database table. It appears as the second value in a CRF barcode. Numbers 1-500 may be used to define study CRFs. Numbers 501-511 are reserved for metadata and Query Reports and plate 0 refers to records in the new image queue.

9.4.3. Visit/Sequence Number

This key field identifies instances of a plate that may recur for: different clinic visits, subject assessments, hospitalizations, adverse events, etc. It may appear as the third and final value in the barcode on plates that occur at one visit only, or as the first data field (field #6) on the CRF page.

When included in the barcode visit numbers can range from 0 to 511, and when included as the first data field can range from 0 to 65535.

Scheduled clinic visits are typically numbered sequentially from 0 or 1 to some final visit number, or using some meaningful coding scheme, e.g. by follow-up day, week or month. Alternatively planned visits might be numbered 0, 10, 20, etc. to leave gaps for extra visits or possible repetitions of an assessment.

Sequence numbers above the final scheduled follow-up are typically used for numbered reports, e.g. adverse events might be numbered 101-199 in a study ending with a follow-up visit number < 101, where no more than 99 AEs are expected for each subject.

9.4.4. Subject ID

The Subject ID is the fourth and final DFdiscover key field. It must appear as a numeric data field immediately following the visit or sequence number. It cannot appear in the barcode. If possible ID numbers should begin with the site number, e.g. 22001 for the first subject at site 22. This makes it easy to record the legal subject IDs for each clinical site, e.g. 22001-22999.

Subjects cannot be identified by more than one ID field thus it is not recommended to use different ID numbers for different phases of a study, e.g. screening vs. on study vs. post study follow-up.

For presentation purposes, Subject Alias Map can be defined, expanding the possibilities.

9.5. Visit Dates

The study CRFs must include fields that can be used to record the date on which each visit occurred and on which subject follow-up was terminated (see Visit Dates).

9.6. General CRF Design Guidelines

9.6.1. Font and Point Size

- Arial, Helvetica or Times-Roman fonts are transmitted/scanned with little distortion.
- Header labels should be in 12 to 14 point and possibly in bold.
- Choice or coding labels, as well as footnotes, should be in 9 or 10 point and possibly italicized.

9.6.2. Line Weight

Each digit in a numeric or date field, and each option in a check or choice field must be enclosed in a bounding box if it is to be read by the ICR software. These boxes must be visible without gaps when displayed at 100 dpi resolution on screen in **DFexplore**. To accomplish this line weights for drawing boxes, and visual analog scales should be 1-2 points wide.

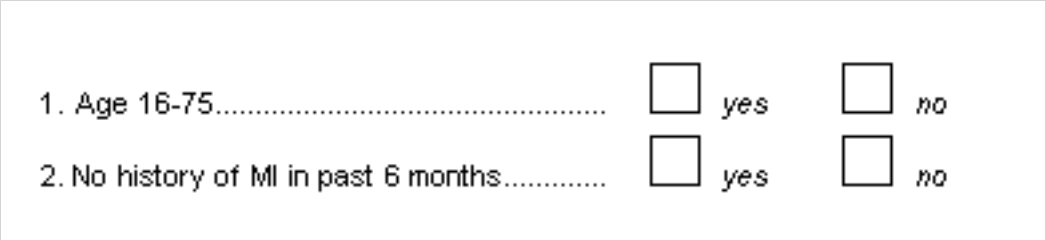
9.6.3. Box Size and Position

DFdiscover ICR is able to read handwritten numbers provided they are written inside boxes between .20" and .30" per side. Smaller sizes (down to .15" per side) are allowed but ICR accuracy will be reduced. The boxes do not need to be square, but the horizontal dimension should never exceed the vertical dimension, and adjacent boxes in multi-digit fields must share the adjacent vertical edges as shown in the following example.

Space may appear between parts of a field, e.g. to separate the integer and decimal parts of a real number, but the gap should be no more than half the box width (see examples below).

ICR is accurate for check and choice fields using boxes between .15" and .25" square.

A .25" space in front of each data field and check box will reduce the risk of it being obstructed when the ICR tries to locate it. It is best to place text labels to the right of check boxes. These guides are illustrated below.



9.6.4. Lines and Decorations

Whether or not logos are placed at the top of pages, and bounding boxes or shading are used to group data fields is a matter of taste, but care must be taken to avoid interfering with the barcode and with the ICR software's ability to locate data fields. Beware of the following problems:

- Shaded areas, logos, and lines drawn around sections of the page should be avoided as they take longer to transmit, may transmit poorly, and increase the disk space required to store the CRF images.
- Shading can be particularly problematic and should be avoided or done in a very light color and tested to demonstrate that it will be dropped by fax machines. Otherwise the ICR software may fail to locate data fields on the page starting from the point at which it was confused by the shading.
- If the barcode is obstructed, faxed CRF pages may end up in the unidentified fax router resulting in the need for manual identification of faxed CRF pages.
- If there is a horizontal line at the bottom of the page it may sometimes be confused with the registration line at the top of the page resulting in a faxed CRF page being flipped upside down.
- If bounding lines coincide or come close to the edge of a data field the ICR software may not be able to locate the field resulting in the need for extra manual data entry.
- Failure to locate a field may result in the ICR software giving up on all subsequent fields on that page.

9.6.5. CRF Instructions

Long instructions should be placed on facing pages and not on the CRFs transmitted to DFdiscover. This reduces the clutter on the CRF and the data entry screen, reduces fax transmission time, and saves disc space when the fax images are stored.

9.7. Supported Field Types

9.7.1. Check

A check field consists of a single box, 1/8" to 1/4" square that can be checked or left blank. ICR accuracy is better if the box is not too small; 1/6" is the recommended minimum length per side.

