

Using PAC with Predict Case or Natural for DB2

This chapter covers the following topics:

- Predict Case
 - Natural for DB2
-

Predict Case

PAC provides the following integration with Predict Case:

- Natural object lists can be generated from a Predict Case object set.
- PAC verifies that Natural object versions are synchronized with the corresponding Predict Case object.
- PAC identifies the Predict Case object versions that were used to compose the actual source of the Natural object and maintains a usage list of them.

Establishing PAC in a Predict Case Environment

When PAC is started, it uses the Entire DB files defined for Predict Case by the NATPARM NTFILE parameter to establish direct communication with Predict Case.

Predict Case must be installed with NTFILE values 249 and 250 established as a valid Predict Case/Entire DB environment.

Generating a Natural Object List

For Predict Case applications, you must specify a Predict Case library in application status link definitions that have an origin status of either "development" or "maintenance". PAC communicates with Predict Case to verify the relationship between the objects residing in the Natural library and the objects defined to the Predict Case library used to compose them.

Although the Predict Case library name can be specified at any time, it is validated only when an event with a development or maintenance origin status is validated and authorized, and again during the actual migration of the objects from development or maintenance.

The Predict Case library contains the Predict Case object set, which is identified by set name and user ID in the PAC Generate List option.

You can create a Natural object list for the Predict Case application by

- manually entering the objects;
- generating the list from a Predict Case object set (GENTYPE C);
- comparing the date-time stamps of objects in the origin and destination statuses and including the objects that are different (GENTYPE P).

In all cases, the Natural objects are compared with the Predict Case library during validation to verify that the Natural objects are consistent with the Predict Case objects used to compose them.

Any inconsistencies detected during validation produce an error message. You can then either remove the object from the list or correct the inconsistency before migrating the object.

As with all PAC migrations, the object list is validated again at execution time. If any inconsistencies are detected at this time, the object is automatically rejected for the migration.

Natural Object List Validation

During the validation of the object list, when it is detected that a particular application status link has an associated Predict Case library, PAC verifies that all objects being processed (except those being rolled) were composed by Predict Case.

Note:

Rules and views are processed normally since Predict Case does not treat these objects in any special manner.

Objects for which an error condition is returned are not migrated.

PAC checks the following for each object in the object list and issues an error if all conditions are not verified:

- The Natural object has corresponding Predict Case source objects from which it was composed. The corresponding Predict Case objects that composed the Natural object must still exist.
- The Natural objects have the save/stow date recorded on the Predict Case object.
- None of the Predict Case objects used to compose the Natural source object has been changed since the Natural object was last composed.

Information Stored by PAC

For auditing purposes, PAC stores the following information about the Predict Case objects used to compose the Natural objects migrated:

- A list of the Predict Case objects used to compose each Natural object as part of that Natural object's usage list;
- The Predict Case source library name recorded in the directory information for the Natural object;
- The date-time stamp of the Predict Case object as it is stored in Predict Case;
- A list of all Natural objects that used a particular Predict Case object to compose.
- The following information about each Predict Case object referenced by a Natural object:
 - Object type
 - Abstract
 - Date created
 - User who created
 - Date modified
 - User who modified

