CA Gen

Host Encyclopedia Public Interface Reference Guide

Release 8.5

CA Technologies Product References

This document references the following CA Technologies products:

- CA Gen

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Appendix D: Possible Joins in the Public Interface 325

Chapter 1: Public Interface Export Function

The CA Gen Public Interface (PI) Export Function lets you tailor reports to your needs by selectively retrieving information from the Host Encyclopedia. You can use it to retrieve information created on the workstation and uploaded to the Host Encyclopedia. Export places the information in the Public Interface tables, as defined in the "Public Interface Table Definitions" appendix. From the tables you can retrieve the data by using SQL, a report writer, or some other program. Use the SQL examples to retrieve the information you need.

The objects in the PI tables are shown in the context of the workstation tools that create, in this sequence:

- Tasks related to all tables
- Planning objects
- Analysis objects
- Design objects
- Construction stage objects
- Model maintenance objects

This section contains the following topics:

Public Interface Tables

While we say you retrieve data from Public Interface tables, the data is actually retrieved from DB2 views on four DB2 tables. The term table is used for convenience.

Before you can access these extract tables, you must run a program that reads the data from the encyclopedia, formats the objects, and writes them to the extract tables.

To retrieve information after loading the tables, use SQL or the language of a report writer or other program to access the tables.

More information:

Export Model to PI Tables (see page 124)

Table Definitions

Each table represents an object, such as an entity type or process, and consists of rows and columns. Each row of the table represents an occurrence of an object, such as an entity named CUSTOMER. Each column of the row represents either a property of that object or a foreign key. Foreign keys are discussed under the next heading.

Each row of each table contains a column called MODEL_ID that uniquely identifies the model that contains the object. Each row also contains a column called ID that uniquely identifies the object within the encyclopedia. Each row contains a column called ORG_ID that gives the Original Object ID of the object within the encyclopedia.

For example, the table ATTRIBUTE contains one row for every attribute of every entity type in every model that has been exported to the Public Interface. Each attribute has a MODEL ID, a unique object ID in the third column, and a foreign key (PARENT_ENTITY_ID) that identifies the entity type to which the attribute belongs. Each row also contains columns for the attribute's properties, such as NAME, LENGTH, and DOMAIN.

Foreign Keys

Most tables contain a column that is a foreign key. This column enables users to join columns and combine tables. When combined, the tables can be referenced as if they were a single table.

For example, one of the columns of table ATTRIBUTE is a foreign key called PARENT_ENTITY_ID. This column can be joined, or matched, with the ID column of table ENTITY_TYPE to discover the parent entity type to which the attribute belongs.

Joining the foreign key column, PARENT_ENTITY_ID with the target column, ID, in the ENTITY_TYPE table shows that the attribute Governor belongs to the entity type State.

For more information, see the illustration in Identification.

The following join technique described and illustrated by the table is used extensively to select related objects from the Public Interface.

Identification

It is easy to identify foreign keys and their targets:

- Columns with names that end with suffix ID (_ID) are foreign keys to other tables
- Columns named ID are targets for foreign keys

All other columns represent properties of the objects in the tables. Details of when to join specific columns are given throughout this chapter.

A typical JOIN, in this case to retrieve all attributes of an entity type, looks like this:

```
ATTRIBUTE.PARENT_ENTITY_ID = ENTITY_TYPE.ID
```

The JOIN format, therefore, is:

```
FROM Table.Foreign Key = TO Table.Target
```

From TABLE ATTRIBUTE

ID	NAME	DSD NAME	SEQ	OPT	TYPE	DOMAIN	LENGTH	DECIMAL PLACES	PARENT ENTITY ID
8194	GOVERNOR	GOVERNOR	01	M	B	T	30	0	7953

Foreign Key

JOIN

Target

From TABLE ENTITY_TYPE

MODEL ID	TABLE	ID NAME	NAME	DSD_NAME	MIN OCC	MAX OCC	AVG OCC	GROW RATE	GROW RATE PER
4782	ENTITY TYPE	7953	STATE	GOVERNOR	50	50	50	0	Y

Frequently, the name of the foreign key without the suffix (_ID) is the same as the name of the TO table. In this example, the attribute table can be joined with the entity type table, entity subtype table, or system entity type table (work attribute set) because the PARENT_ENTITY_ID represents the foreign key of one of these three tables.

Task Index

Export tasks and the SQL required to retrieve information are discussed in the context of the tables. An index to all tasks follows.

Tables are arranged in groups that you may wish to use together. Following the tasks related to all tables, Planning tables are presented, followed by Analysis tables, External Design tables, Internal Design tables, Construction Management tables, and Model Management tables.

Export Function Task Index

The following index references discussions of tables for the various stages of development. These sections also contain SQL statements created for various SQL queries.

To Create Queries Related To This Stage	See This Section
All Stages	Tasks Related to All Tables
Planning	Planning Objects
Analysis	Data-Related Objects Activity-related Objects Analysis Tasks Related to Processes Intersection-Related Objects
External Design	Dialog Flow Diagram Objects Screen Design Tool Objects Business System Defaults
Internal Design	Internal Schema Definition Objects External Schema Definition Objects Design Tasks For Data Implementation Objects Access and Connection Objects
Construction	Construction DB2 Objects Construction Objects Construction Objects for Cascade Delete
Model Maintenance	Model Management Objects

Tasks Related to All Tables

This section discusses the tasks related to all the tables.

Model Table

The Model Table (MODEL) can be referenced from all other tables and is used for model selection. Two of its columns are described in the following:

- **NAME**—Host Encyclopedia name of the model
- **ID**—A unique identifier of the model across the encyclopedia

All other tables contain at least two columns, described in the following:

- **ID**—A unique identifier of this object across the encyclopedia
- MODEL_ID—Identifies the model that contains this object.

  This column is joined with the ID column in the MODEL table to restrict selection to objects in the selected model.

The following syntax:

```
SELECT
  FROM MODEL, table
  WHERE MODEL.NAME = 'my model name'
  AND table.MODEL_ID = MODEL.ID;
```

Selects all rows from table that are in model *my model name*.

Qualifying Objects

When you reference the DB2 tables, DB2 rules relating to table names apply. That is, if your DB2 authorization ID is different from the ID under which the DB2 tables were created, you must use the qualified name. The qualified name consists of three parts: the authorization ID under which the DB2 table was created, followed by a period, then the table name. For example, use A12345.ENTITY_TYPE where A12345 is the authorization ID under which the table ENTITY_TYPE was created. Synonyms can be created and used. This guide uses the unqualified name.

Typically, you should qualify only the highest object in the selection with the model ID. For example, if you want to select entity types and their attributes, qualify entity type within model and attribute within entity type. For performance reasons, it can be advantageous to qualify all tables with model ID, if the join limits permit.

Important! If both entity type and attribute are qualified by model ID, but attribute is not qualified by entity type, a Cartesian product of the two tables results: The number of rows returned is the product of the number of rows selected from each table.

Description Tables

Descriptions (DESC and TEXT) are created throughout the toolsets.

Desc Descriptions

The description table (DESC) contains the textual description for all objects. When you select Detail and Description in a toolset, you create a row in this table. The text of the description of each object that has a description is maintained in this table.

This table has two columns. They are:

- **ID**—Identifier of the object for which this is a description.
- **TEXT**—Text of the description

Join ID of DESC with ID of the table for which the description is sought. Remember, if no description exists, nothing is returned. TEXT is a variable length text field.

To select the name and description of objects in table, enter the following syntax:

```
SELECT table.NAME, DESC.TEXT
  FROM MODEL, table, DESC
  WHERE MODEL.NAME= 'my model name'
  AND table.MODEL_ID = MODEL.ID
  AND DESC.ID = table.ID;
```

The following tables have descriptions:

Table Name	Table Contents
ACTION_BLOCK	Action Block Definition (Analysis and Design)
ACTIVITY_CLUSTER	Activity Cluster
ADD_DEPENDNCY	ADD Dependency
ALIAS	Alias name for an entity type, subtype, or attribute
ATTRIBUTE	Attribute of an entity type, subtype, or system entity type (work attribute set)
BAA_ACTN_BLK	Analysis Action Block Definition
BSD_ACTN_BLK	Design Action Block
BUS_GOAL	Business Goal
BUS_LOCATION	Location of Business Assets
BUS_OBJECTIVE	Business Objective
BUS_PROC_STEP	Business Procedure Step
BUSINESS_SYS	Business System
BUSINESS_PROC	Business Procedure
COMPONENT_IMPLEM	Component Implementation
COMPONENT_SPEC	Component Specification
CRITICAL_SUCCESS	Critical Success Factor
CURRENT_INFO_SYS	Current Information System
CURRENT_DATA	Current Database or Store

Table Name	Table Contents
DATA_BASE	Database
DATA_CLUSTER	Data Cluster
DATA_STORE_INDEX	Indexspace Data Store
DATA_STORE_TBLSP	Tablespace Data Store
DATASET_INDEX	Indexspace Data Set
DATASET_TBLSP	Tablespace Data Set
DERIVATION_ALGOR	Derivation Algorithm
DIALECT	Dialect
DIALOG_FLOW	Dialog Flow
ENTITY_SUBTYP	Entity subtype
ENTITY_TYPE	Entity type
ENTITY_VIEW	Entity View
ENTRY_POINT	Entry Point Definition
ENVIRONMENT	Hardware/Software Environment
EXPECT_EFFECT	Expected Effect
FACILITY	Computing/Communication Facility
FIELD	Field Definition
FUNCTION_DEF	Function Definition
GROUP_VIEW	Group View
IDENTIFIER	Identifier
INFORMATION_NEED	Information Need
INTERFACE_TYPE	Interface Type
INTRFCE_TYPE_MDL	Interface Type Model
LINKAGE	Linkage
MATRIX	Matrices
OBJECT_CLASS	Object Class (System- and User-defined)
ORGANIZAT_UNIT	Organizational Unit
PAD_FUNCTION	System-supplied Functions
PARTITIONING	Partitioning of an entity type or subtype
PERMIT_VALUE	Permitted Value of an attribute

Table Name	Table Contents
PROCESS_DEF	Process Definition
RECORD	Record Definition
REL_MUTL_EXCL	Mutually exclusive relationship
RELATIONSHIP	Relationship between two entity types/subtypes
SCREEN_DEF	Screen Definition
SCREEN_TMPLT	Screen Template
SPEC_TYPE	Specification Type
STRATEGY	Strategy
SUBJECT AREA	Subject Area
SYS_ATTRIBUTE	System Attribute
SYS_ENT_TYPE	System Entity Type (Work Attribute Set)
TACTIC	Tactic
USER_DEF_OBJECT	User-defined Object
XTERNL_EVENT	External Event
XTERNL_OBJECT	External Object

TEXT Descriptions

The text table (TEXT) is created when a text string is to be saved. In most instances, such as PROMPT, SCRN_FLD_LIT, CSTM_EDT_PTRN, DFLT_EDT_PTRN, SYS_ATTRIBUTE LIBRARY, DIALECT_TEXT, and FLD_ENTPT_VALUE, the text is part of the object table.

Exceptions:

- To retrieve the exit state message, join the EXIT_STATE table with the TEXT table
- To retrieve the organization mission for the organizational unit, join the ORGANIZAT_UNIT with the TEXT table
- To retrieve the long name of an ALIAS, join the ALIAS table with the TEXT table

TEXT usage is identical to that of DESC.

The text table SCRN_HELP is created when a screen help identifier string is to be saved. To retrieve the screen help identifier, join the tables SCREEN_DEF, SCRN_SYS_DEF, and SCRN_VAR_DEF with the SCRN_HELP table. SCRN_HELP usage is identical to that of DESC.

The text table PARM is created when the parameters for a fieldproc routine is to be saved. To retrieve the parameters to the Fieldproc Routine, join FIELD with the PARM table. PARM usage is also identical to DESC usage.

The text table PERMIT_VALUE_LOW is created when a permitted value (or the lower limit of a range) is to be saved. The text table PERMIT_VALUE_HI is created when a permitted value of the upper limit of a range is to be saved. If the permitted value is a range, join the PERMIT_VALUE with PERMIT_VALUE_ LOW and PERMIT_VALUE_HI. Otherwise, join the table PERMIT_VALUE with only PERMIT_VALUE_LOW to retrieve the permitted value. PERMIT_ VALUE_LOW and PERMIT_VALUE_HI usage is also identical to DESC usage.

Planning Objects

Information strategy planning objects are entered through the Planning function.

