

DB2 and SQL/DS

DB2 objects can be documented in Predict and generation, incorporation, comparison and administration functions can be applied to them.

Note:

To use functions of Predict that support DB2, Natural/DB2 must be installed. Most functions described in this section apply both to DB2 and SQL/DS. Exceptions to this rule are listed as appropriate.

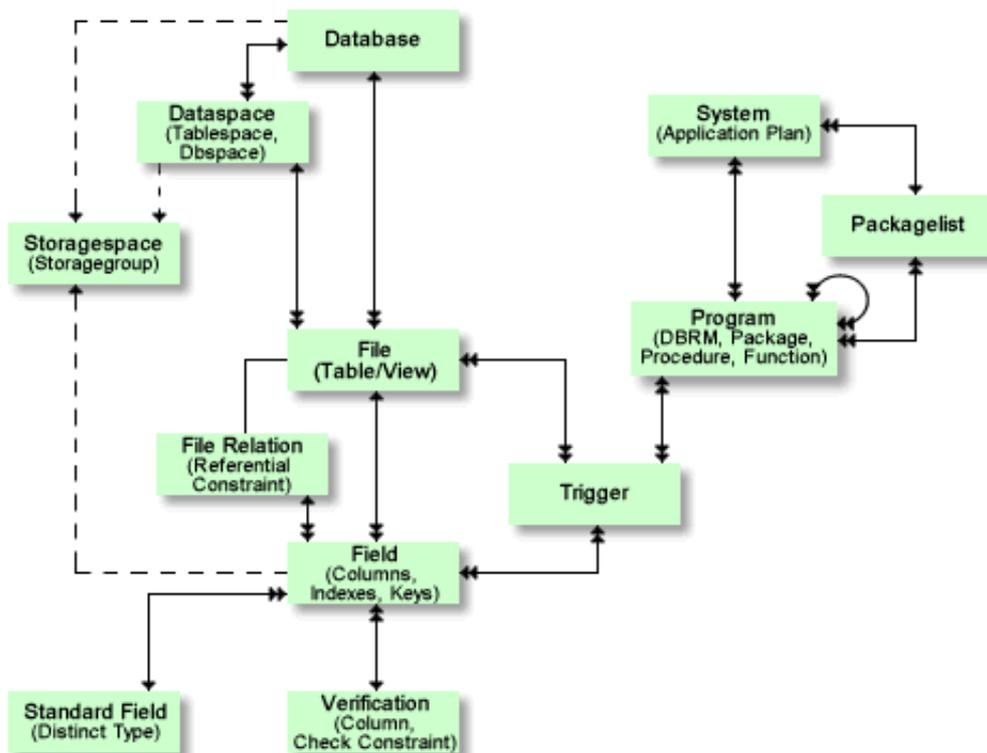
This section covers the following topics:

- Documenting DB2 in Predict
- Naming Conventions for DB2
- Generating, Incorporating and Comparing DB2 Objects
- Administrating Implemented DB2 Objects

Documenting DB2 in Predict

General Information

DB2 storagegroups, databases, tablespaces, tables, views, columns, distinct types, indexes, referential constraints, triggers, packages, application plans, procedures and functions can be documented in Predict.



The following table gives an overview of how different DB2 objects are documented.

DB2 Object	Documented in Predict with
Database	Database object of type D
Storagegroup	Storagespace object
DB2 tablespace /SQL/DS Dbspace	Dataspace object
Table / view	File object of type D or E
Table check constraint	Attribute of file object (type D)
Column check constraint	Verification object
Application plan	System object of type P
DBRM	Program object of type P and language Q
Package	Program object of type P and language B, C, F, H, P or Q or user-defined. Packages are linked to application plans with objects of type packagelist.
Collection	Packagelist attributes Collection name and Location name.
Index	Field attributes.
Column	Field objects.
Distinct Type	Field object of standard file SAG-DISTINCT-TYPE.
Trigger	Trigger object.
Proccedure	Program object of type R.
Function	Program object of type U.