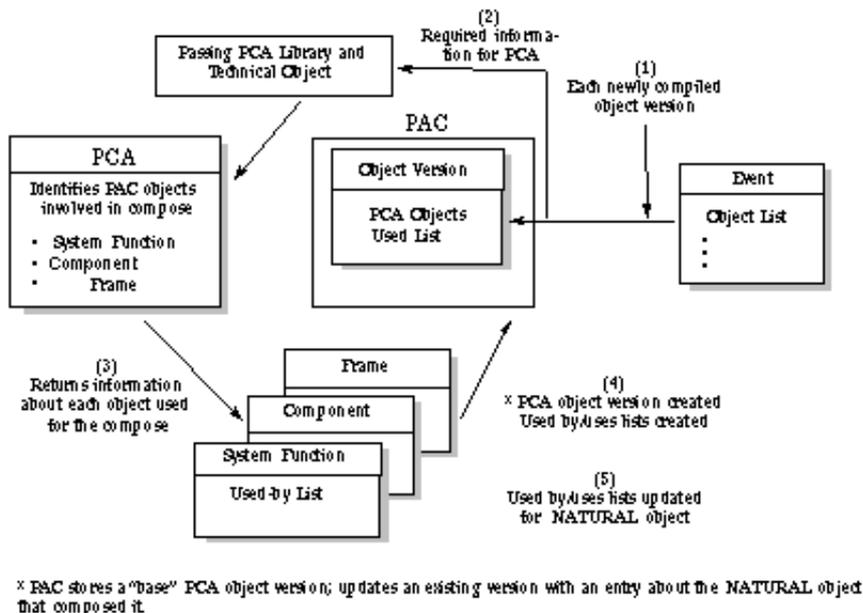
Reporting Facilities

PAC makes the following information available through the PAC reporting facilities in the PAC reporting subsystem:

- The Predict Case source objects used to compose each Natural object;
- The Natural objects that were composed using Predict Case source objects;

- The Natural/Predict Case object versions used, including the date migrated, the date effective in production, and the date superseded;
- Audit trail information in audit reports.
The relevant audit reporting screens display the following:
 - Objects that Predict Case uses
 - Information about specific Predict Case objects
 - The data structure of the Predict Case object
 - The objects that use the Predict Case object

The diagram below illustrates the life-cycle of a Natural object for Predict Case (PCA):



Explanation of the Steps in the Diagram

Step Action

- 1 Natural objects are composed from Predict Case objects specified for the migration of objects into the PAC controlled environment; each composed Natural object is verified before being newly compiled and versioned.
- 2 Predict Case library and technical object information about the Natural objects composed in PAC from Predict Case objects is communicated to Predict Case. Predict Case identifies the Predict Case objects involved in the composition of Natural objects in PAC.
- 3 Predict Case returns to PAC system function, component, and frame information about each Predict Case object used for the composition of Natural objects in PAC. The Predict Case objects are assigned a version number.
- 4 An object version is created for the Natural composed objects; used-by and uses lists are created. PAC updates the object version with an entry concerning the Predict Case objects used to compose it and including their version numbers.
- 5 Used-by and uses lists are updated for the Natural object versions.

Predict Case Object Classes

A Predict Case object is an entity used to compose Natural objects in a Predict Case environment. Examples of Predict Case object classes supported by PAC include

- System Function
- Frame Component

Natural for DB2

If you are running Natural for DB2 applications, PAC supports migrations for both qualified and non-qualified DB2 tables/views. The Truncate Creator option is supported for non-qualified tables/views.

Predict Event Prerequisite

Migrations for qualified and non-qualified DB2 tables/views must be performed separately. Ensure that the Predict generation defaults for views are set correctly before running the migration.

Migrating Qualified DB2 Tables/Views

If you are migrating qualified DB2 tables/views, then before running the Predict migration into PAC, check that the Truncate Creator option in the Predict generation defaults for views is set to N.

Migrating Non-Qualified DB2 Tables/Views

If you are migrating non-qualified DB2 tables/views, then before running the Predict migration into PAC, check that the Truncate Creator option in the Predict generation defaults for views is set to Y.

For information about copying Predict generation defaults, refer to the section General Defaults Maintenance in the PAC Administration Manual. For information about the Truncate Creator option in Predict generation defaults, refer to the Predict Administration Manual.