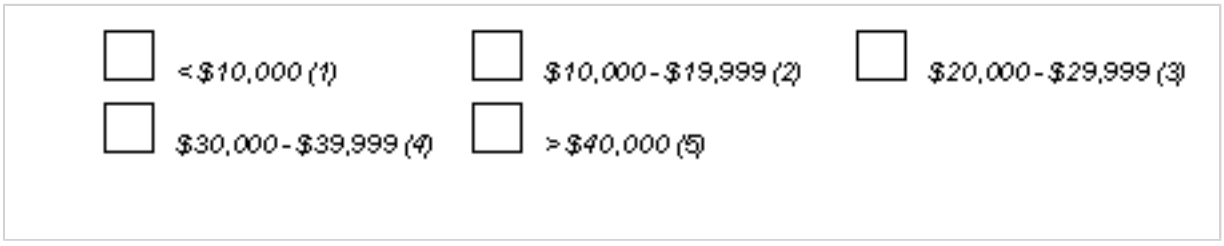
Example 9.3. Check field example

9.7.2. Choice

A choice field consists of two or more response options each represented by a box 1/8" to 1/4" square, where only one box may be checked. A maximum of 98 choice boxes may be used for each choice field.

The response options may be laid out horizontally, vertically or in a grid as illustrated below. Remember to leave 1/4" white space before each box, and to position choice labels to the right of each box.

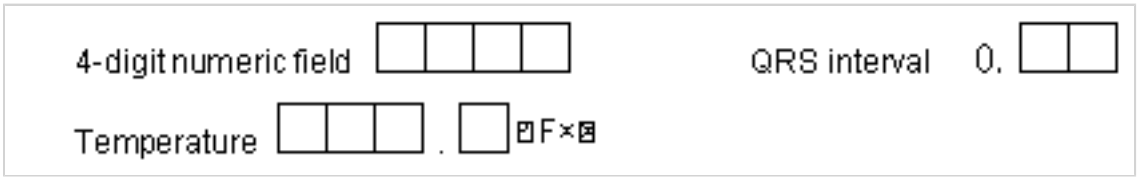
Example 9.4. Choice field example



9.7.3. Numeric

Numeric data fields are used to capture integer and real numbers.

Example 9.5. Numeric field examples



In some cases you may want to pre-print fixed values into some or all of the boxes of a numeric field. Avant Garde-Book, 18-point is recommended for maximum ICR accuracy, with CG Omega or Century Gothic being the next best choice. Make sure each digit is printed in the center of each box; do not touch box edges.

In addition to font type and point size, the following attributes should be specified for pre-printed numbers.

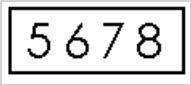
- Angle = Regular
- Weight = Regular

- Variation = Regular
- Color = Black
- Spread = 0.0%
- Stretch = 100.0%

ICR accuracy is particularly sensitive to the degree of stretch.

Pre-printed numbers can also be placed in a single bounding box as shown below.

Example 9.6. Pre-printed numbers in a single bounding box

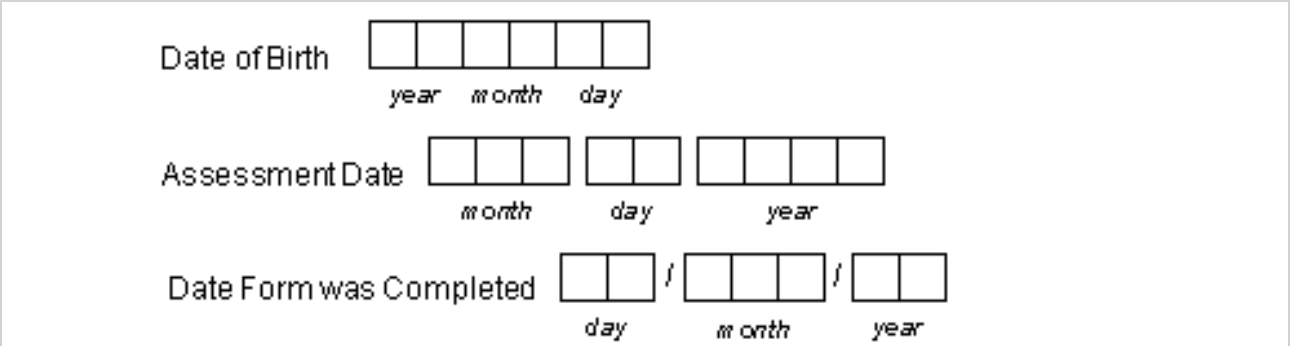


Accuracy is best if the box height is 1/3" and Avant Garde-Book, 18 point font is used with a spread of 15-20% to provide clear separation between the digits. Testing has shown this to be very accurately read by the ICR software with an error rate of less than 1 per 1000 data fields.

9.7.4. Date

Dates may be composed of a 2 or 4-digit year, a 2 digit or 3-character month, and a 2-digit day in any order. If characters are used for month only the English abbreviations, JAN, FEB, etc. may be used, and only uppercase is read by the ICR software. Each digit or letter of the day, month and year must appear in a separate box. Space or delimiters may appear between the day, month and year components as illustrated in the following examples.

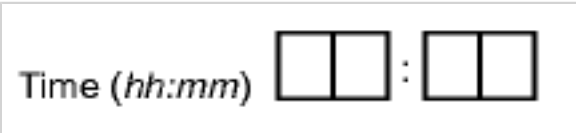
Example 9.7. Date field examples



9.7.5. Time

Times may include a 2-digit hour (24 hour clock), 2-digit minute, and optionally a 2-digit second field. Times must be in the range 00:00:00-23:59:59. Each digit must appear in a separate box. Colon delimiters may appear between the hour and minute component and the minute and optional seconds component.