Documenting DB2 objects is described in the sections below.

Documenting DB2 Storagegroups

Storagegroups are documented as objects of type storagespace.

Documenting DB2 Databases

Databases are documented as objects of type database with database type D.

A database of type D has a flag determining the kind of database. Two types are distinguished:

- DB2 databases that can be implemented in DB2 by issuing a CREATE DATABASE statement.
- SQL/DS databases that can be addressed with a CONNECT statement.

Only files of type D (DB2 table) can be linked to databases of type D.

Documenting DB2 Tablespaces and SQL/DS Dbspaces

In DB2/SQL/DS, tables/views are not directly linked to databases: a DB2 tablespace or SQL/DS Dbspace establishes the connection of tables/views and databases.

DB2 tablespaces and SQL/DS Dbspaces are documented with Predict dataspace objects of type D (DB2) or S (SQL/DS).

Note:

DB2 tablespaces need only be documented with Predict dataspace objects if you intend to generate the DB2 tablespace from the Predict dataspace object. If you use the option to create the DB2 tablespace implicitly when generating tables/views, the tablespace need not be documented with a Predict dataspace object.

Partitioned or segmented tablespaces are not created implicitly.

A SQL/DS Dbspace must be documented with a Predict tablespace object because a Dbspace cannot be created implicitly.

No auxiliary tablespaces are supported. See Columns With Format LOB for further information.

Documenting DB2 Tables and Views

Tables are documented as files of type D. Views are documented as files of type E.

Note:

If a table contains a partitioning index, the number of partitions must be documented as an attribute of the file if the file is not linked to a dataspace via association *Contains FI*.

Subselect Clauses and Expressions in field Definitions

The documentation of views is supported by an additional editor to specify the part of the subselect clause starting from the first FROM clause.

The selection clause of the subselect clause is documented by the specification of the field list of the view. The specified list of tables/views in the first FROM clause of the subselect clause is generated by Predict and will be updated if a field from an additional table/view is added to the view. Correlation names can be added to the tables and views in the list (using editor functions). The remaining part of the subselect clause is left unchanged.

The expression used to define DB2 or SQL/DS fields can contain complex expressions. Fields that are defined not only by a single column name but use either a constant or a more complex expression are called derived fields. A special editor is provided for specifying the expression of derived fields.

In the field expression and in the subselect clause, comment lines (lines starting with * or **) and remarks within a line (starting with /*) are allowed.

Beginning with version 5, DB2 offers the possibility to use subqueries and joined tables in the FROM clause of the view definitions. This functionality is also supported under Predict.

You can define in the view whether the FROM clause is made up of a INNER, LEFT OUTER, RIGHT OUTER, or FULL OUTER JOIN. Then you indicate both masterfiles that are to be joined. The syntax for this is already generated in the Subquery Editor (FROM <master file name 1><join type> JOIN <master file name 2> ON). You can add the Join condition to the text.

This text is provided with protection characters, which means that you can only add correlator names, but you cannot change the Join type. In order to change the Join type or the master files, you must clear the editor of its contents and then enter the FROM clause and search condition of your choice. Then you can only use the fields of the two master files in the view with Join.

Predict has a new file of type IV (Intermediate View) that enables you to use subselects in the FROM clause of a view definition. Files of type IV have a field list to show the selection clause and a subquery to show the search condition. Just like views, files of type IV can have a Join type. They can be used as master files for views and files of type IV. When a view is generated that has a file of type IV as a master file, a SELECT partial statement is generated into the FROM clause. Files of type IV are not in the DB2 catalog. Nevertheless, the same naming standards are valid for files of type IV as for tables and views. See Naming Conventions for DB2 for further information.

When using Incorporate and Compare, files of type IV are created in addition to the view in order to represent subselects in the FROM clause. Files of type IV are implemented as file objects in Predict, in order to ensure the consistency of the field definitions via rippling.