Planning Object	Description
MATRIX	Planning Matrices
OBJECT_CLASS	Object class (system- and user-defined)
MATRIX_USAGE_X	Matrix usage for X axis
MATRIX_USAGE_Y	Matrix usage for Y axis
CELL_VALUE	Matrix cell value
USER_DEF_OBJECT	User-defined object
ACTIVITY_CLUSTER	Activity cluster (nat bus sys)
BUS_GOAL	Business goal
BUS_LOCATION	Location of business assets
BUS_OBJECTIVE	Business objective
CRITICAL_SUCCESS	Critical success factor
CURRENT_DATA	Current database or data store
CURRENT_INFO_SYS	Current information system
DATA_CLUSTER	Data cluster
ENVIRONMENT	Hardware/Software environment
FACILITY	Computing/Communication Facility
INFORMATION_NEED	Information need
ORGANIZAT_UNIT	Organizational unit

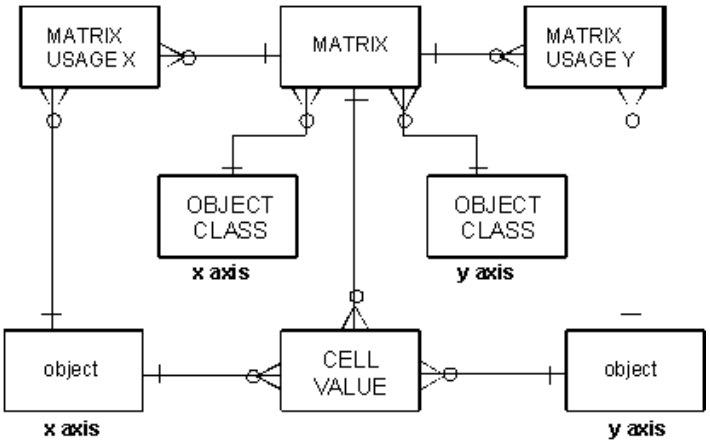
Planning Object	Description
PERFORM_MEASURE	Performance measure
STRATEGY	Strategy
TACTIC	Tactic
CURRENT_EFFECT	Current system/data effect

System-defined Matrix

The system-defined matrices (MATRIX) are created automatically when a model is created. Each matrix has a class (OBJECT_CLASS) on the X axis and a class (OBJECT_CLASS) on the Y axis. These classes are system-defined classes. There is an association (MATRIX_USAGE_X) between each object (X) on the X axis and the matrix. There is also an association between each object (Y) on the Y axis and the matrix. A cell value (CELL_VALUE) can exist between each object on the X axis and each object on the Y axis.

Replace X and Y with the name of the Public Interface table that contains the objects on the X and Y axes.

For relationships between Planning objects within system-defined matrices, see the following illustration:



User-defined Matrix

The user-defined matrices (MATRIX) are created through Matrices in the Planning toolset. Each matrix has a class (OBJECT_CLASS) on the X axis and a class (OBJECT_CLASS) on the Y axis. These classes can be system-defined classes or user-defined classes. A user-defined class contains many user-defined objects (USER_DEF_OBJECT).

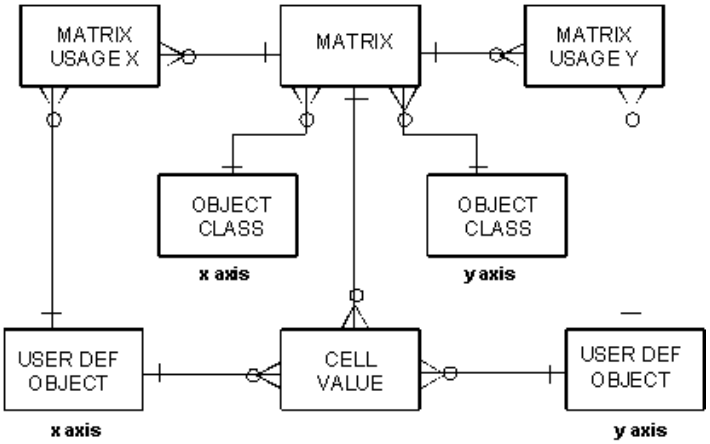
Matrix Usage

An association (MATRIX_USAGE_X) exists between each object(X) on the X axis and the matrix. An association (MATRIX_USAGE_Y) also exists between each object (Y) on the Y axis and the matrix.

Cell Values

A cell value (CELL_VALUE) can exist between each object on the X axis and each object on the Y axis.

For an example of a user-defined matrix with user-defined objects on both axes, see the following illustration:



For a user-defined matrix with system-defined classes, replace X and Y with the name of the Public Interface table that contains the objects on the X and Y axes.

Planning Tasks

The following table directs you to the PI table explanation that contains SQL for each Planning task listed. In general, the tasks select all occurrences of a Planning object within a model or all occurrences on the X or Y axis of a matrix.

If You Want to Select	See This PI Table
The activity clusters of a model	ACTIVITY_CLUSTER
The activity clusters on the Y axis	ACTIVITY_CLUSTER
The business goals for a model	BUS_GOAL
The business goals on the X axis	BUS_GOAL
The business objectives of a model	BUS_OBJECTIVE

If You Want to Select	See This PI Table
The business objectives on the Y axis	BUS_OBJECTIVE
The computing/communication facilities of a model	FACILITY
The computing/ communication facilities on the X axis	FACILITY
The critical success factors of a model	CRITICAL_SUCCESS
The critical success factors on the X axis	CRITICAL_SUCCESS
The current databases or data stores of a model	CURRENT_DATA
The current databases or the data stores on the X axis	CURRENT_DATA
The current effects of each current information system	CURRENT_EFFECT
The current information systems of a model	CURRENT_INFO_SYS
The current information systems on the X axis	CURRENT_INFO_SYS
The data clusters of a model	DATA_CLUSTER
The data clusters on the X axis	DATA_CLUSTER
The entity types along each axis and the corresponding cell value	CELL_VALUE
The entity types along the X axis	MATRIX_USAGE_X
The entity types along the Y axis	MATRIX_USAGE_Y
The hardware/software environments of a model	ENVIRONMENT
The hardware/software environments on the X axis	ENVIRONMENT
The information needs of a model	INFORMATION_NEED
The information needs on the Y axis	INFORMATION_NEED
The locations of business assets of a model	BUS_LOCATION
The locations of business assets on the X axis	BUS_LOCATION
All the system-defined matrices and the classes on the X and Y axes	MATRIX
All the user-defined matrices and the classes on the X and Y axes	MATRIX
The organizational units of a model	ORGANIZAT_UNIT

If You Want to Select	See This PI Table
The organizational units on the X axis	ORGANIZAT_UNIT
Each system-defined matrix and system-defined classes on each axis and the corresponding PI table name for each class	OBJECT_CLASS
All the user-defined classes	OBJECT_CLASS
All the object classes (system- and user-defined)	OBJECT_CLASS
All the matrices that have the object class ENTITY_TYPE on the X axis	OBJECT_CLASS
All the matrices that have the object class ENTITY_TYPE on the X axis	OBJECT_CLASS
The user-defined classes on the X and Y axes, and the user-defined objects on both axes of a user-defined matrix	USER_DEF_OBJECT
The performance measures of a model	PERFORM_MEASURE
The performance measures on the Y axis	PERFORM_MEASURE
The strategies on the X axis	STRATEGY
The strategies of a model	STRATEGY
The tactics for a model	TACTIC
The tactics on the X axis	TACTIC

MATRIX (Planning Matrices)

The system-defined matrices are created automatically when a model is created. The user-defined matrix is created by using Matrices in the Planning toolset. Each matrix has a class on the X axis and a class on the Y axis. The name of the system-defined matrix is name of the class on Y axis/name of the class on X axis.

For example, the name of the matrix with business functions on the Y axis and entity types on the X axis is BUSINESS_FUNCTION/ENTITY_TYPE.

Note: The ROTATE command on the toolset does not change the axis where the classes are stored. It is only for display purposes on the toolset.

Task

Join a matrix to its class on the X axis and the Y axis.

```
MATRIX.OBJ_CLASS_X_ID = CX.ID
AND MATRIX.OBJ_CLASS_Y_ID = CY.ID
```

where CX and CY refer to the OBJECT_CLASS table.

Task

Select all the system-defined matrices and the classes on the X and Y axes. Order by name of matrix.

```
SELECT MT.NAME, CX.NAME, CY.NAME
  FROM
  MODEL M,
  MATRIX MT,
  OBJECT_CLASS CX,
  OBJECT_CLASS CY
  WHERE
  M.NAME = 'my model name'
  AND MT.MODEL_ID = M.ID
  AND MT.IEF_OR_USER_SUP = 'I'
  AND MT.OBJ_CLASS_X_ID = CX.ID
  AND MT.OBJ_CLASS_Y_ID& = CY.ID
ORDER BY MT.NAME;
```

Task

Select all the user-defined matrices and the classes on the X and Y axis. Order by name of matrix.

```
SELECT MT.NAME, CX.NAME, CY.NAME
  FROM
  MODEL M,
  MATRIX MT,
  OBJECT_CLASS CX,
  OBJECT_CLASS CY
  WHERE
  M.NAME = 'my model name'
  AND MT.MODEL_ID = M.ID
  AND MT.IEF_OR_USER_SUP = 'U'
  AND MT.OBJ_CLASS_X_ID = CX.ID
  AND MT.OBJ_CLASS_Y_ID = CY.ID
ORDER BY MT.NAME;
```

OBJECT_CLASS (System- and User-defined)

The system-defined classes are created automatically when a model is created. The user-defined classes are created by using Matrices in the Planning toolset. Each system-defined class has a Public Interface table that contains the objects of that class. The column PI_Name stores the name of the Public Interface table. For user-defined classes PI_name is blank. For user-defined classes, the Public Interface table that contains user-defined objects is USER_DEF_OBJECT. For more information, see USER_DEF_OBJECT.

Note: The ROTATE command on the toolset does **not** change the axis on which the classes are stored. The command is only for display purposes on the toolset.

Task

Select each system-defined matrix, class name of X axis, PI table name for class on X axis, class name of Y axis, and PI table name of Y axis. Order by name of matrix.

```
SELECT MT.NAME, CX.NAME, CX.PI_NAME, CY.NAME, CY.PI_NAME
  FROM
  MODEL M,
  MATRIX MT,
  OBJECT_CLASS CX,
  OBJECT_CLASS CY
  WHERE
  M.NAME = 'my model name'
  AND MT.MODEL_ID = M.ID
  AND MT.IEF_OR_USER_SUP = 'I'
  AND MT.OBJ_CLASS_X_ID = CX.ID
  AND MT.OBJ_CLASS_Y_ID = CY.ID
ORDER BY MT.NAME;
```

Task

Select all the user-defined classes.

```
SELECT C.NAME
  FROM
  MODEL M,
  OBJECT_CLASS C
  WHERE
  M.NAME = 'my model name'
  AND C.MODEL_ID = M.ID
  AND C.IEF_OR_USER_SUP = 'U'
ORDER BY C.NAME;
```

Task

Select all the object classes (both system-defined and user-defined).

```
SELECT C.NAME
  FROM
  MODEL M,
  OBJECT_CLASS C
  WHERE
  M.NAME = 'my model name'
  AND C.MODEL_ID = M.ID
ORDER BY C.NAME;
```

Task

Select all the matrices that have the object class FUNCTION_ DEFINITION on the X axis.

```
SELECT MT.NAME
FROM
MODEL M,
MATRIX MT,
OBJECT_CLASS C
WHERE
M.NAME = 'my model name'
AND MT.MODEL_ID = M.ID
AND MT.OBJ_CLASS_X_ID = C.ID
AND C.NAME = 'FUNCTION_DEFINITION'
ORDER BY MT.NAME;
```

MATRIX_USAGE_X (Usage for X Axis)

The MATRIX_USAGE_X table represents the association between a matrix and each object on the X axis of the matrix. There is a row in the MATRIX_USAGE_X table for each object along the X axis of a matrix.

Task

Join a matrix usage to its matrix and its object on the X axis.

```
MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = X.ID
```

Replace X with the table that contains the objects on X axis. (For information on determining PI table name for objects on X axis, see OBJECT_CLASS).

Task

Select all the matrices that have the object class FUNCTION_ DEFINITION on the X axis.

```
SELECT MT.NAME
FROM
MODEL M,
MATRIX MT,
OBJECT_CLASS C
WHERE
M.NAME = 'my model name'
AND MT.MODEL_ID = M.ID
AND MT.OBJ_CLASS_X_ID = C.ID
AND C.NAME = 'FUNCTION_DEFINITION'
ORDER BY MT.NAME; purposes on the toolset.
```

MATRIX_USAGE_Y (Usage for Y Axis)

The MATRIX_USAGE_Y table represents the association between a matrix and each object on the Y axis of the matrix. Each row in the MATRIX_USAGE_Y table represents an object along the Y axis of a matrix.

Task

Join a matrix usage to its matrix and its object on the Y axis.

```
MATRIX_USAGE_Y.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_Y.OBJECT_ID = Y.ID
```

Replace Y with the table that contains the objects on Y axis. (For information on determining PI table name for objects on X axis, see OBJECT_CLASS).