Example 9.8. Time field example

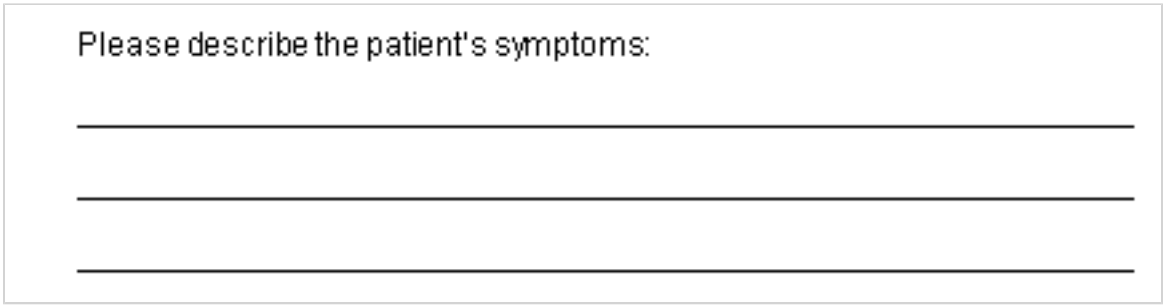


9.7.6. Text

Text fields have few restrictions except that the DFdiscover field delimiter | cannot be used and the total length of a data record including all fields and delimiters cannot exceed 16384 ASCII characters (or 4096 UNICODE characters). Text is never read by ICR but can be entered manually in **DFexplore**.

A text field is usually represented by one or more horizontal lines. The length of each line should be indicative of the maximum response length, and inter-line spacing should be .25" to .30". Long responses requiring several lines may be defined as multiple fields or as one field.

Example 9.9. Text field example



Text fields used to enter a small fixed number of characters should use either 0.25" square boxes or horizontal lines with half-height ticks as illustrated below. ICR does not read any text field including these.

Example 9.10. Fixed Text field example

9.7.7. Visual Analog Scales (VAS)

Visual analog scales can be used to collect a response indicated by a vertical line drawn by the subject across a horizontal scale. This is converted to a numeric value equal to the percentage of the scale range at which the response line was drawn, calculated from the left end. Small vertical tick marks may be placed at the ends and other points along the scale, but they must be smaller than the response line drawn across the scale.

Example 9.11. VAS field example

9.8. Printing

It is essential that each CRF page be printed without shrinkage, stretching or any other distortion. This is required both for the routing of CRFs to the correct study database (using the registration line and barcodes), and to maximize the accuracy of ICR. Failure to get the CRFs printed faithfully can defeat much of the automated processing that DFdiscover performs, and leave staff with more CRF pages to identify manually and more data entry to perform.

9.9. CRF Design Limits

The table provides a concise listing of DFdiscover limits and formats relevant to CRF design.

Table 9.1. Case report form design limits

Description	Limit	Comments
DFdiscover Study Number	1-255	The suggested range for study numbers is 1-249 as study numbers of 250-255 are reserved for DFdiscover test and validation studies (e.g. ATK = 254). With the appropriate license, it is possible to extend the range of study numbers to include 256-999, for EDC studies.
Plate Number	1-500	
Visit/Sequence Number	0-511	if included in the barcode
	0-65535	if included as the first data field on the plate (field #6)
Site ID	0-21460	This limit applies to the site ID only. A subject ID# may be concatenated to the site ID to obtain the fully-specified subject ID.
Subject ID	0-281474976710655	Subject IDs are (often) composed of site ID + subject #. In this case the limit applies to the concatenated value of the two. The Subject ID number must always be defined as field #7 using DFdiscover schema numbering. For presentation purposes, Subject Alias Map can be defined, expanding the possibilities.
Page Size	8.3-8.5" wide by 4.75-11" long	US Letter, European A4 and half page lengths are supported. Landscape orientation is not supported.
Page Margins	0.5” minimum on all edges	FDA Guidelines recommend a minimum of 1.0” margins.
Line Weight	1.0 point	A minimum line weight of 1.0 is recommended for data field boxes to minimize scanning resolution problems.
Box Size for Numeric and Date Fields	.20-.30"	Each box holds a single digit (for numeric fields) or digit/character (for date fields). For 3 character months use .25" square boxes to maximize ICR accuracy.
Box Size for Choice and Check Fields	.15-.25"	ICR requires square check boxes and suffers below .15" per side.
Number of Choice Boxes per Choice Field	2-98	Each check box should be preceded by at least .25" of white space.
Number of Check Boxes in a Check Field	1	
Length of Text Lines	maximum depends on page margins	e.g. 7.5" max for pages 8.5" wide with .5" margins
Vertical Spacing Between Text Lines	.25-.30”	.25" is the minimum for stacked text fields
Visual Analog Scale (VAS) length	maximum depends on page margins	by convention VAS scales are typically 10 cm long.
VAS Tick Marks	.125” maximum	vertical tick marks on a horizontal VAS may appear anywhere: ends, mid point, etc.

Appendix A. Appendix

A.1. Data Types

DFdiscover supports 7 data types:

- **Number.** Use the numeric type for both integer and fixed-point real numbers, signed or unsigned. When collecting CRFs by fax, data fields based on this data type can be read by DFdiscover's intelligent character recognition (ICR) module. Signed numbers require a special format for ICR (see Specific Properties for Numeric Fields). The largest permitted value for a numeric field, up to 10 digits (including the leading signs and the decimal points), is the maximum 32-bit unsigned number, or `2147483647`. Subject IDs (field 7) are 48-bit values, up to 15 digits long with a maximum 48-bit unsigned value of `281474976710655`. Floating-point real numbers and numbers in exponential notation must be stored as string data types.