The number of master files for a view or file of type IV has been raised to 100 in Predict.

Documenting Referential Constraints

Referential constraints are documented as file relations of type R (referential constraint). The relation is established between a unique constraint and a foreign key. Unique constraint and foreign key can belong to the same or to different tables.

Documenting DB2 Application Plans

DB2 application plans are documented as Predict system objects of type P.

Documenting DB2 Packages

With DB2 Version 2.3 or above, DBRMs can be grouped to packages. Packages are linked to plans dynamically (at run-time). DB2 packages are documented in Predict with program objects of with language B, C, F, H, P or Q or user-defined.

Linking Packages to Application Plans with Packagelists

Packages are linked to application plans with objects of type packagelist (PG). The subtypes of the object type packagelist determine the type of inclusion of packages into an application plan when binding a plan. Valid values are:

Q
DBRM,

T
total collection and

S
subcollection.

The different subtypes are described in the section Using Predict Information when Binding Application Plans.

Packages are referenced in a plan by collections. Any collection is a virtual summary of packages, used to simplify references to packages. Any package can be contained in several collections. Collections are documented as attributes of packagelists.

Documenting DB2 Triggers

Trigger objects represent DB2 triggers.

You can link an unlimited number of triggers to a table. Update triggers that can only be executed if certain table fields are changed are then linked to these fields. The appropriate update-clauses are created during the table generation process.

Using Incorporate and Compare on the tables creates the trigger objects in Predict and establishes the links to the file, or to the files, whichever is relevant. The connection to the DB2 catalog is not the Predict object ID, but an attribute Triggername. See Naming Conventions for DB2 for further information.

In addition, triggers contain information as to when they are executed (during Insert, Update or Delete) and whether they are to be executed before or after a certain statement is executed. The code that is to be executed is noted in the trigger body, which is a text attribute in Predict. The use of procedures in triggers is retrieved from the text of the trigger body.

Documenting DB2 Procedures and Functions

DB2 gives you the possibility to write procedures and user-defined functions. These objects can be documented as programs in Predict. The program type R (SQL procedure) has been modified to accept many specifications that are only related to DB2. Procedures can be implemented in third generation programming languages or SQL.

Program type U (Database function) is a new feature of Predict. These functions can return a table as a result. There are two new associations between PR and FI with the names *Input FI* and *Returns FI*. The association names are supposed to indicate that the linked files represent the structure of the input parameter or the structure of the results table.

Only files of type IT (Intermediate Table) can be linked (see below). If the entered value or the results table is a scalar value, then you still must create a file that has exactly one field. In DB2, you can use table functions in the definition from views. Predict does not support this.

Files of Type IT

There is a new file type IT (Intermediate Table) for documenting the formal parameters of database functions. They do not exist in DB2. Their fields can, like tables, have a link to the standard file SAG-DISTINCT-TYPE, that is interpreted as being distinct type.

Documenting Other DB2 Objects

DB2 Distinct Types

In Predict, distinct types are implemented with the help of an indicated standard file called SAG-DISTINCT-TYPE.

The connection with the DB2 catalog is established by the names of the fields of this standard file. The field names of SAG-DISTINCT-TYPE consist of a SCHEMA_NAME and a TYPE_NAME. See Naming Conventions for DB2 for further information. Table fields that are connected to a standard field in SAG-DISTINCT-TYPE have the predefined format which is the basis for the type. Changes in the type definition are spread via rippling to the derived fields.

Note: The name of the standard field is not a valid column name. After copying from SAG-DISTINCT-TYPE via the SEL command into the field list of a table, the field name must be changed so that it conforms with the SQL naming standards.

When a table is generated, a CREATE DISTINCT TYPE statement is created for every type in the fields with distinct type, if the type has not already been defined in the DB2 catalog. The type definition of the table field is in this case the distinct type name.

When using the commands Incorporate and Compare on the table, the connections to the type definition are also compared. If the type definition in Predict is different from the one in DB2, then the field format of the table field is adapted to the catalog entry, and the field is marked as NON-Standard.