Task

Select the entity types along the Y axis of the matrix ENTITY_TYPE/ENTITY_TYPE. Order by entity type as specified in toolset.

```
SELECT MT.NAME, Y.NAME. UY.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_Y UY,
  ENTITY_TYPE Y
  WHERE
  M.NAME = 'my model name'
  AND UY.MODEL_ID = M.ID
  AND UY.MATRIX_ID = MT.ID
  AND MT.NAME = 'ENTITY_TYPE/ENTITY_TYPE'
  AND UY.OBJECT_ID = Y.ID
ORDER BY MT.NAME, UY.SEQ;
```

CELL_VALUE (Cell Value for a Matrix)

The CELL_VALUE table represents the cell value between an object on the X axis and an object on the Y axis of a matrix. For each non-blank cell value, there is a row in the CELL_VALUE table.

Task

Join the cell value with its matrix, object on X axis, and object on Y axis.

```
CELL_VALUE.MATRIX_ID = MATRIX.ID
AND CELL_VALUE.X_OBJECT_ID = X.ID
AND CELL_VALUE.Y_OBJECT_ID = Y.ID
```

Replace X with the PI table for objects on X axis. Replace Y with the PI table for objects on Y axis. For more information, see OBJECT_CLASS.

Task

Select the name of class on X axis, name of class on Y axis, name of object on X axis and its sequence, name of object on Y axis and its sequence, and the cell value for the matrix ENTITY_TYPE/ENTITY_TYPE.

```
SELECT MT.NAME, CX.NAME, CY.NAME, X.NAME, UX.SEQ, Y.NAME,
UY.SEQ, V.CELL_VALUE
FROM
MODEL M,
MATRIX MT,
OBJECT_CLASS CX,
OBJECT_CLASS CY,
MATRIX_USAGE_X UX,
ENTITY_TYPE X,
MATRIX_USAGE_Y, UY,
ENTITY_TYPE Y,
CELL_VALUE V
WHERE
M.NAME = 'my model name'
AND MT.MODEL_ID = M.ID
AND MT.NAME = 'ENTITY_TYPE/ENTITY_TYPE'
AND MT.OBJ_CLASS_X_ID = CX.ID
AND MT.OBJ_CLASS_Y_ID = CY.ID
AND UX.MATRIX_ID = MT.ID
AND UX.OBJECT_ID = X.ID
AND UY.MATRIX_ID = MT.ID
AND UY.OBJECT_ID = Y.ID
AND V.MATRIX_ID = MT.ID
AND V.X_OBJECT_ID = X.ID
AND V.Y_OBJECT_ID = Y.ID
ORDER BY MT.NAME, UX.SEQ;
```

Note: Objects can appear on an axis without a cell value. Such objects are not found by the SELECT.

Some matrices use other views to retrieve cell values. See the following table:

To Select Cell Values for this Matrix	Use this View
CURRENT_INFO_SYS/CURRENT_DATA_STORE	CURRENT_EFFECT
BUSINESS_FUNCTION/ENTITY_TYPE	EXPECT_EFFECT
ELEMENTARY_PROCESS/ENTITY_TYPE	EXPECT_EFFECT

USER_DEF_OBJECT (User-defined Object)

The user-defined objects are created by using the Object Definition function in the Planning toolset. The name of the user-defined class is stored in the Class_Type column.

Task

Select all the user-defined objects for a user-defined class.

```
SELECT O.NAME
  FROM
  MODEL M,
  USER_DEF_OBJECT O
  WHERE
  M.NAME = 'my model name'
  AND O.MODEL_ID = M.ID
  AND O.CLASS_TYPE = 'name of user-defined class'
ORDER BY O.NAME;
```

Task

Select user-defined classes on the X and Y axis, and the user-defined objects on both axis of a user-defined matrix.

```
SELECT MT.NAME, CX.NAME, CY.NAME, X.NAME, UX.SEQ, Y.NAME, UY.SEQ,
.CELL_VALUE
  FROM
  MODEL M,
  MATRIX MT,
  OBJECT_CLASS CX,
  OBJECT_CLASS CY,
  MATRIX_USAGE_ UX,
  USER_DEF_OBJECT X,
  MATRIX_USAGE_Y UY,
  USER_DEF_OBJECT Y,
  CELL_VALUE V
  WHERE
  M.NAME = 'my model name'
  AND MT.MODEL_ID = M.ID
  AND MT.NAME = 'name of user-defined matrix'
  AND MT.OBJ_CLASS_X_ID = CX.ID
  AND MT.OBJ_CLASS_Y_ID = CY.ID
  AND UX.MATRIX_ID = MT.ID
  AND UX.OBJECT_ID = X.ID
  AND UY.MATRIX_ID = MT.ID
  AND UY.OBJECT_ID = Y.ID
  AND V.MATRIX_ID = MT.ID
  AND V.X_OBJECT_ID = X.ID
  AND V.Y_OBJECT_ID = Y.ID
ORDER BY MT.NAME, UX.SEQ;
```

Note: A user-defined matrix can contain system-defined object classes. The user-defined matrix is not required to have user-defined objects. In that case, the X and Y table would be replaced with the name of the table that contains the objects on the axis. For more information, see OBJECT_CLASS.

ACTIVITY_CLUSTER

The activity clusters are created by using Matrices in the Planning toolset. The system-defined class for the activity cluster ACTIVITY_CLUSTER. The system-defined matrices that contain activity clusters are:

- BUSINESS_AREA/ACTIVITY_CLUSTER
- ACTIVITY_CLUSTER/BUS_FUNCTION
- ACTIVITY_CLUSTER/LOCATION

Task

Select the activity clusters for a model.

```
ACTIVITY_CLUSTER.MODEL_ID = MODEL.ID
```

Task

Select the activity clusters on the Y axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_Y.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_Y.OBJECT_ID = ACTIVITY_CLUSTER.ID
```

Task

Select the activity clusters along the Y axis of the matrix ACTIVITY_CLUSTER/LOCATION. Order by activity cluster as specified in toolset.

```
SELECT MT.NAME, Y.NAME. UY.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_Y UY,
  ACTIVITY_CLUSTER Y
  WHERE
  M.NAME = 'my model name'
  AND UY.MODEL_ID = M.ID
  AND UY.MATRIX_ID = MT.ID
  AND MT.NAME = 'ACTIVITY_CLUSTER/LOCATION'
  AND UY.OBJECT_ID = Y.ID
ORDER BY MT.NAME, UY.SEQ;
```

BUS_GOAL (Business Goal)

The business goals are created by using Matrices in the Planning toolset. The system-defined class for the business goal is BUSINESS_GOAL. The system-defined matrix that contains business goals is INFORMATION_NEED/GOAL.

Task

Select the business goals of a model.

```
BUS_GOAL.MODEL_ID = MODEL.ID
```

Task

Select the business goals on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = BUS_GOAL.ID
```

An SQL example appears following:

Task

Select the business goals along the X axis of the matrix INFORMATION_NEED/GOAL. Order by business goal as specified in the toolset.

```
SELECT MT.NAME, X.NAME. UX.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_X UX,
  BUS_GOAL X
  WHERE
  M.NAME = 'my model name'
  AND UX.MODEL_ID = M.ID
  AND UX.MATRIX_ID = MT.ID
  AND MT.NAME = 'INFORMATION_NEED/GOAL'
  AND UX.OBJECT_ID = X.ID
ORDER BY MT.NAME, UX.SEQ;
```

BUS_LOCATION (Location of Assets)

The locations of business assets are created by using Matrices in the Planning toolset. The system-defined class for the location of business assets is LOCATION_OF_BUSINESS_ASSETS. The system-defined matrices that contain locations of business assets are:

- ORGANIZATIONAL_UNIT/LOCATION
- BUSINESS_FUNCTION/LOCATION
- ENTITY_TYPE/LOCATION
- CURRENT_INFO_SYSTEM/LOCATION
- CURRENT_DATA_STORE/LOCATION
- ACTIVITY_CLUSTER/LOCATION
- DATA_CLUSTER/LOCATION
- LOCATION/LOCATION

Task

Select the locations of business assets for a model.

```
BUS_LOCATION.MODEL_ID = MODEL.ID
```

Task

Select the locations of business assets on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = BUS_LOCATION.ID
```

Task

Select the locations of business assets along the X axis of the matrix BUSINESS_FUNCTION/LOCATION. Order by location of business assets as specified in the toolset.

```
SELECT MT.NAME, X.NAME. UX.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_X UX,
  BUS_LOCATION X
  WHERE
  M.NAME = 'my model name'
  AND UX.MODEL_ID = M.ID
  AND UX.MATRIX_ID = MT.ID
  AND MT.NAME = 'BUSINESS_FUNCTION/LOCATION'
  AND UX.OBJECT_ID = X.ID
ORDER BY MT.NAME, UX.SEQ;
```

BUS_OBJECTIVE (Business Objective)

The business objectives are created by using Matrices in the Planning toolset. The system-defined class for the business objective is BUS_OBJECTIVE. The system-defined matrices that contain business objectives are:

- INFORMATION_NEED/OBJECTIVE
- OBJECTIVE/STRATEGY
- OBJECTIVE/CRIT._SUCCESS_FACTOR

Task

Select the business objectives for a model.

```
BUS_OBJECTIVE.MODEL_ID = MODEL.ID
```

Task

Select the business objectives on the Y axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_Y.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_Y.OBJECT_ID = BUS_OBJECTIVE.ID
```

Task

Select the business objectives along the Y axis of the matrix OBJECTIVE/STRATEGY. Order by business objective as specified in the toolset.

```
SELECT MT.NAME, Y.NAME. UY.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_Y UY,
  BUS_OBJECTIVE Y
  WHERE
  M.NAME = 'my model name'
  AND UY.MODEL_ID = M.ID
  AND UY.MATRIX_ID = MT.ID
  AND MT.NAME = 'OBJECTIVE/STRATEGY'
  AND UY.OBJECT_ID = Y.ID
ORDER BY MT.NAME, UY.SEQ;
```

CRITICAL_SUCCESS (Critical Success Factor)

The critical success factors are created by using Matrices in the Planning toolset. The system-defined class for the critical success factor is CRITICAL_SUCCESS_FACTOR. The system-defined matrices that contain critical success factors are:

- CRIT_SUCCESS_FACTOR/ORG_UNIT
- OBJECTIVE/CRIT_SUCCESS_FACTOR
- INFO_NEED/CRIT_SUCCESS_FACTOR

Task

Select the critical success factors for a model.

```
CRITICAL_SUCCESS.MODEL_ID = MODEL.ID
```

Task

Select the critical success factors on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = CRITICAL_SUCCESS.ID
```

Task

Select the critical success factors along the X axis of the matrix OBJECTIVE/CRIT._SUCCESS_FACTOR. Order by critical success factor as specified in the toolset.

```
SELECT MT.NAME, X.NAME. UX.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_X UX,
  CRITICAL_SUCCESS X
  WHERE
  M.NAME = 'my model name'
  AND UX.MODEL_ID = M.ID
  AND UX.MATRIX_ID = MT.ID
  AND MT.NAME = 'OBJECTIVE/CRIT._SUCCESS_FACTOR'
  AND UX.OBJECT_ID = X.ID
ORDER BY MT.NAME, UX.SEQ;
```

CURRENT_DATA (Database or Data Store)

The current databases or data stores are created by using Matrices in the Planning toolset. The system-defined class for the current database or data store is CURRENT_DATA. The system-defined matrices that contain current databases or data stores are:

- ENTITY_TYPE/CUR._DATA_STORE
- ORG_UNIT/CURRENT_DATA_STORE
- SUBJECT_AREA/CURRENT_DATA_STORE
- CURRENT_DATA_STORE/LOCATION

Task

Select the current databases or data stores for a model.

```
CURRENT_DATA.MODEL_ID = MODEL.ID
```

Task

Select the current databases or data stores on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = CURRENT_DATA.ID
```

Task

Select the current databases or data stores along the X axis of the matrix ENTITY_TYPE/CUR_DATA_STORE. Order by current database or data store as specified in the toolset.

```
SELECT MT.NAME, X.NAME. UX.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_X UX,
  CURRENT_DATA X
  WHERE
  M.NAME = 'my model name'
  AND UX.MODEL_ID = M.ID
  AND UX.MATRIX_ID = MT.ID
  AND MT.NAME = 'ENTITY_TYPE/CUR._DATA_STORE'
  AND UX.OBJECT_ID = X.ID
ORDER BY MT.NAME, UX.SEQ;
```

CURRENT_INFO_SYS (Current Information System)

The current information systems are created by using Matrices in the Planning toolset. The system-defined class for the current information system is CURRENT_BUSINESS_SYSTEM. The system-defined matrices that contain current information systems are:

- CUR._INFO_SYS/CUR_DATA_STORE
- BUS_FUNCTION/CUR_INFO_SYSTEM
- ORG_UNIT/CUR_INFO_SYSTEM
- CURRENT_INFO_SYSTEM/LOCATION
- BUSINESS_AREA/CURRENT_INFO_SYS
- CURRENT_INFOSYS/INFO_NEED

Task

Select the current information systems for a model.

```
CURRENT_INFO_SYS.MODEL_ID = MODEL.ID
```

Task

Select the current information systems on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = CURRENT_INFO_SYS.ID
```

Task

Select the current information systems along the X axis of the matrix BUS_FUNCTION/CUR_INFO_SYSTEM. Order by current information system as specified in the toolset.