- **Date.** Use the date type for calendar dates in year, month, day format. All 3 components are required but DFdiscover supports partial dates in which day or day and month are filled with zeros to indicate that these parts are unknown or irrelevant. The use of delimiters between the 3 components, e.g. '/' or '-', is optional and any ordering of year, month, and day is allowed. Day must be a 2-digit number, Month must be either a 2-digit number or a 3-character string, and Year must be either a 2-digit or a 4-digit number. Zero padding is required on each component where needed to meet the length requirements. Date fields consisting entirely of numeric components as well as those containing 3-character months, can be read by the ICR program. Allowable date formats are described in Specific Properties for Date Fields.

- **Time.** Time fields can store clock time (hours, minutes, and optionally seconds) using either the hh:mm or hh:mm:ss formats. Time fields can be read by the ICR program. For more information, see Specific Properties for Time Fields.

- **String.** String data fields are commonly used for comments, names, and descriptions. DFdiscover supports 2 types of string fields, unscannable text that is not read by the ICR program and numerals which are. A numeric text field is comprised of a string of digits only (0-9) in a single rectangular box. Such fields are subsequently treated as numbers for legal range testing and edit checks. The specification of such fields is described in Specific Properties for String Fields.

- **Choice.** The choice data type is used for fields comprised of 2 or more mutually exclusive options. They can be defined using a drop-down list of options, which ICR cannot read from a faxed CRF page, or as a set of check boxes where the user selects one and only one box. Choice fields that are comprised of check boxes can be read by the ICR program. A unique positive integer code (in the range 0-65535) and text label must be specified for each response option, and also for the case in which selection is made. DFdiscover supports a maximum of 98 response options per choice field. The specification of codes and labels is described in Specific Properties for Check & Choice Fields.

- **Check.** The check data type is used for fields defined by a single response option using a box that is either checked or left blank. For questions where the instruction is "Check all that apply" a separate check field is defined for each response option. Check fields are read by the ICR program. A unique positive integer code (in the range 0-65535) and text label is assigned to the check box, and to the case in which the box is not checked. The specification of codes and labels is described in Specific Properties for Check & Choice Fields.

- **VAS.** Visual Analog Scales are used for questions which ask the user to respond by marking their position along a line which represent some content dimension. Visual analog scales must have the following attributes on DFdiscover case report forms:

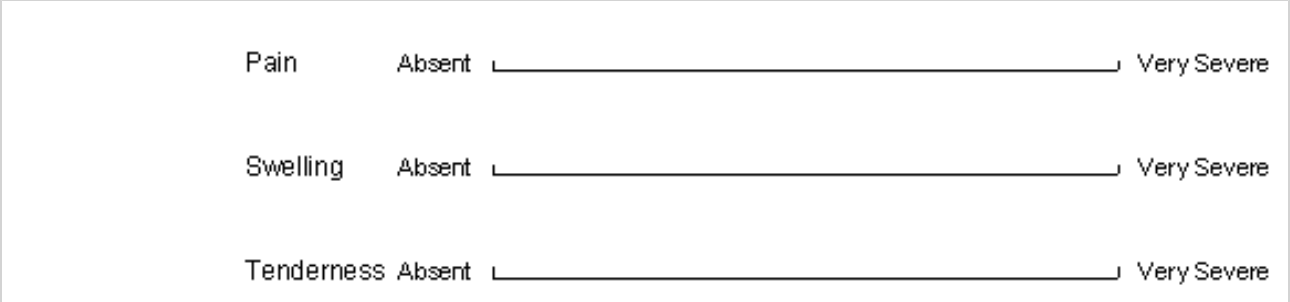
  - Single horizontal line

  - Point size for the line must be in the range 0.5pt to 1pt, preferably 0.5pt.

  - The two endpoints of the line may each be marked with a short 0.125" vertical line if desired.

Optional anchors may be added anywhere along the line using a short vertical mark. For example, an anchor might be included to mark the indifference point at the midpoint of an Agree/Disagree scale.

VAS Examples



These fields are typically scored from 0-100 representing the percentage of the distance along the scale at which the user's mark was made, but other scoring may be specified as described in Specific Properties for Visual Analog Scales.

VAS fields can be read by the ICR program. The ICR algorithm first measures the length of the VAS baseline and then looks for the user's mark across it. By using the percentage distance along the scale at which the mark occurs any stretching or shrinkage during scanning is accounted for. The ICR algorithm also accounts correctly for most user corrections by measuring the length of all cross marks and using the longest one. The ICR scoring can be verified in **DFexplore** by 3 clicks on the faxed image: left end, cross mark and right end.

A.2. Setup & Database Limits

The table lists DFdiscover study setup and database limits.

Table A.1. Setup & Database Limits

Description	Limit	Comments
Study Number	1-999	The recommended range is 1-249, numbers 250-255 may be used for DFnet demo and validation studies (e.g. ATK = 254). With the appropriate license, it is possible to extend the range of study numbers to include 256-999, for EDC studies.
Plate Number	1-500	All other plate numbers are reserved. Values currently in use include: 0=New record queue, 501=Query Reports, 510=queries, 511=reasons.
Assessment Number (barcoded)	0-511	
Assessment Number (1st data field)	0-65535	The assessment number must be defined as field 6 on all plates.
Site ID	0-21460	
Subject ID	0-281474976710655	Because IDs are typically composed of site# + subject#, this limit applies to the concatenated value of these two components. This field could contain 15 digits at maximum. The ID number must be defined as field 7 on all plates.
Plate Label	0-100 chars	This label is used to identify missing pages in Query Reports.
Query Category Code	30-99	Query category codes 1-6 and 21-23 are reserved for default DFdiscover categories.

