Generating Natural Data Definition Modules (DDMs)

To enable Natural to access an SQL/DS table, a Natural DDM of the table must be generated. This is done either with Predict (see the relevant Predict documentation for details) or with the Natural utility SYSDDM; see also *SYSDDM Utility* in the Natural *Editors* documentation.

If you do not have Predict installed, use the SYSDDM function **SQL Services** to generate Natural DDMs from SQL/DS tables. This function is invoked from the main menu of SYSDDM and is described on the following pages.

For further information on Natural DDMs, see *Data Definition Modules - DDMs* in the Natural *Programming Guide*.

This section covers the following topics:

- SQL Services

SQL Services

The **SQL Services** function of the Natural SYSDDM utility (see *Using SYSDDM Maintenance and Service Functions* in the Natural *Editors* documentation) is used to access SQL/DS tables. You access the catalog of the SQL/DS server to which you are connected, for example, by using the CONNECT command of the *SYSSQL Utility* (see the section *Database Management*), or by entering the name of a server in the **Server Name** field on the **SQL Services Menu**. The name of the SQL/DS server to which you are connected is then displayed in the top left-hand corner of the screen SQL Services Menu. You can access any SQL/DS server that is located on either a mainframe (z/OS or z/VSE) or a UNIX platform if the servers have been connected via DRDA (Distributed Relational Database Architecture). For further details on connecting SQL/DS servers and for information on binding the application package (SYSDDM uses I/O module NDBIOMO) to access data on remote servers, refer to the relevant IBM literature.

The SQL Services function determines whether you are connected to a mainframe DB2 (z/OS or z/VSE) or a UNIX DB2, access the appropriate DB2 catalog and performs the functions listed below.

Note:
If you use SYSDDM **SQL Services** in a CICS environment without file server, specify CONVERS=ON in the NDBPARM module (see the relevant section in Installing Natural for SQL/DS); otherwise you might get SQL code -518.

- Using SQL Services
- Select SQL Table from a List
- Generate DDM from an SQL Table
- List Columns of an SQL Table

The individual functions are described below.

Using SQL Services

To invoke the SQL Services function

1. In the command line, enter the Natural system command SYSDDM and press Enter.

   Or:

   1. From the Natural main menu, choose **Maintenance and Transfer Utilities** to display the **Maintenance and Transfer Utilities** menu.
   2. From the **Maintenance and Transfer Utilities** menu, choose **Maintain DDMs**.

   The menu of the SYSDDM utility is displayed. The fields and functions provided on the SYSDDM utility menu are explained in the section *Using SYSDDM Maintenance and Service Functions*.

2. In the **Code** field of the Natural SYSDDM utility **Menu**, enter code B and press Enter.

   The **SQL Services Menu** is displayed.

```
14:43:41             ***** NATURAL SYSDDM UTILITY *****             2009-12-04
 Server DAVNDB2              - SQL Services: Menu -




                   Code  Function

                     S   Select SQL Table from a List
                     G   Generate DDM from an SQL Table
                     L   List Columns of an SQL Table
                     ?   Help
                     .   Exit

            Code ... _
      Table name ... ________________________________
      Creator ..... ________________________________
      Replace ..... N (Y,N)          DDM Name with Creator .. Y (Y/N)
      Server name .. DAVNDB2___________


Command ===>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit                                                  Canc
```

   The functions available on this screen are described in the corresponding sections.

Select SQL Table from a List

This function is used to select an SQL/DS table from a list for further processing.

To invoke the Select SQL Table from a List function

- On the **SQL Services Menu**, enter Function Code S.
  - If you enter the function code only, you obtain a list of all tables defined to the SQL/DS catalog.
  - If you do not want a list of all tables but would like only a certain range of tables to be listed, you can, in addition to the function code, specify a start value in the **Table Name** and/or **Creator** fields. You can also use asterisk notation (*) for the start value.

  Press Enter.

  The **Select SQL Table From A List** screen is invoked displaying a list of all SQL/DS tables requested. On the list, you can mark an SQL/DS table with a function code:

Code	Function	Description
G	Generate DDM from an SQL Table	This function can be used to generate a Natural DDM from an SQL/DS table, based on the definitions in the SQL/DS catalog.
L	List Columns of an SQL Table	This function lists all columns of a specific SQL/DS table.

Generate DDM from an SQL Table

This function is used to generate a Natural DDM from an SQL/DS table, based on the definitions in the SQL/DS catalog.

The following topics are covered below:

- Invoking the Generate DDM from an SQL Table function
- DBID/FNR Assignment
- Long Field Redefinition
- Length Indicator for Variable Length Fields: VARCHAR, LONG VARCHAR, VARGRAPHIC, LONG VARGRAPHIC
- Null Values

Invoking the Generate DDM from an SQL Table function

To invoke the function

- On the **SQL Services Menu**, enter function code G along with the name and creator of the table for which you wish a DDM to be generated.
  - If you do not know the table name/creator, you can use the function **Select SQL Table from a List** to choose the table you want.

- If you do not want the creator of the table to be part of the DDM name, enter an `N` (No) in the field **DDM Name with Creator** when you invoke the Generate function. The default setting is is `Y` (Yes).

Important:
Since the specification of any special characters as part of a field or DDM name does not comply with Natural naming conventions, any special characters allowed within SQL/DS must be avoided. SQL/DS delimited identifiers must be avoided, too.

- If you wish to generate a DDM for a table for which a DDM already exists and you want the existing one to be replaced by the newly generated one, enter a `Y` (Yes) in the **Replace** field when you invoke the Generate function.

- By default, **Replace** is set to `N` (No) to prevent an existing DDM from being replaced accidentally. If **Replace** is `N`, you cannot generate another DDM for a table for which a DDM has already been generated.