```
SELECT MT.NAME, X.NAME. UX.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_X UX,
  CURRENT_INFO_SYS X
  WHERE
  M.NAME = 'my model name'
  AND UX.MODEL_ID = M.ID
  AND UX.MATRIX_ID = MT.ID
  AND MT.NAME = 'BUS_FUNCTION/CUR._INFO_SYSTEM'
  AND UX.OBJECT_ID = X.ID
ORDER BY MT.NAME, UX.SEQ;
```

DATA_CLUSTER

The data clusters are created by using Matrices in the Planning toolset. The system-defined class for the data cluster is DATA_CLUSTER_(NAT_ DATA_STORE). The system-defined matrices that contain data clusters are:

- BUSINESS_AREA/DATA_CLUSTER
- DATA_CLUSTER/ENTITY_TYPE
- DATA_CLUSTER/LOCATION

Task

Select the data clusters of a model.

```
DATA_CLUSTER.MODEL_ID = MODEL.ID
```

Task

Select the data clusters on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = DATA_CLUSTER.ID
```

Task

Select the data clusters along the X axis of the matrix BUSINESS_AREA/ DATA_CLUSTER. Order by data cluster as specified in the toolset.

```
SELECT MT.NAME, X.NAME. UX.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_X UX,
  DATA_CLUSTER X
  WHERE
  M.NAME = 'my model name'
  AND UX.MODEL_ID = M.ID
  AND UX.MATRIX_ID = MT.ID
  AND MT.NAME = 'BUSINESS_AREA/DATA_CLUSTER'
  AND UX.OBJECT_ID = X.ID
ORDER BY MT.NAME, UX.SEQ;
```

ENVIRONMENT (Hardware/Software)

You create hardware/software environments by using Matrices in the Planning toolset. The system-defined class for the hardware/software environment is HW/SW_ENVIRONMENT. There are not any system-defined matrices that contain hardware/software environments. To create hardware/software environments, first create a user-defined matrix with one of its classes as hardware/software environment.

Task

Select the hardware/software environments of a model.

```
ENVIRONMENT.MODEL_ID = MODEL.ID
```

Task

Select the hardware/software environments on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = ENVIRONMENT.ID
```

FACILITY (Computing/Communication)

The computing/communication facilities are created by using Matrices in the Planning toolset. The system-defined class for the computing/communication facility is: COMPUTING/COMMUNICATION_FACILITY.

There are no system-defined matrices that contain computing/communication facilities. To create computing/communication facilities, a user defined matrix must be created first with one of its classes as computing/communication facility.

Task

Select the computing/communication facilities of a model.

```
FACILITY.MODEL_ID = MODEL.ID
```

Task

Select the computing/communication facilities on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = FACILITY.ID
```

INFORMATION_NEED

The information needs are created by using Matrices in the Planning toolset. The system-defined class for the information need is INFORMATION_NEED. The system-defined matrices that contain information needs are:

- ENTITY_TYPE/INFORMATION_NEED
- INFORMATION_NEED/OBJECTIVE
- INFORMATION_NEED/GOAL
- INFORMATION_NEED/STRATEGY
- INFO_NEED/CRIT._SUCCESS_FACTOR
- CURRENT_INFO_SYS/INFO_NEED
- INFO_NEED_CATEGORY/INFO_NEED
- BUSINESS_FUNCTION/INFO_NEED
- PERFORMANCE_MEASURE/INFO_NEED

Task

Select the information needs of a model.

```
INFORMATION_NEED.MODEL_ID = MODEL.ID
```

Task

Select the information needs on the Y axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_Y.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_Y.OBJECT_ID = INFORMATION_NEED.ID
```

Task

Select the information needs along the Y axis of the matrix INFORMATION_NEED/GOAL. Order by information need as specified in the toolset.

```
SELECT MT.NAME, Y.NAME. UY.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_Y UY,
  INFORMATION_NEED Y
  WHERE
  M.NAME = 'my model name'
  AND UY.MODEL_ID = M.ID
  AND UY.MATRIX_ID = MT.ID
  AND MT.NAME = 'INFORMATION_NEED/GOAL'
  AND UY.OBJECT_ID = Y.ID
ORDER BY MT.NAME, UY.SEQ;
```

ORGANIZAT_UNIT (Organizational Unit)

The organizational units are created by using the Organization Hierarchy function in the Planning toolset. The system-defined class for the organizational unit is ORGANIZATIONAL_UNIT. The system-defined matrices that contain organizational units are:

- BUSINESS_FUNCTION/ORG_UNIT
- INFORMATION_NEED/ORG_UNIT
- PERFORMANCE_MEASURE/ORG_UNIT
- CRIT_SUCCESS_FACTOR/ORG_UNIT
- STRATEGY/ORGANIZATIONAL_UNIT
- ORG_UNIT/CUR_INFO_SYSTEM
- ORG_UNIT/CURRENT_DATA_STORE
- ORGANIZATIONAL_UNIT/LOCATION

Task

Select the organizational units of a model.

```
ORGANIZAT_UNIT.MODEL_ID = MODEL.ID
```

Task

Select the organizational units on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = ORGANIZAT_UNIT.ID
```

Task

Select the organizational units along the X axis of the matrix BUSINESS_FUNCTION/ORG_UNIT. Order by organizational unit as specified in the toolset.

```
SELECT MT.NAME, X.NAME. UX.SEQ
FROM
MODEL M,
MATRIX MT,
MATRIX_USAGE_X UX,
ORGANIZAT_UNIT X
WHERE
M.NAME = ‘my model name’
AND UX.MODEL_ID = M.ID
AND UX.MATRIX_ID = MT.ID
AND MT.NAME = ‘BUSINESS_FUNCTION/ORG_UNIT’
AND UX.OBJECT_ID = X.ID
ORDER BY MT.NAME, UX.SEQ;
```

Task

Select the performance measures for a model.

```
PERFORM_MEASURE.MODEL_ID = MODEL.ID
```

Task

Select the performance measures on the Y axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_Y.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_Y.OBJECT_ID = PERFORM_MEASURE.ID
```

Task

Select the performance measures along the Y axis of the matrix PERFORMANCE_MEASURE/BUS_FUNCTION. Order by performance measure as specified in the toolset.

```
SELECT MT.NAME, Y.NAME. UY.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_Y UY,
  PERFORM_MEASURE Y
  WHERE
  M.NAME = 'my model name'
  AND UY.MODEL_ID = M.ID
  AND UY.MATRIX_ID = MT.ID
  AND MT.NAME ='PERFORMANCE_MEASURE/BUS_FUNCTION'
  AND UY.OBJECT_ID = Y.ID
ORDER BY MT.NAME, UY.SEQ;
```

PERFORM_MEASURE (Performance Measure)

The performance measures are created by using Matrices in the Planning toolset. The system-defined class for the performance measure is PERFORMANCE_MEASURE. The system-defined matrices that contain performance measures are:

- PERFORMANCE_MEASURE/ORG_UNIT
- PERFORMANCE_MEASURE/BUS_FUNCTION
- PERFORMANCE_MEASURE/INFO_NEED

Task

Select the performance measures for a model.

```
PERFORM_MEASURE.MODEL_ID = MODEL.ID
```

Task

Select the performance measures on the Y axis of a matrix.

```
MATRIX.NAME = ‘name of matrix’

AND MATRIX_USAGE_Y.MATRIX_ID = MATRIX.ID

AND MATRIX_USAGE_Y.OBJECT_ID = PERFORM_MEASURE.ID
```

Task

Select the performance measures along the Y axis of the matrix PERFORMANCE_MEASURE/BUS_FUNCTION. Order by performance measure as specified in the toolset.

```
SELECT MT.NAME, Y.NAME. UY.SEQ
FROM
MODEL M,
MATRIX MT,
MATRIX_USAGE_Y UY,
PERFORM_MEASURE Y
WHERE
M.NAME = 'my model name'
AND UY.MODEL_ID = M.ID
AND UY.MATRIX_ID = MT.ID
AND MT.NAME ='PERFORMANCE_MEASURE/BUS_FUNCTION'
AND UY.OBJECT_ID = Y.ID
ORDER BY MT.NAME, UY.SEQ;
```

STRATEGY

The strategies are created by using Matrices in the Planning toolset. The system-defined class for the strategy is STRATEGY. The system-defined matrices that contain strategies are:

- INFORMATION_NEED/STRATEGY
- OBJECTIVE/STRATEGY
- STRATEGY/ORGANIZATIONAL_UNIT

Task

Select the strategies of a model.

```
STRATEGY.MODEL_ID = MODEL.ID
```

Task

Select the strategies on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = STRATEGY.ID
```

Task

Select the strategies along the X axis of the matrix OBJECTIVE/STRATEGY. Order by strategy as specified in the toolset.

```
SELECT MT.NAME, X.NAME. UX.SEQ
  FROM
  MODEL M,
  MATRIX MT,
  MATRIX_USAGE_X UX,
  PERFORM_MEASURE X
  WHERE
  M.NAME = 'my model name'
  AND UX.MODEL_ID = M.ID
  AND UX.MATRIX_ID = MT.ID
  AND MT.NAME ='OBJECTIVE/STRATEGY'
  AND UX.OBJECT_ID = X.ID
ORDER BY MT.NAME, UX.SEQ;
```

TACTIC

The tactics are created by using Matrices in the Planning toolset. The system-defined class for the tactic is TACTIC. There are not any system-defined matrices that contain tactics. To create tactic, a user-defined matrix must be created first with one of its classes as tactic.

Task

Select the tactics for a model.

```
TACTIC.MODEL_ID = MODEL.ID
```

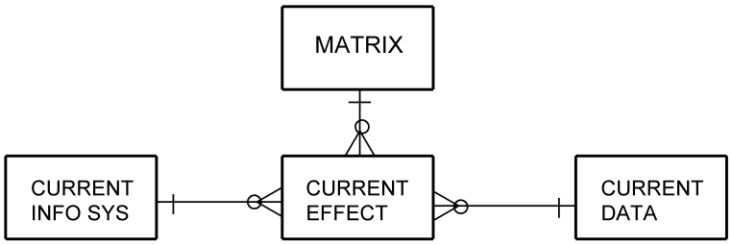
Task

Select the tactics on the X axis of a matrix.

```
MATRIX.NAME = 'name of matrix'
AND MATRIX_USAGE_X.MATRIX_ID = MATRIX.ID
AND MATRIX_USAGE_X.OBJECT_ID = TACTIC.ID
```

CURRENT_EFFECT (Current System/Data Effect)

Current effects specify the actions a current information system is expected to perform on current database or data store.



Task

Select the current effects for each current information system.

```
SELECT I.NAME, E.CREATE, E.READ, E.UPDT, E.DLET, D.NAME
  FROM
  MODEL M,
  CURRENT_DATA D,
  CURRENT_INFO_SYS I,
  CURRENT_EFFECT E
  WHERE
  M.NAME = 'my model name'
  AND E.MODEL_ID = M.ID
  AND E.CUR_DATA_ID = D.ID
  AND E.CUR_INFO_SYS_ID = I.ID;
```

Analysis Objects

Following are the analysis objects:

- Data-related objects
- Activity-related objects
- Intersection-related objects

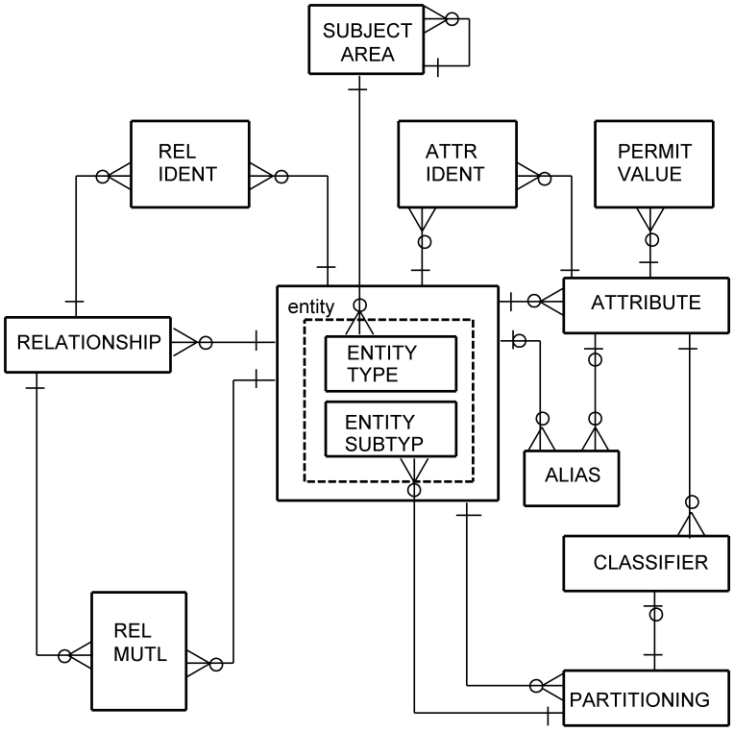
Data-Related Objects

Data-related objects are entered through the Data Modeling Tool (DM).