Description	Limit	Comments
Query Category Label	1-20 chars	Query category labels must be unique.
Page Map Label	0-32 chars	This limit requires enabling 40 character variable descriptions in **DFsetup** Global Settings, otherwise the upper limit is 17 characters.
Visit Label	1-40 chars	This label is used to identify overdue visits in Query Reports.
Visit Acronym	0-8 chars	This limit requires enabling 40 character variable descriptions in **DFsetup** Global Settings, otherwise the upper limit is 6 characters.
Style Name	1-15 chars	Style names must be unique, are case sensitive, and may include letters, digits, underscores and spaces.
Alias	1-80 chars	Aliases must be unique across all study data fields.
Name	1-80 chars	Names are used in SAS jobs and should thus comply with your SAS version.
Field Description	1-40 chars	This limit requires enabling 40 character variable descriptions in **DFsetup** Global Settings, otherwise the limit is 25 characters.
Legal Range Definition	0-256 chars	This limit includes any spaces or commas in the legal range list.
String Field Length (store value)	~4000 chars	Limited by overall record length of 16384 ASCII characters (or 4096 UNICODE characters)
Numeric Field Values	-2147483647 to 2147483647	Numeric fields (except the Subject ID field) could contain 10 digits at maximum, which include the leading signs and decimal points.
Check Field Codes	0-65535	A code must be specified for “check on” and “check off”.
Choice Field Codes	0-65535	A code must be specified for each choice box and for no choice.
VAS Minimum and Maximum	-32768 to 32767	Typical visual analog scale values are 0-10 and 0-100.
VAS Precision	0 to 65535	Typically only 0-2 decimal places are used.
Date Field Length	6-64 chars	The minimum value of 6 would require a format without delimiters (e.g. yymmdd).
Edit check Call List	0-500 chars	This limit includes any spaces, commas and other characters that may be used in edit check parameter lists.
Maximum Data Fields per Plate	999	Includes 10 required fields, 7 at beginning and 3 at end of each record.
Maximum Length of a Data Record	16384 ASCII characters (or 4096 UNICODE characters)	Includes required fields at beginning and end of each record.
Fax Number	4096 chars	This limit includes all punctuation symbols, and applies to the entire list of fax numbers and email addresses that can be entered into this sites database field.
The '|' character		This character is used as the field delimiter in data records and thus cannot appear inside a data field.
Prompt	256 chars	

Description	Limit	Comments
Help	16383 chars	
Constant	~4000 chars	depends on overall length of data record as this constant value becomes part of a data record. Data records are limited to 16384 characters.
Units	40 chars	
Module Name	15 chars	
Module Description	80 chars	
Plate Arrival Trigger	32767 chars	
Table Name	30 chars	
Table Title	1500 chars	
Table Header	256 chars	
Table Custom Text	256 chars	
Instruction	1500 chars	

A.3. UNICODE Support

DFsetup supports UNICODE characters in the following widgets:

Global Settings Study Help,Workflow Labels

Field and Style Properties Description, Prompt, Constant value, Legal, Help, Skip condition, Codings labels, Units, Comment

Plate Properties Label

Order Fields Screen name

Edit checks Editor Edit check comments, message and text

Lookup Tables Editor Lookup table records

Sites Name, Contact, Address, Investigator

Missing Value Codes Code, Label

Query Category Map Label

Visit Map Acronym, Label

Page Map Plate/Visit Title, Label

CRF Type Map Label

CRF Background Map Comments

Conditional Terminations Description, Test Value (Has a value that is)

Conditional Visits Description, Test Value (Has a value that is)

Conditional Cycles Description, Test Value (Has a value that is)

Conditional Plates Description, Test Value (Has a value that is)

Query Titles	Query titles text
Query Covers	Query covers text
Query Messages	Query messages text

A.4. Meta-Words

The following meta-words can be used in style and field property specifications as described below:

Meta-Words available for Legal Value specifications

`today` `today` evaluates to the current calendar date from the system clock of the DFdiscover server. This meta-word is useful when defining a legal range for date fields, e.g. 12/25/2007-today

`$(choices)` `$(choices)` evaluates to the numeric codes for all possible response options (including no box checked) for check and choice fields and can thus be used to specify legal values for check and choice fields.

`$(ids)` `$(ids)` evaluates to the concatenated list of subject ID ranges from all sites listed in the sites database. This meta-word is typically used to specify the legal values property of the subject ID field.

Meta-Words available for Help specifications

`$(legal)` `$(legal)` evaluates to the list of legal values, defined by the Legal property, and can be used in help messages, e.g. Legal values are: $(legal)

Meta-Words available for Field Alias specifications

`$(field)` `$(field)` evaluates to the field number as displayed in the center of the field widget in the CRF window. It can be used to define field aliases, e.g. MEDHX$(field)

`$(plate)` `$(plate)` evaluates to the CRF plate number in 3-digit zero padded format. For example, `$(plate)` for CRF plate numbers 1, 55 and 101 will be 001, 055, 101.

`$(rplate)` `$(rplate)` also evaluates to the CRF plate number. It is the same as `$(plate)` except that the value is not leading zero padded.

Meta-Words available for Skip specifications

`$(blank)` `$(blank)` evaluates to the code that indicates no box was checked for check and choice fields, and to a null string for all other field types. This meta-word can be used in the 'Skip if value =' specification to indicate that the specified number of fields are to be skipped when the current field is blank, and `!$(blank)` can be used to skip fields when the current field is not blank.

`$(legal)` `$(legal)` evaluates to the list of legal values, defined by the Legal property. It can be used to indicate that fields are to be skipped if the current field contains one of these values, and `!$(legal)` can be used to skip fields when the current field does not contain one of these values. This meta-word cannot be combined with any other values when constructing a skip specification.

A.5. Conditional Tests

The conditional dialogs can be used to specify conditions under which cycles, visits and plates become required, optional or excluded, and under which subject follow-up is terminated.