The type definitions are always regarded by the tables using it. Therefore, there are no explicit Generate, Incorporate and Compare distinct type functions. A type definition in DB2 is also deleted via Drop if the last table using it is dropped by Administration (Database, dataspace or file). Since every distinct type is based on a predefined type, the table fields derived from these types are represented in the DDM with the predefined type.

DB2 Columns

DB2 columns are documented as field objects.

Columns With Format LOB

LOBs are represented as the field format LO. The character set determines whether it is a BLOB, CLOB or a DCLOG. The lengths of these fields can be declared in the following units: bytes, kilobytes, megabytes or gigabytes.

In order to create a connection between a row in the original table and a LOB value, DB2 uses so-called auxiliary tablespaces, auxiliary tables and an index to save the LOB values. Predict does not support the creation of these database objects.

Instead Predict uses the DB2 feature that automatically create these auxiliary objects. This is achieved by generating a statement SET CURRENT RULE='STD' whenever a table with LOB column is to be generated, provided that the Special Register Current Rule does not already have this setting.

Columns With Format ROWID

ROWID fields are documented with fieldtype QN. Their format is A and their length is 40. The field maintenance ensures that only one ROWID field exists per table. It also ensures that every table containing a LOB column also contains a ROWID column. This is necessary, so that DB2 can create the index to connect the row in the original table with the auxiliary table.

When deleting databases, tablespaces and tables, the auxiliary objects are deleted as well. It is possible to define an identity property for numeric fields. The contents of these fields can be generated by DB2. This is an easy way to create a primary key.

LOB fields are skipped during the DDM generation process, since NAT 3.x does not make any dynamic variables available. ROWID fields are only represented by A40 fields in the DDM.

DB2 Indexes

DB2 indexes are documented with field objects as follows:

- Field attributes
 - index name
 - definition of index
 - using- and free-block
- If the index consists of only **one** column, the field documenting the column is marked as a descriptor with descriptor type D, P or F.
- If the index consists of **multiple** columns, it is documented as a field with field type SP (superfield) and descriptor type D, P or F. The descriptor types have the following meaning:
 - **D**
Field is an index.
 - **F**
Field is a foreign key and an index.
 - **P**
Field is a primary key. This always implies that the field is also a unique index.

Unique Constraints

Unique constraints are documented as follows:

- If the unique constrain applies to only **one** column, the field documenting the column is marked U in column Unique option.
- If the unique constraint applies to **multiple** columns, it is documented as a field of type SP (superfield) with

descriptor type D, F or P, and U in column Unique option. The descriptor types have the following meaning:

- **D**
Field is a unique index.
- **F**
Field is a foreign key with unique index.
- **P**
Field is a primary key. This always implies that the field has a unique constraint.

Foreign Keys

Foreign keys are documented as follows:

- If the foreign key consists of only one column, the field documenting the foreign key is marked as a descriptor with descriptor type F or E.
- If the foreign key consists of multiple columns, it is documented as a field with field type SP (superfield) and descriptor type F or E. The descriptor type F means the field is a foreign key and an index. E means the field is a foreign key without an index.

Column Check Expressions

Check expressions for single columns are documented with verifications of status SQL. The check expression is stored as the rule of the verification.

Check expressions can be edited with the Predict Rule Editor.

Comment lines (lines starting with * or **) and remarks within a line (starting with /*) are allowed.

Table Check Expression

A table check expression is a check expression that applies to more than one column. A table check expression is an attribute of a file.

To edit table check expressions, enter Y in the field EDIT: Trigger of the corresponding file object.

Comment lines (lines starting with * or **) and remarks within a line (starting with /*) are allowed.