Data-Related Object	Description
SUBJECT_AREA	Subject area

Data-Related Object	Description
ENTITY_TYPE	Entity type
ATTRIBUTE	Attribute of an entity type or subtype
ATTR_IDENT	Attribute identifier
PERMIT_VALUE	Permitted values of an attribute
ALIAS	Alias for an entity type, subtype, or attribute
PARTITIONING	Partitioning of an entity type or subtype
ENTITY_SUBTYP	Entity subtype
CLASSIFIER	Classifying attribute for a partitioning
RELATIONSHIP	Relationship between two entity types/subtypes
REL_MUTL_EXCL	Relationship mutually exclusive
REL_IDENT	Relationship identifier
IDENTIFIER	Identifier (attribute or relationship)
INTERFACE_TYPE	Interface type
SPEC_TYPE	Specification type
COMPONENT_SPEC	Component specification
COMPONENT_IMPLEM	Component implementation
INTRFCE_TYPE_MDL	Interface type model

Data Related Objects



The Data Modeling Tool allows groupings of entity types and their relationships. These groupings are called Subject Areas (SUBJECT_AREA). A subject area can also include many other subject areas.

Within a Subject Area are many entity types (ENTITY_TYPE). Each entity type can be partitioned into many partitionings (PARTITIONING). Each partitioning consists of many subtypes (ENTITY_SUBTYP). A classifier (CLASSIFIER) associates the partitioning with its classifying attribute.

Each entity type or subtype can contain many attributes (ATTRIBUTE). An attribute can serve as an identifier (IDENTIFIER, ATTR_IDENT) for its parent entity type or subtype. Each attribute can be permitted only certain values, which are known as permitted values (PERMIT_VALUE).

For entity types, subtypes, and attributes, alternate names can be documented. The alternate name is called an alias (ALIAS).

Relationships exist between entity types/subtypes. RELATIONSHIP provides the from and to entity type/subtype, the names of the from and to relationship, and the properties of the from relationship.

Relationships occur in pairs. For example, there is a row for customer places order and a row for order is placed by customer. Relationships can serve as identifiers (IDENTIFIER, REL_IDENT) for entity type/subtypes.

REL_MUTL_EXCL specifies the mutually exclusive relationships of an entity type/subtype. If there are n relationships from an entity type/subtype that are mutually exclusive, then there will be n rows in this table with the same MODEL_ID, ID, and ENTITY_ID. The Relationship_ID in each of these n rows will correspond to the n mutually exclusive relationships. Join RELATIONSHIP_ID with ID of RELATIONSHIP to retrieve the properties.

Objects and Properties.

Data-related objects and properties are created through the DM Tool. Using the Add command in the DM, you create subject areas, entity types, partitionings and entity subtypes. When you join two entity types, you create a relationship. All other data-related objects are created through the Detail command. The Detail command also allows you to specify property values for each of the objects.

Foreign Keys

By using the foreign keys, you can relate tables to other tables. Join tables only when they contain matching foreign key and target columns. Otherwise, the results are meaningless.

Analysis Tasks Related to Data.

The following table directs you to the PI table explanation that contains SQL for each Analysis task listed:

If You Want To	See
Find the attributes of an entity type or subtype	ATTRIBUTE
Find the classifying attribute of a partitioning	PARTITIONING/CLASSIFIER
Find the identifying attribute(s) of an entity type	ATTR_IDENT or IDENTIFIER
Find the entity types in a model	ENTITY_TYPE
Find the relationships between entity types	RELATIONSHIP
Find the identifying relationship of an entity type	REL_IDENT or IDENTIFIER
Find the mutually exclusive relationships of an entity type	REL_MUTL_EXCL

If You Want To	See
Find the permitted values of an attribute	PERMIT_VALUE
Select entity types, classifying attributes, and subtypes	PARTITIONING/CLASSIFIER
Select the mutually exclusive relationships from each of the entity types in a model	REL_MUTL_EXCL
Select the root subject area in a model	SUBJECT_AREA
Select the subject areas and their parent subject areas	SUBJECT_AREA
Select the entity types in a subject area	ENTITY_TYPE
Select the attributes with default permitted values	ATTRIBUTE
Select the derived attributes for a derivation algorithm	ATTRIBUTE
Select the permitted value ranges for an attribute	PERMIT_VALUE
Select the classifying value for a subtype	ENTITY_SUBTYPE

SUBJECT_AREA

Each model has at least one subject area. Each subject area can also include many other subject areas. The PARENT_SUBJ_ID will be 0 for the root subject area.

Task

Select the root subject area for a model.

```
SELECT S.NAME
  FROM
  MODEL M,
  SUBJECT_AREA S
  WHERE
  M.NAME = 'my model name'
  AND S.MODEL_ID = M.ID
  AND S.PARENT_SUBJ_ID = 0;
```

Task

Select all the subject areas and their parent subject areas. Order by parent subject area.

```
SELECT P.NAME, C.NAME
  FROM
  MODEL M,
  SUBJECT_AREA P,
  SUBJECT_AREA C
  WHERE
  M.NAME = 'my model name'
  AND P.MODEL_ID = M.ID
  AND C.PARENT_SUBJ_ID = P.ID
ORDER BY P.NAME;
```

ENTITY_TYPE

Each entity type belongs to a model.

Task

Join an entity type to the model that contains it.

```
ENTITY_TYPE.MODEL_ID = MODEL.ID
```

Task

Join an entity type to the subject area that contains it.

```
ENTITY_TYPE.SUBJECT_AREA_ID = SUBJECT_AREA.ID
```

ATTRIBUTE (Attribute of an Entity Type or Subtype)

When you create an attribute under an entity type or subtype, a foreign key to the entity type or subtype is created in the attribute row.

Task

Join an attribute to its parent entity type or subtype.

```
ATTRIBUTE.PARENT_ENTITY_ID = ENTITY_TYPE.ID or
ATTRIBUTE.PARENT_ENTITY_ID = ENTITY_SUBTYPE.ID or
ATTRIBUTE.PARENT_ENTITY_ID = SYS_ENT_TYPE.ID
```

Task

Select all entity types and their attributes from model 'my model.' Order entity types by name. Order attributes as they are in the diagram.

```
SELECT E.NAME, A.NAME, A.LENGTH, A.DOMAIN, A.SEQ
  FROM
  MODEL M,
  ENTITY_TYPE E,
  ATTRIBUTE A
  WHERE
  M.NAME = 'my model name'
  AND E.MODEL_ID = M.ID
  AND A.PARENT_ENTITY_ID = E.ID
ORDER BY E.NAME, A.SEQ;
```

Task

Join an attribute to its default permitted value.

```
ATTRIBUTE.DEF_PERM_VAL_ID = PERMIT_VALUE.ID
```

Task

Join a derived attribute to its derivation algorithm action block.

```
ATTRIBUTE.ACTION_BLOCK_ID = DERIVATION_ALGOR.ID or
ATTRIBUTE.ACTION_BLOCK_ID = ACTION_BLOCK.ID
```

Note: For more information, see SYS_ENT_TYPE and SYS_ATTRIBUTE.

PARTITIONING/CLASSIFIER

For each partitioning of an entity type or subtype, there is a partitioning object. Within the partitioning object is a foreign key of the parent entity type or subtype.

Each partitioning object is the parent of the entity subtypes. Therefore, within each subtype is a foreign key of the partitioning to which it belongs. The classifier contains foreign keys of both the partitioning and the attribute that serves as the classifier.

Task

Join a partitioning and its classifier to their parent entity type or subtype.

```
PARTITIONING.PARENT_ENTITY_ID = ENTITY_TYPE.ID
```

or

```
PARTITIONING.PARENT_ENTITY_ID = ENTITY_SUBTYP.ID
```

and

```
ENTITY_SUBTYP.PARTITIONING_ID = PARTITIONING.ID
```

and

```
CLASSIFIER.PARTITIONING_ID = PARTITIONING.ID
CLASSIFIER.ATTRIBUTE_ID = ATTRIBUTE.ID
```

Task

Select entity type name, classifying attribute name, and subtype name. Order by entity type name.

```
SELECT E.NAME, A.NAME, S.NAME
  FROM
  MODEL M,
  ENTITY_TYPE E,
  PARTITIONING P,
  CLASSIFIER C,
  ENTITY_SUBTYP S,
  ATTRIBUTE A
  WHERE
  M.NAME = 'my model name'
  AND E.MODEL_ID = M.ID
  AND P.PARENT_ENTITY_ID = E.ID
  AND C.PARTITIONING_ID = P.ID
  AND A.ID = C.ATTRIBUTE_ID
  AND S.PARTITIONING_ID = P.ID
ORDER BY E.NAME;
```

Note: SQL retrieves only those rows for which *all* the criteria are met. This means that if a classifying attribute has not been specified for a partitioning, *no* information on that partitioning will be returned.

PERMIT_VALUE (Permitted Values for an Attribute)

Each attribute has zero or more permitted values.

Task

Join permitted values to their attribute.

```
PERMIT_VALUE.ATTRIBUTE_ID = ATTRIBUTE.ID
```

Task

Join a subtype to its classifying value.

```
ENTITY_SUBTYP.PERMIT_VALUE_ID = PERMIT_VALUE.ID
```

Task

Select all attributes with permitted values. Order by attribute name.

```
SELECT A.NAME, L.VALUE
  FROM
 MODEL M,
  ATTRIBUTE A,
  PERMIT_VALUE P,
  PERMIT_VALUE_LOW L
  WHERE
  M.NAME = 'my model name'
  AND A.MODEL_ID = M.ID
  AND A.ID = P.ATTRIBUTE_ID
  AND P.LOW_VALUE = L.ID
ORDER BY A.NAME;
```

Task

Select all attributes with a permitted value range. Order by attribute name.

```
SELECT A.NAME, L.VALUE, H.VALUE
  FROM
  MODEL M,
  ATTRIBUTE A,
  PERMIT_VALUE P,
  PERMIT_VALUE_LOW L,
  PERMIT_VALUE_HI H
  WHERE
  M.NAME = 'my model name'
  AND A.MODEL_ID = M.ID
  AND A.ID = P.ATTRIBUTE_ID
  AND P.LOW_VALUE = L.ID
  AND P.HIGH_VALUE = H.ID
ORDER BY A.NAME;
```

Note: For permitted values that are not a range, the low value and high value are equal.

ATTR_IDENT (Attribute Identifier)

An entity type or subtype can have zero or more attributes as identifiers.

Task

Join an attribute identifier with its entity type.

```
ATTR_IDENT.ENTITY_ID = ENTITY_TYPE.ID
ATTR_IDENT.ATTRIBUTE_ID = ATTRIBUTE.ID
```

Task

Select entity type name and attribute name for those attributes that are identifiers. Order by entity type name.

```
SELECT E.NAME, A.NAME
  FROM
  MODEL M,
  ENTITY_TYPE E,
  ATTR_IDENT I,
  ATTRIBUTE A
  WHERE
  M.NAME = 'my model name'
  AND E.MODEL_ID = M.ID
  AND A.PARENT_ENTITY_ID = E.ID
  AND I.ENTITY_ID = E.ID
  AND I.ATTRIBUTE_ID = A.ID
ORDER BY E.NAME;
```

Note: The table IDENTIFIER contains both attribute identifiers and relationship identifiers. Therefore, the table ATTR_IDENT can be replaced with the table IDENTIFIER.

RELATIONSHIP (Relationship of an Entity Type or Subtype)

If a relationship exists between Entity Type A and Entity Type B, an inverse relationship exists between Entity Type B and Entity Type A. The RELATIONSHIP table has one entry for the relationship and another for the inverse relationship.