Truncate Creator Support for Non-Qualified DB2 Tables/Views

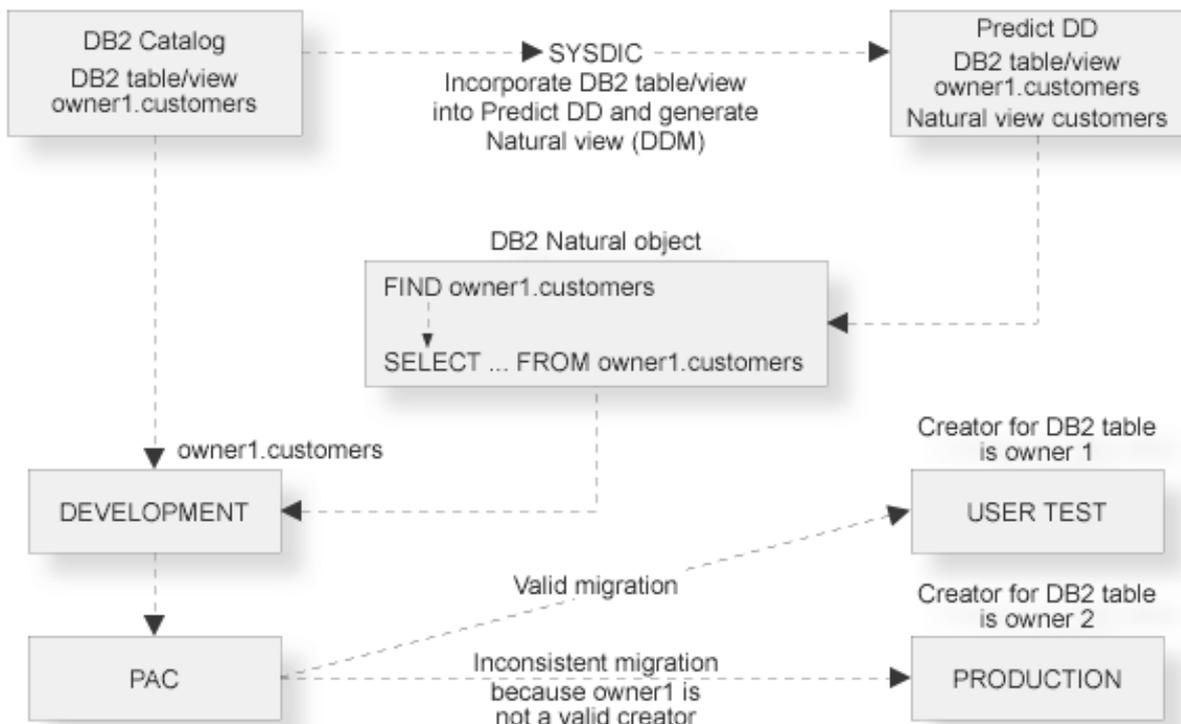
You may write Natural FOR DB2 applications with non-qualified tables/views by using the Truncator Creator option in the Generation Defaults for Natural views.

PAC supports the Truncate Creator option for DB2 tables/views with the following restrictions:

- The truncated view name of the DB2 table/view must not correspond to a view name which already exists for another
 - file; or
 - DB2 table/view.

Qualified and Non-Qualified Tables/Views

The diagrams in the following sections show examples of qualified and non-qualified DB2 tables/views:



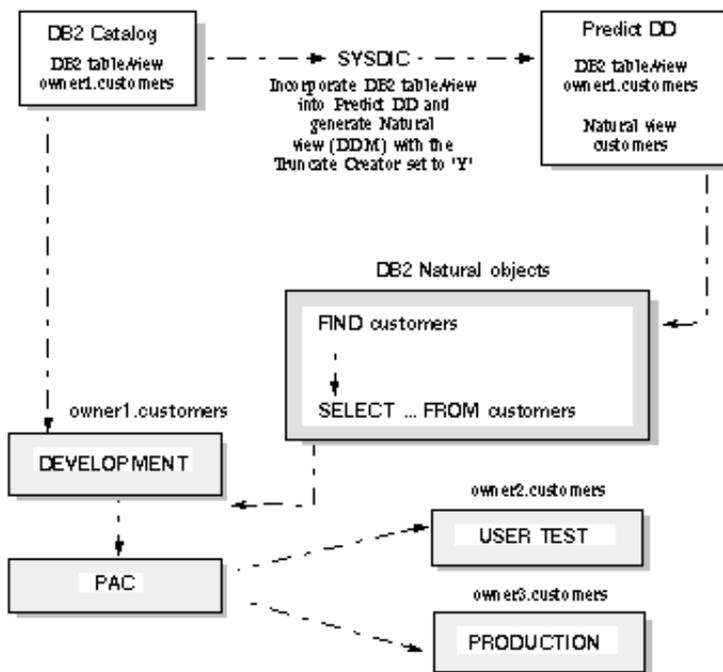
In the example above for qualified DB2 tables/views, objects that access owner1.customers should not be migrated to any other environment unless that environment supports the high level qualifier owner1.

If all environments support qualifier owner1, then the objects are valid for all environments.