DBID/FNR Assignment

When the **Generate DDM from an SQL Table** function is invoked for a table for which a DDM is to be generated for the first time, the **DBID/FNR Assignment** screen is displayed. If a DDM is to be generated for a table for which a DDM already exists, the existing DBID and FNR are used and the **DBID/FNR Assignment** screen is suppressed.

On the **DBID/FNR Assignment** screen, enter one of the database IDs (DBIDs) chosen at Natural installation time, and the file number (FNR) to be assigned to the SQL/DS table. Natural requires these specifications for identification purposes only.

The range of DBIDs which is reserved for SQL/DS tables is specified in the `NTDB` macro of the Natural parameter module (see the Natural *Parameter Reference* documentation) in combination with the `NDBID` macro of the parameter module `NDBPARM`. Any DBID not within this range is not accepted. The FNR can be any valid file number within the database (between 1 and 255).

After a valid DBID and FNR have been assigned, a DDM is automatically generated from the specified table.

Long Field Redefinition

The maximum field length supported by Natural is 1 GB-1 (1073741823 bytes). If an SQL/DS table contains a column which is longer than 253 bytes, the pop-up window **Long Field Generation** will be displayed automatically.

A field which is longer than 253 bytes may be defined as a simple Natural field with a maximum length of 32KB-1, or as an array. In the DDM, such an array is represented as a multiple-value variable.

On the **Long Field Generation** screen you specify the element length of the array, which means the length of the occurrences. The number of occurrences depends on the length you specify.

If, for example, an SQL/DS column has a length of 2000 bytes, you can specify an array element length of 200 bytes, and you receive a multiple-value field with 10 occurrences, each occurrence with a length of 200 bytes.

Since redefined long fields are no multiple-value fields in the sense of Natural, the Natural C* notation cannot be applied to those fields.

When such a redefined long field is defined in a Natural view for being referenced by Natural SQL statements (that is, by host variables which represent multiple-value fields), both when defined and when referenced, the specified range of occurrences (index range) must always start with occurrence 1. If not, a Natural syntax error is returned.

Example:

```
UPDATE table SET varchar = #arr(*)
SELECT ... INTO #arr(1:5)
```

Note:
When such a redefined long field is updated with the Natural native DML `UPDATE` statement, care must be taken to update each occurrence appropriately.

Length Indicator for Variable Length Fields: VARCHAR, LONG VARCHAR, VARGRAPHIC, LONG VARGRAPHIC

For each variable length column, an additional length indicator field (format/length I2) is generated in the DDM. The length is always measured in number of characters, not in bytes. To obtain the number of bytes of a `VARGRAPHIC` or `LONG VARGRAPHIC` field, the length must be multiplied by 2.

The name of a length indicator field begins with "L@" followed by the name of the corresponding field. The value of the length indicator field can be checked or updated by a Natural program.

If the length indicator field is not part of the Natural view and if the corresponding field is a redefined long field, the length of this field with `UPDATE` and `STORE` operations is calculated without trailing blanks.

Null Values

With Natural, it is possible to distinguish between a null value and the actual value zero (0) or blank in an SQL/DS column.

When a Natural DDM is generated from the SQL/DS catalog, an additional `NULL` indicator field is generated for each column which can be `NULL`; that is, which has neither `NOT NULL` nor `NOT NULL WITH DEFAULT` specified.

The name of the `NULL` indicator field begins with `N@` followed by the name of the corresponding field.

When the column is read from the database, the corresponding indicator field contains either zero (`0`) (if the column contains a value, including the value `0` or blank) or `-1` (if the column contains no value).

Example:

The column `NULLCOL CHAR(6)` in an SQL/DS table definition would result in the following view fields:

```
NULLCOL          A 6.0
N@NULLCOL    I 2.0
```

When the field NULLCOL is read from the database, the additional field N@NULLCOL contains:

- 0 (zero) if NULLCOL contains a value (including the value 0 or blank),
- -1 (minus one) if NULLCOL contains no value.

A null value can be stored in a database field by entering -1 as input for the corresponding NULL indicator field.

Note:
If a column is NULL, an implicit RESET is performed on the corresponding Natural field.

List Columns of an SQL Table

This function lists all columns of a specific SQL/DS table.

To invoke the List Columns function

- On the **SQL Services Menu**, enter function code L along with the name and creator of the table whose columns you wish to be listed, and press Enter.

  The **List Columns** screen for this table is invoked, which lists all columns of the specified table and displays the following information for each column:

Variable	Content	
Name	The SQL/DS name of the column.	
Type	The column type.	
Length	The length (or precision if type is DECIMAL) of the column as defined in the SQL/DS catalog.	
Scale	The decimal scale of the column (only applicable if type is DECIMAL).	
Update	Y	The column can be updated.
	N	The column cannot be updated.
Nulls	Y	The column can contain null values.
	N	The column cannot contain null values.
Not	A column which is of a scale length or type not supported by Natural is marked with an asterisk (*). For such a column, a view field cannot be generated. The maximum scale length supported is 7 bytes. Types supported are: CHAR, VARCHAR, LONG VARCHAR, GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC, DECIMAL, INTEGER, SMALLINT, DATE, TIME, TIMESTAMP, and FLOAT.	

  The data types DATE, TIME, TIMESTAMP, and FLOAT are converted into numeric or alphanumeric fields of various lengths: DATE is converted into A10, TIME into A8, TIMESTAMP into A26, and FLOAT into F8.

For SQL/DS, Natural provides an SQL/DS `TIMESTAMP` column as an alphanumeric field (A26) in the format *`YYYY-MM-DD-HH.SS.MMMMMM`*.