Task

Retrieve the properties of a relationship.

```
RELATIONSHIP.SOURCE_ENTITY_ID = ENTITY_TYPE.ID
```

To retrieve the from and to entity types.

```
RELATIONSHIP.SOURCE_ENTITY_ID = ENTITY_TYPE.ID
RELATIONSHIP.DEST_ENTITY_ID = ENTITY_TYPE.ID
```

Task

Select the from and to entity type names and the from and to relationship names. Order by from and to entity type name.

```
SELECT F.NAME, T.NAME, R.NAME, R.NAME_INVERSE
  FROM
  MODEL M,
  ENTITY_TYPE F,
  ENTITY_TYPE T,
  RELATIONSHIP R
  WHERE
  M.NAME = 'my model name'
  AND F.MODEL_ID = M.ID
  AND T.MODEL_ID = M.ID
  AND R.SOURCE_ENTITY_ID = F.ID
  AND R.DEST_ENTITY_ID = T.ID
ORDER BY F.NAME, T.NAME;
```

REL_IDENT (Relationship Identifier)

An entity type or subtype can have zero or more relationships as identifiers.

Task

Join a relationship identifier with the entity type it identifies.

```
REL_IDENT.ENTITY_ID = ENTITY_TYPE.ID
REL_IDENT.RELATIONSHIP_ID = RELATIONSHIP.ID
```

Note: The table IDENTIFIER contains both attribute identifiers and relationship identifiers. Therefore, the table REL_IDENT can be replaced with the table IDENTIFIER.

REL_MUTL_EXCL (Mutually Exclusive Relationships)

Relationships from an entity type/subtype can be mutually exclusive. That is, an occurrence of only one of the set of relationships can exist at any given time. To retrieve this information.

```
REL_MUTL_EXCL.ENTITY_ID = ENTITY_TYPE.ID
REL_MUTL_EXCL.RELATIONSHIP_ID = RELATIONSHIP.ID
```

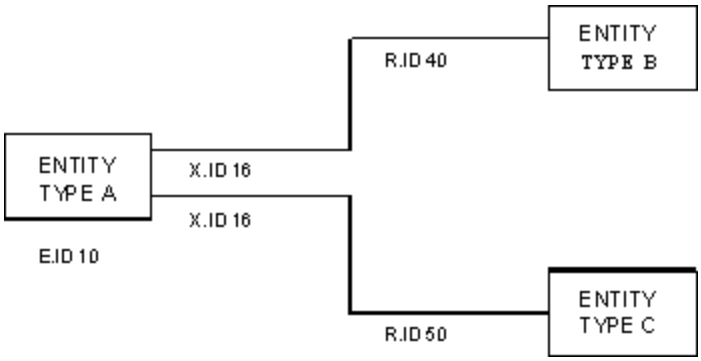
Task

Select the mutually exclusive relationships from each of the entity types in a model.

```
SELECT E.NAME, R.NAME, X.ID, X.ENTITY_ID
  FROM
  MODEL M,,
  ENTITY_TYPE E,
  REL_MUTL_EXCL X,
  RELATIONSHIP R
  WHERE
  M.NAME = 'my model name'
  AND E.MODEL_ID = M.ID
  AND R.MODEL_ID = M.ID
  AND X.ENTITY_ID = E.ID
  AND X.RELATIONSHIP_ID = R.ID
ORDER BY X.ID, X.ENTITY_ID;
```

Those relationships for which X.ID and X.Entity_ID are th

e same are mutually exclusive relationships of entity type E.Name. For example, in the illustration Mutually Exclusive Relationship, Entity Type A has a mutually exclusive relationship with Entity Type B and Entity Type C.



If M.ID is 28, the records selected from the REL_MUTL_EXCL table might look like those in the next table.

In the following table, the object IDs are the same and the Entity Type IDs are the same (16 and 10, respectively):

Model_ID	Table Name	ID	Entity_ID	Relationship_ID
28	REL_MUTL_EXCL	16	10	40
28	REL_MUTL_EXCL	16	10	50

Activity-Related Objects

Activity-related objects are entered through the Activity Hierarchy Diagram (AHD), Activity Dependency Diagram (ADD), and Process Action Diagram (PAD). The process-related objects are:

Object	Definition
FUNCTION_DEF	Function Definition
PROCESS_DEF	Process Definition
ACTIV_USAGE	Activity (Function or Process) Usage

The following table describes process dependency objects:

Object	Definition
ADD_ATOM_DEP	ADD Atomic Dependency
ADD_CLOSURE	ADD Closure
ADD_DEPENDNCY	ADD Dependency
ADD_EXT_FLOW	ADD External Flow
ADD_MUTL_EXCL	ADD Mutually Exclusive
ADD_PARALLEL	ADD Parallel
PDD_ATOM_DEP	PDD Atomic Dependency (see the following note)
PDD_CLOSURE	PDD Closure (see the following note)
PDD_DEPENDNCY	PDD Dependency (see the following note)
PDD_EXT_FLOW	PDD External Flow (see the following note)
PDD_MUTL_EXCL	PDD Mutually Exclusive (see the following note)
PDD_PARALLEL	PDD Parallel (see the following note)
XT_EVNT_USAGE	External Event Usage
XT_OBJ_USAGE	External Object Usage
XTERNL_EVENT	External Event
XTERNL_OBJECT	External Object

Note: To be deleted in a future release.

The following table describes action diagram objects:

Object	Definition
ACTN_BLK_USE	Action Block Usage
ACTION_BLOCK	Action Block Definition
BAA_ACTN_BLK	Analysis Action Block Definition
DERIVATION_ALGOR	Derivation Algorithm Action Block Definition
PAD_CREATE	Action Diagram Create Statement
PAD_DELETE	Action Diagram Delete Statement
PAD_FUNCTION	System-Supplied Function
PAD_READ	Action Diagram Read Statement
PAD_UPDATE	Action Diagram Update Statement
PAD_SET_ATTR	Action Diagram Set Attribute Statement
USE_DATA_RTND	Action Diagram Use Statement, Data Returned
USE_DATA_SENT	Action Diagram Use Statement, Data Sent

Definition Versus Usage

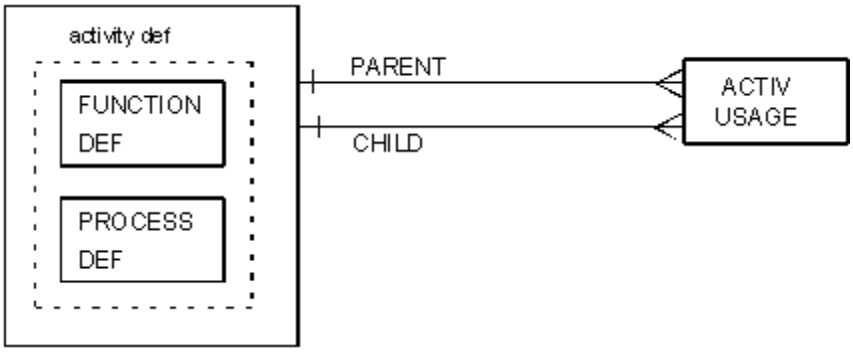
Each time an object is created, a definition for the object is also created. The definition contains information, such as name, that is unrelated to how or where the object is used. Each time an object is used (or reused) a usage is created for it and related to the definition. An object has only one definition, but it can have many usages.

Function/Process Hierarchy Objects

Each model has one root function. The root function decomposes into activity definitions: function definitions (FUNCTION_DEF) and process definitions (PROCESS_DEF). The root can decompose into either many functions or many processes. Each function definition can decompose into either functions or processes. Each process definition can decompose into other processes.

For each parent activity (function or process) and child activity combination, there is an activity usage (ACTIV_USAGE) that relates them. If a child is reused somewhere else, another usage is created, but only the one definition exists.

The Activity Hierarchy Diagram (AHD) adds function definitions, process definitions, and activity usages.



Action Diagram Objects

When an action diagram is created for an elementary process, an action block (ACTION_BLOCK, BAA_ACTN_BLK) is created.

An action block is also created when an action diagram is created to be used by another action diagram (when an action diagram contains a USE statement, as for a derivation algorithm (DERIVATION_ALGOR).

If the action diagram represents an elementary process, it is associated with that process definition.

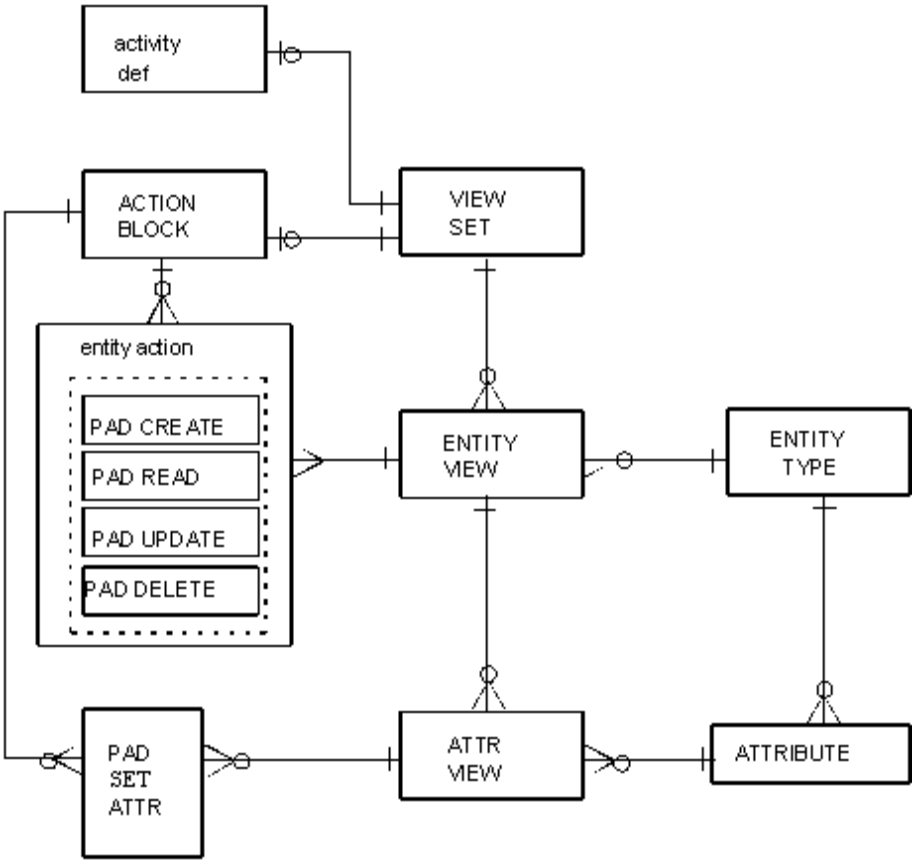
System-Supplied Functions

The system-supplied functions (PAD_FUNCTION) are created automatically when a model is created. These functions are available for use in action blocks in both Analysis and Design. These functions are used to convert data from one domain to another, as well as to help in stringing and unstringing data.

Entity Actions

When an action diagram contains an entity action (PAD_CREATE, PAD_READ, PAD_UPDATE, or PAD_DELETE), the entity action is associated with an entity view (ENTITY_VIEW). The entity view is associated with the entity type and with the Entity Action view set (VIEW_SET).

The following graphic is a pictorial illustration.



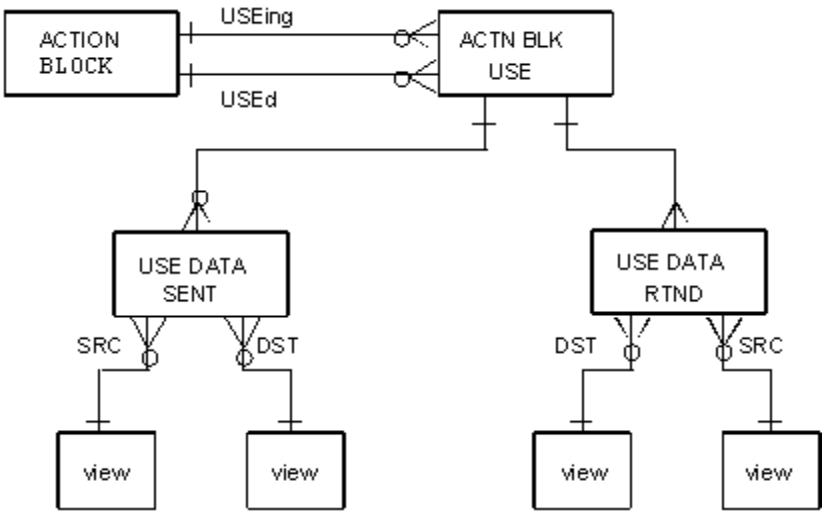
The view set is associated with the action block. For more information on view sets, see Intersection-Related Objects.

When an action diagram contains a SET statement (PAD_SET_ATTR), the SET action is associated with an attribute view (ATTR_VIEW). The attribute view is associated with the attribute being set and with the associated entity view.

USE Action

When an action diagram contains a USE statement, an action block usage (ACTN_BLK_USE) is created. The usage is associated with the action block that contains the USE statement (parent), and with the action block that referenced in the USE statement (child).