If the objects are valid only for some environments, then the objects must be recompiled with different DB2 tables/views so that the objects may be executed against these DB2 tables.

Note:

Even if the SET CURRENT SQLID DB2 command is issued in dynamic mode, or the owner parameter is specified on the BIND for the application PLAN in static mode, the qualifier of customers will still be owner1.



In the example above for non-qualified DB2 tables/views in dynamic mode, the user refers to the truncated name of the DB2 view and will run the application in dynamic mode.

All objects may refer to customers and the qualifier will be determined during execution; therefore, the objects may be migrated to any environment where they can access the DB2 table based on the qualifier assigned by the SET CURRENT SQLID DB2 command.

For dynamic PLANs, the creator must be set up before executing the code that accesses the table in each environment. The setup may be performed in several ways; the following program is one example. This program may be found in library SYSDB2:

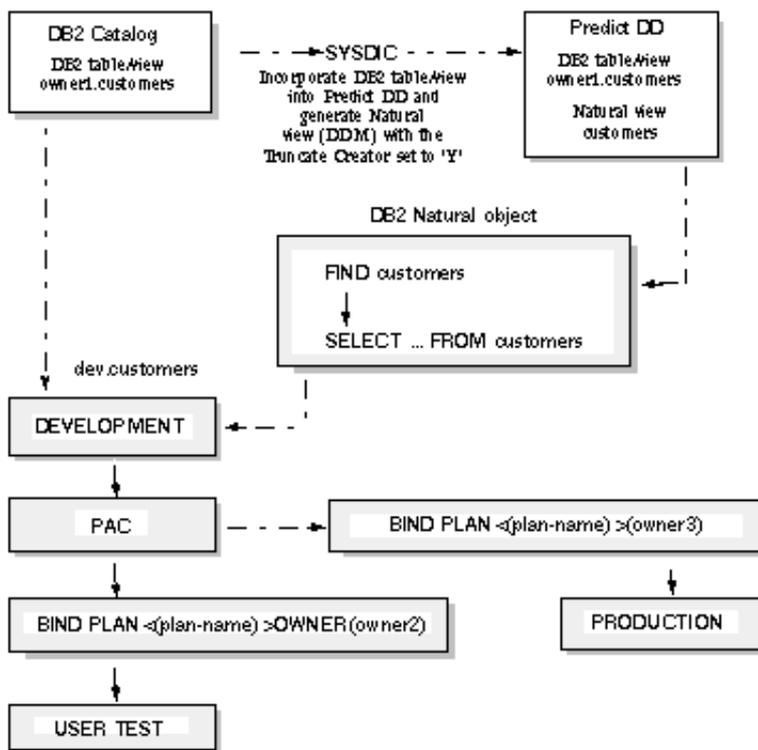
```

* SET CURRENT SQLID
*
DEFINE DATA
LOCAL USING DEMSQLCA
LOCAL
*
*                               Parameter for DB2SERV
1 STMT          (A250)
1 STMTL         (I2)          CONST <250>
1 RETCODE       (I2)
1 OLDSQLID     (A8)
1 NEWSQLID     (A8)
*
END-DEFINE
*
SELECT DISTINCT CURRENT SQLID
INTO OLDSQLID
FROM SYSIBM.SYSTABLES
ESCAPE BOTTOM
END-SELECT
*
MOVE 'SET CURRENT SQLID="PROD"'
TO STMT
CALL 'DB2SERV' 'D' STMT STMTL SQLCA RETCODE
*
IF RETCODE > 0
FETCH
    
```

```

'SQLERR' 0290 ELSE
  SELECT DISTINCT CURRENT SQLID
    INTO NEWSQLID
    FROM SYSIBM.SYSTABLES
  ESCAPE BOTTOM
  END-SELECT
*
  WRITE ' Old SQLID was :' OLDSQLID
  WRITE ' New SQLID is  :' NEWSQLID
END-IF
*
END

```



In the example above for non-qualified DB2 tables/views in static mode, the user's objects reference the truncated name of the DB2 table/view, and the qualifier will be determined by the BIND statement of the application PLAN.

Using this method, the user compiles the code only once and then can migrate the Natural for DB2 objects to any environment where the DB2 tables/views are accessible.