Naming Conventions for DB2

DB2 naming conventions must be observed when creating or maintaining Predict objects for DB2. The following rules apply:

- Valid identifiers are from 2 to 27 characters long, must start with an alpha character (A - Z) and may be followed by either an alpha, US national character (#, \$, @), a digit or underscore character. Identifiers must comply with these rules if Predict maintenance and generation functions are to be applied.
- The identifier of a table, view, index or field of file SAG-DISTINCT-TYPE (representing distinct types) must be given in qualified form: the creator name (maximum length 8 characters), a delimiter and the table/view/index name (maximum length 18 characters). A hyphen is used as a delimiter (not a period as in SQL). Example: SYSIBM-SYSCOLUMNS. Hyphens in names are treated as follows:
 - When a table/view is generated from a Predict file object, the hyphen will be transformed into a period (.).
 - Because hyphens are used as delimiters, only one hyphen can occur in the file ID. Column names must not contain a hyphen.
 - The hyphen can be used as a minus sign or negative sign in the field expression or the subselect clause and must then be preceded by a blank.

Correlation Names

Correlation names can be defined in the subselect clause of a view. If a correlation name is defined for a table/view in the subselect clause, all references (in field expressions as well as in the field editor of the view) to columns of the table/view must be qualified with the correlation name. If no correlation name is defined for a table/view in the subselect clause, all references to columns of the table/view must be fully qualified with creator-tablename-columnname (for example: SYSIBM-SYSCOLUMNS-COLNAME).

Distinct Types

Distinct types consist of SCHEMA_NAME and TYPE_NAME concatenated by a hyphen (used as qualification character).

Procedure Name

Procedure names must be defined in unqualified form (a long SQL identifier).

Index Names

Index names consist of the creator name and the index concatenated by a hyphen (used as qualification character).

Function Name

Function names must be defined in unqualified form (a long SQL identifier).

Trigger Names

Trigger names consist of the creator name and the Trigger_Name concatenated by a hyphen (used as qualification character).

Delimited Identifiers

With DB2 or SQL/DS, special characters can be used in identifiers of tables and views. Identifiers that contain special characters have to be delimited (usually with single or double quotes) and are therefore called delimited identifiers.

DB2 or SQL/DS tables and views with delimited identifiers can be incorporated. They can then be renamed with Predict maintenance functions and retrieval functions can be applied to them. It is strongly advisable to rename delimited identifiers for the following reasons:

- The only Predict functions that can be applied without restriction to objects with delimited identifiers are Incorporate and Rename.
- If identifiers contain special characters such as blank or asterisk, results of retrieval functions are unpredictable.
- Views can only be generated if the subselect clause and the column expressions do not contain references to delimited identifiers enclosed by quotation marks.

As the SQL escape character Predict uses quotation marks (") and as the SQL string delimiter apostrophes (') are used. The Predict Incorporate function converts other escape characters or string delimiters to quotation marks (") and apostrophes (').

Generating, Incorporating and Comparing DB2 Objects

Prerequisites

Generation, Incorporation and Comparison are subject to DB2 security mechanisms:

- To perform the Generate function and administration functions Purge and Refresh, the user must have the appropriate privileges within DB2 / SQL/DS.
- To perform Incorporation and Comparison functions, the user must have SELECT privilege on nearly all catalog tables.
 - The SELECT privilege in DB2 is the minimum prerequisite for incorporation of DB2 tables/Views (see **GRANT (TABLE or VIEW PRIVILEGES)** in your DB2 documentation for the following tables:

SYSIBM-SYSCHECKDEP
 SYSIBM-SYSCHECKS
 SYSIBM-SYSCOLUMNS
 SYSIBM-SYSDATABASE
 SYSIBM-SYSFIELDS
 SYSIBM-SYSFOREIGNKEYS
 SYSIBM-SYSINDEXES
 SYSIBM-SYSINDEXPART
 SYSIBM-SYSKEYS
 SYSIBM-SYSPLAN
 SYSIBM-SYSSTOGROUP
 SYSIBM-SYSRELS
 SYSIBM-SYSSYNONYMS
 SYSIBM-SYSTABLEPART
 SYSIBM-SYSTABLES
 SYSIBM-SYSTABLESPACE
 SYSIBM-SYSVIEWDEP
 SYSIBM-SYSVIEWS
 SYSIBM-SYSVOLUMES
 SYSIBM-SYSDATATYPES
 SYSIBM-SYSTRIGGERS