The following graphic is a pictorial illustration.



View Matching

View matching is part of adding the USE statement. This process associates views in the USEing, or calling, action block with views in the USEd, or called, action block. Both the sent data (USE_DATA_SENT) and the returned data (USE_DATA_RTND) are matched.

The data sent from the USEing action block is referenced in the USE statement in the Which IMPORTS clause. Data sent is also referenced in the USEd action block in the Import Views of the view definitions. To find data sent, use the USE_DATA_SENT object.

The data returned from the USEd action block is referenced in the USE statement in the Which EXPORTS clause. It is also referenced in the USEd action block in the Export Views section of the view definitions. To find data returned, use the USE_DATA_RTND object.

Objects and Properties

Activity-related objects are entered through the AHD, ADD, and PAD tools. Using the Add command in the AHD, you add function definitions, process definitions, and activity usages. Using the same command in the ADD, you can also add external objects and external events.

The following list is a definition of related terms.

- **Dependencies**—When you join one activity to another, you create a dependency and an atomic dependency. When joining a dependency to an activity, you create a mutually exclusive control construct, a parallel control construct, or a closure control construct, and one or more atomic dependencies.
- **Usages**—When you join an external object or external event to a process, you create a usage of the external object or event, an atomic dependency, and a dependency.
- **Data Flows**—When you detail a dependency between an activity and an external object and you associate it with a data view, you add an external data flow.
- **Action Diagram Statements and References to Views**—When adding a statement in the action diagram, you add an object for that statement (CREATE, READ, SET, and so forth) and create references to existing entity views or attribute views.

Analysis Tasks Related to Processes

The following table directs you to the PI table explanation that contains SQL for Analysis tasks related to processes.

If You Want To	See
Select an activity and all its dependent activities	FUNCTION_DEF PROCESS_DEF ACTIV_USAGE_
Select the functions and processes in a model's activity hierarchy	FUNCTION_DEF PROCESS_DEF ACTIV_USAGE_
Select all dependencies between processes	ADD_DEPENDNCY/ ADD_ATOM_DEP
Select all dependencies from a process to an external object	ADD_DEPENDNCY/ ADD_ATOM_DEP
Select all elementary processes that create an entity type	PAD_CREATE VIEW_SET ENTITY_VIEW
Select a process definition and its action block	PROCESS_DEF/ACTION_BLOCK
Select the derivation algorithms in a model	DERIVATION_ALGOR

If You Want To	See
Select the system-supplied functions in a model	PAD_FUNCTION

FUNCTION_DEF, PROCESS_DEF, and ACTIV_USAGE

A FUNCTION_DEF is a function definition. PROCESS_DEF is a process definition, and ACTIV_USAGE is activity usage of both functions and processes.

Each time an activity is used, it is within a parent (hierarchy). Each model has one root function. The next level activity (function or process) is within the root, and so on.

When an activity is created, a function definition or process definition is created. Each time an activity is used (that is, a usage of it is added in the AHD or ADD), a usage of that activity is created. Therefore, you can retrieve an activity and all its dependent activities. The following table details the ways to retrieve.

```
ACTIV_USAGE.PARENT_ACTIV_ID = PROCESS_DEF.ID
ACTIV_USAGE.CHILD_ACTIV_ID = PROCESS_DEF.ID
```

Task

Select all the functions and processes in the activity hierarchy. Order in hierarchical sequence.

- If parent is function, child is function.

```
SELECT P.NAME, C.NAME, U.SEQH
  FROM
  MODEL M,
  FUNCTION_DEF P,
  FUNCTION_DEF C,
  ACTIV_USAGE U
  WHERE
  MODEL.NAME = 'my model name'
  AND P.MODEL_ID = M.ID
  AND C.MODEL_ID = M.ID
  AND U.PARENT_ACTIV_ID = P.ID
  AND U.CHILD_ACTIV_ID = C.ID
UNION
```

- If parent is function, child is process.

```
SELECT P.NAME, C.NAME, U.SEQH
  FROM
  MODEL M,
  FUNCTION_DEF P,
  PROCESS_DEF C,
  ACTIV_USAGE U
  WHERE
  MODEL.NAME = 'my model name'
  AND P.MODEL_ID = M.ID
  AND C.MODEL_ID = M.ID
  AND U.PARENT_ACTIV_ID = P.ID
  AND U.CHILD_ACTIV_ID = C.ID
UNION
```

- If parent is process, child is process.

```
SELECT P.NAME, C.NAME, U.SEQH
  FROM
  MODEL M,
  PROCESS_DEF P,
  PROCESS_DEF C,
  ACTIV_USAGE U
  WHERE
  MODEL.NAME = 'my model name'
  AND P.MODEL_ID = M.ID
  AND C.MODEL_ID = M.ID
  AND U.PARENT_ACTIV_ID = P.ID
  AND U.CHILD_ACTIV_ID = C.ID
ORDER BY 3;
```

Task

Join a process definition with its action block.

```
PROCESS_DEF.ACTION_BLOCK_ID = ACTION_BLOCK.ID
```

DERIVATION_ALGOR (Derivation Algorithm)

A derivation algorithm is used to calculate a derived attribute.

Task

Join a derivation algorithm to the model that uses it.

```
DERIVATION_ALGOR.MODEL_ID = MODEL.ID
```

PAD_FUNCTION (System-Supplied Functions)

Each model contains the system functions.

Task

Join a system function to the model that uses it.

```
PAD_FUNCTION.MODEL_ID = MODEL.ID
```

ADD_DEPENDENCY/ADD_ATOM_DEP

The atomic dependency is the key to selecting dependencies between activity-related objects. It contains an import usage and an export usage.

The import usage identifies the usage of the from object, and the export usage identifies the usage of the to object. To retrieve the name of the object, join the usage with the object's definition. All the atomic dependencies for a given dependency are collected by associating them with a dependency object, which also has a name.

In the following illustrations, from and to can be activity usage (ACTIV_USAGE), external object usage (XT_OBJ_USAGE), external event usage (XT_EVNT_USAGE), parallel construct (ADD_PARALLEL), mutually exclusive construct (ADD_MUTL_EXCL), or closure (ADD_CLOSURE), with the same constraints as are imposed by the ADD.

Task

Select all dependencies between processes as defined in the ADD.

```
SELECT D.ID, FD.NAME, TD.NAME
  FROM
  MODEL M,
  PROCESS_DEF FD,
  ACTIV_USAGE F,
  PROCESS_DEF TD,
  ACTIV_USAGE T,
  ADD_DEPENDNCY D,
  ADD_ATOM_DEP A
  WHERE
  MODEL.NAME = 'my model name'
  AND F.MODEL_ID = M.ID
  AND T.MODEL_ID = M.ID
  AND D.MODEL_ID = M.ID
  AND A.ADD_DEPENDNCY_ID = D.ID
  AND A.EXPORT_USAGE_ID = F.ID
  AND A.IMPORT_USAGE_ID = T.ID
  AND F.CHILD_ACTIV_ID = F.ID
  AND T.CHILD_ACTIV_ID = TD.ID
ORDER BY D.ID, FD.NAME, TD.NAME;
```

Task

Select all dependencies from a process to an external object as defined in the PDD.

```
SELECT D.ID, FD.NAME, TD.NAME
  FROM
  MODEL M,
  PROCESS_DEF FD,
  ACTIV_USAGE F,
  XTERNL_OBJECT TD,
  XT_OBJ_USAGE T,
  ADD_DEPENDNCY D,
  ADD_ATOM_DEP A
  WHERE
  MODEL.NAME = 'my model name'
  AND F.MODEL_ID = M.ID
  AND T.MODEL_ID = M.ID
  AND D.MODEL_ID = M.ID
  AND A.ADD_DEPENDNCY_ID = D.ID
  AND A.EXPORT_USAGE_ID = F.ID
  AND A.IMPORT_USAGE_ID = T.ID
  AND F.CHILD_ACTIV_ID = FD.ID
  AND T.XTERNL_OBJECT_ID = TD.ID
ORDER BY D.ID, FD.NAME, TD.NAME;
```

Note: The tables ADD_DEPENDNCY and ADD_ATOM_DEP contain the same information as tables PDD_DEPENDNCY and PDD_ATOM_DEP.

PAD_CREATE, VIEW_SET, ENTITY_VIEW

Tables are provided for only those action diagram statements that act on data and use other action diagrams. This allows you to determine which action diagrams act on what data, and what data an action diagram passes to a USEd action diagram.

You can use the view set (VIEW_SET), entity view (ENTITY_VIEW), and attribute view (ATTR_VIEW) tables to determine what data the action diagram references. You can also use the action diagram print facility to list the text of the action diagram.

Task

Select all the elementary processes that create an entity type. Order by process name.

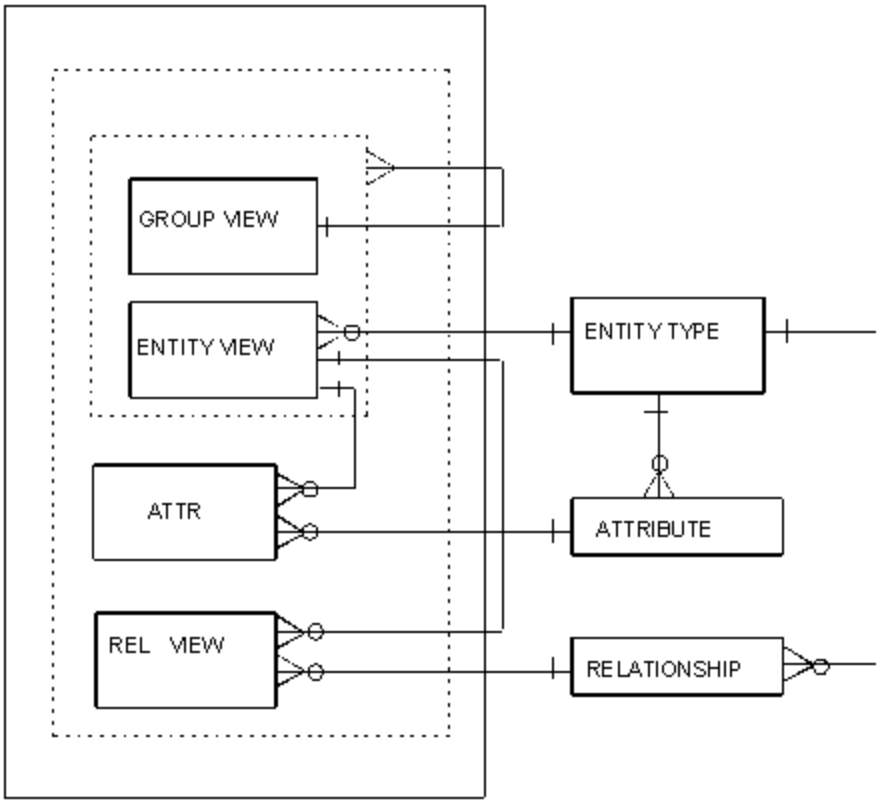
```
SELECT P.NAME, V.NAME, E.NAME
  MODEL M,
  PROCESS_DEF P,
  VIEW_SET S,
  ENTITY_VIEW V,
  ENTITY_TYPE E,
  PAD_CREATE C
  WHERE
  M.NAME = 'my model name'
  AND P.MODEL_ID = M.ID
  AND E.MODEL_ID = M.ID
  AND S.ACTIVITY_ID = P.ID
  AND V.PARENT_ID = S.ID
  AND V.ENTITY_ID = E.ID
  AND C.ENTITY_VIEW_ID = V.ID
ORDER BY P.NAME;
```

Intersection-Related Objects

Intersection-related objects go between or across the data and processing objects. They are either created for you as part of model creation or entered through View Maintenance, the Detail command, or the Entity Life Cycle Diagram. The following table gives the details:

Object	Definition
ATTR_VIEW	Attribute View
DESC	Description
ENT_ST_TRANS_USE	Entity State Transition Usage
ENTITY_ST_TRANS	Entity State Transition
ENTITY_VIEW	Entity View
EXPECT_EFFECT	Expected Effect
GROUP_VIEW	Group View
MODEL	Model
REL_VIEW	Relationship View
VIEW_SET	View Set

The most important of the intersection objects are the views. Created through view maintenance, views specify the data that can be used by a given process. Only those data objects specified in a view of a process can be used by that process (in an action diagram, for example).



Analysis Tasks Related to Intersection Objects

The following table directs you to the PI table explanation that contains SQL for Analysis tasks related to intersection objects:

If You Want To	See
List all group views and entity views of all processes in a model	ENTITY_VIEW
Determine the expected effects for each process in a model	EXPECT_EFFECT

VIEW_SET (View Set)

The view set is a means of collecting views. In the toolset, you specify that a view is import, export, local, or entity action. Each process has a view set for each of these categories (a total of four).