Conditions override the requirements specified in the visit map. If more than one condition is met for the same cycle, visit or plate, and the conditions differ in their result, the last condition defined in the conditional dialog wins.

The same rules apply to the specification of conditions in all 4 conditional maps: termination, cycle, visit and plate.

Each condition starts with an IF statement followed by 0 or more AND statements. A condition that is only true when more than one test is met can be specified by including as many AND statements as required. More than one condition can be defined for the same termination event thus OR statements are not necessary.

The Visit component of each IF and AND statement may be specified using: a single visit number, a comma delimited list of visit numbers and ranges, or an asterisk (*) for all visits.

The Plate and Field components of each IF and AND statement must identify a specific data field and thus must contain a single value.

The table shows the types of tests that may be specified.

Test Type	Example Test Value(s)	The Test Complement
Numbers		
- a single value	42	!42
- a value list	42-44,49,55-59,111	!42-44,49,55-59,111
- negative values	-4~-1,1~4	!-4~-1,1~4
- less than	<5	!<5
- greater than	>10	!>10
Dates		
- on a specified date	12/31/04	!12/31/04
- after a specified date	>12/31/04	!>12/31/04
- before a specified date	<12/31/04	!<12/31/04
Strings		
- field contains a specified string [a]	~AE	!~AE
- field begins with a specified string [a]	~^AE	!~^AE
- field ends with a specified string [a]	~AE$	!~AE$
- field contains specified string only [a]	~^AE$	!~^AE$
- field contains any character in the set [a]	~[ABC]	!~[ABC]
Special		
- field is blank	blank	!blank
- field contains a missing value code	missing	!missing

[a]This notation is not supported in Schedule View.

In general the test should be appropriate for the type of data field being tested (numeric, date or string). However, string tests can be performed on any data field. In such cases the field is first converted to a string. Thus for numeric fields leading zeros become significant when doing a string match, but not when doing a numeric test.

An exclamation mark, appearing as the first character in a value test, indicates that the test evaluates true when the test that follows the exclamation mark is not met. Note that the resulting complement of a specified test is not just all other numbers, dates or strings, but also includes blanks and missing values. For example, the test !<5 (not less than 5), will be met when the data field being tested contains 5, any number greater than 5, a missing value code, or is blank (because none of these values is a number less than 5).

A value list can only be specified for numeric fields, and only numbers can be included in the list. When a list of numbers is specified the test is met if the data field contains any of the listed values. If the value list contains all positive numbers,

ranges may be specified using either a tilde or a dash. But if a value list includes negative numbers the range operator must be a tilde. Within a value list the same range operator must be used for all ranges. If a tilde is used as one of the range operators, any dashes in the list are assumed to denote negative values.

When specifying test values for a date field, the values must be specified using the date format specified for that field in the study schema.

String matches are performed using UNIX general regular expressions as implemented in the awk programming language. The above table includes some of the more common examples. **Partial string matches using these regular expressions are not supported in Schedule View.**

A.6. DFengage Considerations

DFengage is an electronic patient reported outcome (ePRO) application fully integrated with DFdiscover, built on the WeGuide platform. It is installed from the Google Play Store (for Android) or Apple App Store (for iOS) on smartphones and tablets. DFengage uses the setup definition defined in DFsetup and role permissions and user accounts defined in DFadmin. This section describes various items to consider when setting up plates and visits that will be completed in DFengage. Refer to *System Administrator Guide, PermissionsDFengageDFengage Considerations* and *System Administrator Guide, DFengage Considerations* for defining roles and user permissions for DFengage.

- **Plate Setup.** Forms to be completed in DFengage may be set up as plates with or without CRF backgrounds. A plate in DFdiscover is referred to as a "task" in DFengage. During CRF development and database setup, consider how best to organize the data to be collected across one or more plates at visits.

- **Module Order.** The module instances set up on plates completed in DFengage must be in sequential order regardless of what modules (e.g., Demographics, Vital Signs) have been added to the plate. The module instance numbers must be organized in the correct order on the plate and numbered in order, starting with 1, 2, 3, etc. You can verify this in the Plates-Modules-Fields panel as well as in the CRF view main panel. To re-order the modules in the Plates-Modules-Fields panel, select the module to reorder and click the up or down arrow buttons at the bottom of the panel. To correct module instance number, select the module instance in the Plates-Modules-Fields panel, then change the Instance Number in the Module Properties panel on the right. DFengage displays fields first in module instance order and then by field number within that module instance. If the module instances are not number sequentially, DFengage will not display the fields in the intended order. If a module instance contains fields that are not sequential (meaning fields from a different module instance are mixed between fields in another module instance), then DFengage will order these questions differently from how they are displayed in DFsetup and DFexplore.

- **Field Setup.** DFengage displays one data field per screen. The Prompt text is displayed as the question text above the data field. If no Prompt is defined, no question text will be shown in DFengage. Fields using the check data type may be grouped to display them as a single field in DFengage. Fields of data types other than check that are grouped together will not be displayed on one screen in DFengage. DFengage currently only submits clean (final) data to the DFdiscover server. Records with missing required fields or illegal values cannot be submitted. Therefore, avoid defining legal ranges and make required fields only for those that will be completed for all subjects.

  - **Date and time fields.** Both the baseline and ePRO visit dates must use DD/MM/YYYY format. If the visit date at the baseline visit is captured using a different date format (e.g., DD-MMM-YYYY), use a separate hidden field and edit check to convert the visit date at the baseline visit to the correct date format required for DFengage. DFengage automatically shows the current date and current time (according to the DFengage user's device) to help prevent data entry errors, but the user can change the date/time if appropriate for the question.