- To use the SQL statements generated by Predict, the corresponding DB2 privileges are also required.
- To incorporate the SQL/DS tables contained in the following list, the SELECT privilege in SQL/DS is also a prerequisite:

SYSTEM-SYSCATALOG
 SYSTEM-SYSCOLUMNS
 SYSTEM-SYSDBSPACES
 SYSTEM-SYSINDEXES
 SYSTEM-SYSKEYCOLS
 SYSTEM-SYSKEYS
 SYSTEM-SYSSYNONYMS
 SYSTEM-SYSUSAGE
 SYSTEM-SYSVIEWS

Generation

DB2 objects can be generated from Predict documentation objects.

DB2 database	The function is not available for SQL/DS. Command: GENERATE DB2-DATABASE
DB2 storagegroup	Command: GENERATE STORAGEGROUP
DB2 tablespace / SQL/DS Dbspace	Command: GENERATE TABLESPACE
DB2/SQL/DS table/view	Columns, indexes, referential constraints, triggers and distinct types are automatically included when generating DB2 tables and views. Command: GENERATE TABLE

Rules Applying when Generating DB2 / SQL/DS Objects

- All objects are generated by first generating the SQL statements that are necessary to implement the object and then executing these statements.
An additional confirmation is requested before a DB2 object is actually implemented.
- The generated SQL statements can be saved in a protocol.
- If a generation function is executed for an object that is already implemented, the existing DB2 object can be updated.
- Tables with LOB column: Statement SET CURRENT RULE='STD' is created when a table with LOB column is generated.
- Auxiliary tablespaces, tables and indexes are created automatically by DB2.

See respective sections in Generation in the **External Objects in Predict documentation** for more information.

Incorporation

The incorporation functions create Predict documentation objects for databases (not for SQL/DS), tablespaces, storagegroups, tables/views, (including columns, indexes and referential constraints) from the system catalog.

See the section Incorporation in the **External Objects in Predict documentation**.

Comparison

The comparison functions list differences between the current implementation in DB2 / SQL/DS and the corresponding documentation. The documentation can be updated to match the implementation.

See the section Comparison in the **External Objects in Predict documentation**.

Administrating Implemented DB2 Objects

Functions for administrating DB2 objects are provided to display, purge or refresh DB2 objects that have been implemented from Predict documentation.

- The Display function lists generation protocols.
- The Purge function drops a table/view physically in DB2 or SQL/DS. If a table holds the last reference to a distinct type, the distinct type is also dropped.
- The Refresh function deletes all data in an implemented table but keeps the table structure.

See the section Administration of External Objects in the **External Objects in Predict documentation**.

Locking the Functions of the DB2 Utilities SYSDDB2 and SYSSQL

With the Natural for DB2 utilities SYSDDB2 (for DB2) and SYSSQL (for SQL/DS) storagegroups, databases, tablespaces/Dbspaces, tables/views and indexes can be created or modified. To avoid undocumented changes to DB2 or SQL/DS concerning these object types, your data dictionary administrator (DDA) may have set the parameter SYSDDB2 utility in the Defaults > General Defaults > Protection screen.

A	Allowed: all SYSDDB2 functions can be executed.
D	Disallowed: the following SYSDDB2 functions cannot be executed: CREATE DATABASE CREATE STORAGEEGROUP CREATE TABLE CREATE TABLESPACE CREATE VIEW CREATE INDEX
I	Incorporate: all SYSDDB2 functions can be executed outside of Predict. If one of the following statements is submitted to DB2, an automatic incorporation in Predict is performed: CREATE DATABASE CREATE STORAGEEGROUP CREATE TABLE CREATE TABLESPACE CREATE VIEW