  - **Choice and check fields.** Both choice and check field option buttons are displayed as circles, whether it's "select one response" or "select all that apply". Add instructions to the prompt to clarify this as needed. Up to 98 choices can be defined in a choice field. The longer the text or the higher the number of choices, the more scrolling needed.

  - **Number fields.** Use the format property to limit possible values entered in DFengage. DFengage will display a warning displayed if "0" is entered in a number field; cannot proceed to the next question with "0" entered. Alpha keys are

hidden from the keyboard, except for the decimal key which is the only symbol available for number fields. DFengage does not support signed numbers (where s or S is included in the format property to allow for + or -).

- **VAS fields.** Use VAS fields to collect numbers within specific limits (e.g., temperature) to restrict data to allowable values only. Use in place of number fields whenever feasible or practical. You can set up VAS fields with as much precision as needed, in the field properties.
- **Skip patterns.** Skips defined in the field properties are followed when the user clicks the "Next" or "Finish" button, to skip the number of fields defined based on the value selected or entered. Note that grouped fields are considered 1 field for DFengage, so when defining skip patterns, ensure that grouped fields are only counted as 1 field.
- **Required fields.** Fields defined as required must be completed by the subject in DFengage. The "Next" button in DFengage is disabled and the subject cannot proceed to the next question until they provide a response. Subjects may skip fields that are defined as optional.
- **Instructions.** Page help, field help, and the Instructions property are not currently supported in DFengage. Add instructions in DFengage using field prompts and choice and check field labels. For example, to provide instructions at the beginning of a set of questions, add an optional check field with the instructions included in the prompt.
- **Edit checks.** Edit checks are not currently supported in DFengage.

- **Page Map.** Use the Page Map to define the task labels that appear in DFengage. The Page Map is described in Page Map.
- **Translations.** Use the Translations configuration in DFsetup to define the available languages and provide translations for DFengage for the prompt and choice and check field labels. Currently task names only display in the default language and app instructions are in English only. The Translations configuration is described in Translations.
- **Visit Map.** The visit map defines the schedule for when the tasks become available in DFengage. Tasks become available in DFengage at 12:01am on the day the visit is due according to the clock on the DFengage user's device. When notifications are enabled by the DFengage user on their device, they will receive a notification at 8:00am on the day the task is due. The Visit Map is described in Visit Map. By default, if a subject does not complete a task in DFengage, the task will remain in the app until it is completed in DFengage or another application (i.e., **DFweb**, **DFcollect**, or DFexplore). To remove tasks that have not been completed within the required window, include a hidden field on each ePRO plate to indicate when that record has "expired." Using edit checks run in batch, identify which records have expired based on the study schedule and check the hidden field and add missing value codes to any required fields on the plate. This will remove the task from the subject's list in DFengage.

A.7. External Software Copyrights

DFdiscover software uses several third-party software components as part of its server side and/or client tools.

The copyright information for each is provided below. If you would like to receive source codes of these third-party components, please send us your request at <help@dfnetresearch.com>.

A.7.1. DCMTK software package

This software and supporting documentation were developed by

OFFIS e.V.
R&D Division Health
Eschereg 2
26121 Oldenburg, Germany

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of OFFIS nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.7.2. Jansson License

Copyright © 2009-2014 Petri Lehtinen <petri&digip.org>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A.7.3. Mimencode

Copyright © 1991 Bell Communications Research, Inc. (Bellcore)

Permission to use, copy, modify, and distribute this material for any purpose and without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies, and that the name of Bellcore not be used in advertising or publicity pertaining to this material without the specific, prior written permission of an authorized representative of Bellcore. BELLCORE MAKES NO REPRESENTATIONS ABOUT THE ACCURACY OR SUITABILITY OF THIS MATERIAL FOR ANY PURPOSE. IT IS PROVIDED “AS IS”, WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES.

A.7.4. RSA Data Security, Inc., MD5 message-digest algorithm

Copyright © 1991-2, RSA Data Security, Inc. Created 1991. All rights reserved. License to copy and use this software is granted provided that it is identified as the “RSA Data Security, Inc. MD5 Message-Digest Algorithm” in all material mentioning or referencing this software or this function. License is also granted to make and use derivative works provided that such works are identified as “derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm” in all material mentioning or referencing the derived work. RSA Data Security, Inc. makes no representations concerning either the

merchantability of this software or the suitability of this software for any particular purpose. It is provided “as is” without express or implied warranty of any kind. These notices must be retained in any copies of any part of this documentation and/or software.

A.7.5. mpack/munpack

Copyright © 1993,1994 by Carnegie Mellon University All Rights Reserved.

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Carnegie Mellon University not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Carnegie Mellon University makes no representations about the suitability of this software for any purpose. It is provided “as is” without express or implied warranty.

CARNEGIE MELLON UNIVERSITY DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL CARNEGIE MELLON UNIVERSITY BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

A.7.6. TIFF

Copyright © 1988-1997 Sam Leffler Copyright © 1991-1997 Silicon Graphics, Inc.

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics.

THE SOFTWARE IS PROVIDED “AS-IS” AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

A.7.7. PostgreSQL

Portions © 1996-2019, PostgreSQL Global Development Group Portions © 1994, The Regents of the University of California

Permission to use, copy, modify, and distribute this software and its documentation for any purpose, without fee, and without a written agreement is hereby granted, provided that the above copyright notice and this paragraph and the following two paragraphs appear in all copies.

IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE

SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATIONS TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

A.7.8. OpenSSL License

Copyright © 1998-2019 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: “This product includes software developed by the OpenSSL Project for use in .the OpenSSL Toolkit.” (http://www.openssl.org/)

4. The names “OpenSSL Toolkit” and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.

5. Products derived from this software may not be called “OpenSSL” nor may “OpenSSL” appear in their names without prior written permission of the OpenSSL Project.

6. Redistributions of any form whatsoever must retain the following acknowledgment: “This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org)”

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT “AS IS” AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (<eay@cryptsoft.com>). This product includes software written by Tim Hudson (<tjh@cryptsoft.com>).

A.7.9. Original SSLeay License

Copyright © 1995-1998 Eric Young (<eay@cryptsoft.com>) All rights reserved.

This package is an SSL implementation written by Eric Young (<eay@cryptsoft.com>). The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the following conditions are aheared to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (<tjh@cryptsoft.com>).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement: “This product includes cryptographic software written by Eric Young (<eay@cryptsoft.com>)” The word “cryptographic” can be left out if the rouines from the library being used are not cryptographic related :-).
4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: “This product includes software written by Tim Hudson (<tjh@cryptsoft.com>)”

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

A.7.10. gawk

GNU GENERAL PUBLIC LICENSE Version 2, June 1991

http://www.gnu.org/licenses/gpl-2.0.html

Copyright © 1989, 1991 Free Software Foundation, Inc.

51 Franklin Street, Fifth Floor,
Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

1. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The “Program”, below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

2. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

3. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.) These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

4. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.) The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

5. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

6. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

7. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

8. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

9. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

10. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

11. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally. NO WARRANTY

12. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

13. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.7.11. Ghostscript

The files in the base, psi, lib, toolbin, examples, doc and man directories (folders) and any subdirectories (sub-folders) thereof are part of GPL Ghostscript.

The files in the Resource directory and any subdirectories thereof are also part of GPL Ghostscript, with the explicit exception of the files in the CMap subdirectory (except “Identity-UTF16-H”, which is part of GPL Ghostscript). The CMap files are copyright Adobe Systems Incorporated and covered by a separate, GPL compatible license.

The files under the jpegxr directory and any subdirectories thereof are distributed under a no cost, open source license granted by the ITU/ISO/IEC but it is not GPL compatible - see jpegxr/COPYRIGHT.txt for details.

GPL Ghostscript is free software; you can redistribute it and/or modify it under the terms the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

GPL Ghostscript is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program so you can know your rights and responsibilities. It should be in a file named doc/COPYING. If not, write to the

Free Software Foundation, Inc., 59 Temple Place Suite 330, Boston, MA
02111-1307, USA.

GPL Ghostscript contains an implementation of techniques covered by US Patents 5,055,942 and 5,917,614, and corresponding international patents. These patents are licensed for use with GPL Ghostscript under the following grant:

Whereas, Raph Levien (hereinafter “Inventor”) has obtained patent protection for related technology (hereinafter “Patented Technology”), Inventor wishes to aid the the GNU free software project in achieving its goals, and Inventor also wishes to increase public awareness of Patented Technology, Inventor hereby grants a fully paid up, nonexclusive, royalty free license to practice the patents listed below (“the Patents”) if and only if practiced in conjunction with software distributed under the terms of any version of the GNU General Public License as published by the

Free Software Foundation, 59 Temple Place, Suite
330, Boston, MA 02111.

Inventor reserves all other rights, including without limitation, licensing for software not distributed under the GNU General Public License.

5055942 Photographic image reproduction device using digital halftoning to para images allowing adjustable coarseness 5917614 Method and apparatus for error diffusion paraing of images with improved smoothness in highlight and shadow regions

A.7.12. MariaDB and FreeTDS

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999 http://www.gnu.org/licenses/lgpl-2.1.html

Copyright © 1991, 1999

Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method:

1. we copyright the library, and

2. we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a “work based on the library” and a “work that uses the library”. The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

1. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called “this License”). Each licensee is addressed as “you”.

   A “library” means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

   The “Library”, below, refers to any such software library or work which has been distributed under these terms. A “work based on the Library” means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term “modification”.)

   “Source code” for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

2. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

3. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. The modified work must itself be a software library.

   b. You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

   c. You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

   d. If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful. (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

      These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

4. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

   Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

   This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

5. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

   If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

6. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

   However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a “work that uses the library”. The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

   When a “work that uses the Library” uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

   If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

   Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

7. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

   You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License.

Also, you must do one of these things:

a. Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b. Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that

   i. uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and

   ii. will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c. Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d. If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e. Verify that the user has already received a copy of these materials or that you have already sent this user a copy. For an executable, the required form of the “work that uses the Library” must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

f. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

   a. Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

   b. Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

g. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

h. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your

acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

i. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

j. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

k. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

l. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

m. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

   NO WARRANTY

n. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY

AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

o. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.7.13. QtAV

© Wang Bin <wbsecg1@gmail.com> Shanghai University->S3 Graphics->Deepin, Shanghai, China 2013-01-21

QtAV is free software licensed under the term of LGPL v2.1. The player example is licensed under GPL v3. If you use QtAV or its constituent libraries, you must adhere to the terms of the license in question.

Rather than repeating the text of the LGPL v2.1, the original text can be found in GNU LESSER GENERAL PUBLIC LICENSE, Version 2.1 [165].

A.7.14. FFmpeg

Most files in FFmpeg are under the GNU Lesser General Public License version 2.1 or later (LGPL v2.1+). Read the file `COPYING.LGPLv2.1` for details. Some other files have MIT/X11/BSD-style licenses. In combination the LGPL v2.1+ applies to FFmpeg.

Rather than repeating the text of the LGPL v2.1, the original text can be found in GNU LESSER GENERAL PUBLIC LICENSE, Version 2.1 [165].

A.7.15. c3.js

The MIT License (MIT) © 2013 Masayuki Tanaka

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A.7.16. d3.js

Copyright 2010-2017 Mike Bostock All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and

the following disclaimer in the documentation and/or other materials provided with the distribution. * Neither the name of the author nor the names of contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Index

R

S

T

U

V