ENTITY_VIEW (Entity View)

An entity view is a selection of attributes or relationships from a single occurrence of an entity type. Entity views are not hierarchical.

Task

List all the group views and entity views of all processes. If the view is an entity view, list the entity type name also.

Note: The string of asterisks is 32 columns in length, corresponding to the length of Entity_Type Name.

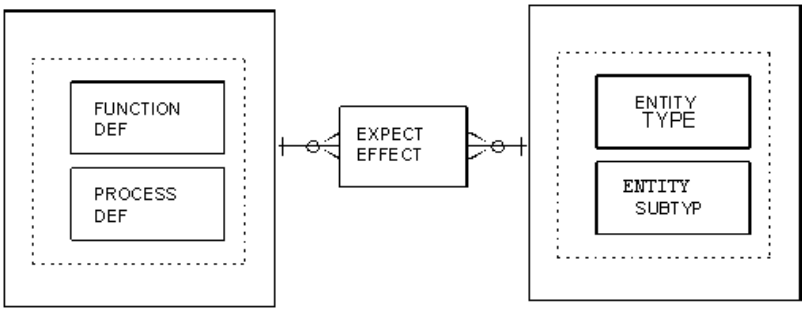
```
SELECT P.NAME, VS.TYPE, VW.TBNAME, VW.NAME, VW.SEQH, E.NAME
  FROM
  MODEL,
  PROCESS_DEF P,
  VIEW_SET VS,
  ENTITY_VIEW VW,
  ENTITY_TYPE E
  WHERE
  MODEL.NAME = 'my model name'
  AND P.MODEL_ID = MODEL.ID
  AND E.MODEL_ID = MODEL.ID
  AND VS.ACTIVITY_ID = P.ID
  AND VW.VIEW_SET_ID = VS.ID
  AND VW.ENTITY_ID = E.ID
UNION
SELECT P.NAME, VS.TYPE, VW.TBNAME, VW.NAME, VW.SEQH,
  '********************************'
  FROM
  MODEL,
  PROCESS_DEF P,
  VIEW_SET VS,
  GROUP_VIEW VW
  WHERE
  MODEL.NAME = 'my model name'
  AND P.MODEL_ID = MODEL.ID
  AND VS.ACTIVITY_ID = P.ID
  AND VW.VIEW_SET_ID = VS.ID
ORDER BY 1, 2, 5;
```

REL_VIEW (Relationship View)

The use of relationships in information views is known as *relationship views*.

EXPECT_EFFECT (Expected Effects)

Expected effects specify the entity actions a process is expected to perform on entity types.



Task

List the expected effects for each process in a model.

```
SELECT P.NAME, F.CREATE, F.READ, F.UPDT, F.DLET, E.NAME
FROM
MODEL M,
PROCESS_DEF P,
EXPECT_EFFECT F,
ENTITY_TYPE E
WHERE
MODEL.NAME = 'my model name'
AND P.MODEL_ID = M.ID
AND E.MODEL_ID = M.ID
AND F.ACTIVITY_ID = P.ID
AND F.ENTITY_ID = E.ID;
```

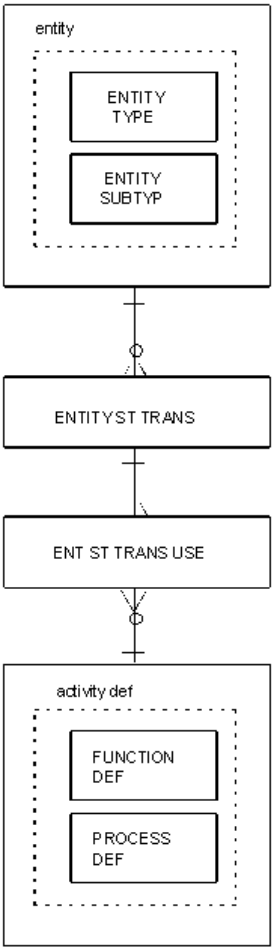
ENTITY_ST_TRANS (Entity State Transition)

Task

An entity state transition is the passage from one stage of an entity type's life cycle to another stage. Entity state transitions are represented as arrows in the Entity Life Cycle Diagram.

ENT_ST_TRANS_USE (Entity State Transition Usage)

Each time an entity state transition is used by a function or process, an entity state transition usage is created.



Design Objects

Business System Design objects are entered using the Business System Definition function of Analysis, the Dialog Design tool, the Screen Design tool, the Procedure Action Diagram, or the System Defaults facility.

Business System Definition Objects

Business System Definition objects are created by the Business System Definition function in the Analysis Toolset by selecting the New command.

The following table defines the various business objects.

Object	Definition
BUSINESS_SYS	Business System
BUSINESS_PROC	Business Procedure
BUS_PROC_STEP	Business Procedure Step
BUS_SYS_SCOPE	Business System Scope
BUS_PROC_SCOPE	Business Procedure Scope
SYS_ENT_TYPE	System Entity Type (Work Attribute Set)
SYS_ATTRIBUTE	System Attribute

BUSINESS_SYS (Business System)

A model can have many Business Systems. Each Business System decomposes into a set of Business Procedures. Business Procedures represent the implementation of Business Processes defined during Analysis.

The structure that represents the defaults associated with a Business System are discussed under the heading, Business System Defaults.

BUSINESS_PROC (Business Procedure)

Business Procedures are created using the Add command in the Dialog Flow Diagram and selecting the option to add a procedure. The Dialog Flow Diagram shows only the procedures and procedure steps that are part of the current Business System.

The name of the procedure can be updated using the Detail command and selecting the Name option.

BUS_PROC_STEP (Business Procedure Step)

When a new Business Procedure is added to the Business System, the system automatically creates an initial procedure step. Additional procedure steps are created by using the Add command of the Dialog Flow Diagram and selecting the option to add a procedure step.

BUS_SYS_SCOPE (Business System Scope)

The Business System Scope table represents the implementation of an Elementary Process by a Business System.

The Scope of a Business System is defined by the scoping facility within the Business System Definition function.

BUS_PROC_SCOPE (Business Procedure Scope)

The Business Procedure Scope table represents the implementation of an Elementary Process by a Business Procedure.

The scope of a Business Procedure is defined by detailing the procedure and selecting Processes Implemented.

SYS_ENT_TYPE (System-Defined Entity Type)

The system-defined entity type is created automatically when a model is created. This entity type is also known as a work attribute set (for example, System-supplied, $IEF). Most work attributes are stored in the ATTRIBUTE table. However, attributes for the $IEF work attribute set are stored in SYS_ATTRIBUTE.

SYS_ATTRIBUTE (System-Defined Attribute)

The system-defined attributes are created automatically for the system-defined entity type $IEF when a model is created. They are used to represent the system-defined attributes, for example, Trancode, System Date, System Time, and so forth.

Dialog Flow Diagram Objects

The PI tables discussed in this section are primarily maintained using the Dialog Flow Diagram (DLG) and its subordinate property entry panels.

The following table defines various diagram objects.

Object	Definition
DIALOG_FLOW	Dialog Flow
DLG_FLWS_EXST	Dialog Flows on Exit State
LNK_RTNS_EXST	Link Returns on Exit State
DLG_SETS_CMD	Dialog Flow Sets Command
LNK_RTNS_CMD	Link Returns Command

Object	Definition
DLG_DATA_SENT	Dialog Flow Sends Data
LNK_DATA_RTND	Link Data Returned
LOCAL_PF_KEY	Procedure Step PF Key Definitions
EXIT_STATE_US	Exit State Usage

Design Tasks Related to Dialog Design Objects

The following table directs you to the PI table explanation that contains SQL for Design tasks related to Dialog Flow Diagram objects:

If You Want To	See
Determine local PF key definitions for a specific procedure step	LOCAL_PF_KEY
Determine the command set by a specific local PF key	LOCAL_PF_KEY
Determine the system-defined PF key definition overridden by the local PF Key definition	LOCAL_PF_KEY
Determine the source and destination procedure steps for a dialog flow	DIALOG_FLOW
Select the command that is set by a specific dialog flow	DLG_SETS_CMD
Select the command that is set by a specific return flow	LNK_RTNS_CMD
Select the exit state that causes a specific dialog flow	DLG_FLWS_EXST
Select the exit state that causes a specific return flow	LNK_RTNS_EXST
Select all local PF key definitions for a specific Business System	LOCAL_PF_KEY
Select all views of a procedure step that provide data when returning from a flow	LNK_DATA_RTND
Select all views of a procedure step that provide data in a dialog flow	DLG_DATA_SENT

DIALOG_FLOW

The Dialog Flow table represents the flow between procedure steps.

A Dialog Flow is created by joining two procedure steps. The first procedure step is assumed to be the Source procedure step and the second, the Destination procedure step.

The type of dialog flow defaults to Transfer. This can be changed by detailing the flow line.

On the same panel, the user can specify Display First or Execute First for the transfer action. If the flow is a Link, the user can specify the same for the return action.

Task

Join the Dialog Flow table with the Procedure Step table to determine the source and destination procedure steps for a specific Dialog Flow.

```
DIALOG_FLOW.INITS_P_STEP_ID = SRC.ID
DIALOG_FLOW.INITBY_P_STEP_ID = DST.ID
```

Where SRC and DST refer to the BUS_PROC_STEP table.

When you put Business System name in quotes, you must supply the underscores in column names.

Task

Select the source and destination procedure steps for a dialog flow. Order by source and destination names.

```
SELECT SRC.NAME, DLG.TYPE, DST.NAME
  FROM
  MODEL M,
  BUSINESS_SYS SYS,
  BUSINESS_PROC BP,
  BUS_PROC_STEP SRC,
  DIALOG_FLOW DLG,
  BUS_PROC_STEP DST
  WHERE
  M.NAME = 'my model name'
  AND SYS.NAME = 'my business system name'
  AND SYS.MODEL_ID = M.ID
  AND SRC.BUS_PROCEDURE_ID = BP.ID
  AND BP.BUSINESS_SYS_ID = SYS.ID
  AND DLG.INITS_P_STEP_ID = SRC.ID
  AND DLG.INITBY_P_STEP_ID = DST.ID
ORDER BY SRC.NAME, DST.NAME
```

DLG_FLWS_EXST (Dialog Flows on Exit State)

The Dialog Flows On Exit State table represents the intersection of exit states and Dialog Flows. It records which Dialog Flows flow on which Exit States.

This table is maintained by detailing a dialog flow, selecting the Flows On option, and selecting the exit state that should result in the flow.

Task

Select the exit state that causes a specific dialog flow.

```
DLG_FLWS_EXST.DIALOG_FLOW_ID = DIALOG_FLOW.ID
AND DLG_FLWS_EXST.EXIT_STATE_ID = EXIT_STATE.ID
```

LNK_RTNS_EXST (Link Returns on Exit State)

The Link Returns On Exit State table represents the intersection of exit states and dialog flows. It records which links return on which Exit States.This table is maintained by detailing a Link dialog flow, selecting the Returns On option, and selecting the exit state that should result in the flow.

Task

Select the exit state that causes a specific return flow.

```
LNK_RTNS_EXST.DIALOG_FLOW_ID = DIALOG_FLOW.ID
AND LNK_RTNS_EXST.EXIT_STATE_ID = EXIT_STATE.ID
```

DLG_SETS_CMD (Dialog Flow Sets Command)

The Dialog Flow Sets Command table represents the intersection of Commands and Dialog Flows. It records the Command set by a specific dialog flow. This table is maintained by detailing a dialog flow, selecting the Properties command option, and selecting the command that should be set by the flow.

Task

Select the command that is set by a specific dialog flow.

```
DLG_SETS_CMD.DIALOG_FLOW_ID = DIALOG_FLOW.ID
AND DLG_SETS_CMD.COMMAND_ID = COMMAND.ID
```

LNK_RTNS_CMD (Link Returns Command)

The Link Returns On Exit State table represents the intersection of exit states and dialog flows. It records which links return on which Exit States.

This table is maintained by detailing a Link dialog flow, selecting the Returns On option, and selecting the exit state that should result in the flow.

Task

Select the command, which is set by a specific return flow.

```
LNK_RTNS_CMD.DIALOG_FLOW_ID = DIALOG_FLOW.ID
AND LNK_RTNS_CMD.COMMAND_ID = COMMAND.ID
```

DLG_DATA_SENT (Dialog Flow Sends Data)

The Dialog Flow Sends Data table represents the data that is sent from one procedure step to another during a link or transfer. It is identified by the dialog flow and identifies the source of the data sent (in an export view of the from procedure step) and the destination of the data sent (in an import view of the to procedure step). It is created when, in the Dialog Flow Diagram, you select Detail, a dialog flow line, and Data Sent from the panel.

Task

Select all views of a procedure step that provide data in a dialog flow, their destination views in the flowed-to procedure step, and the associated entity types.

```
SELECT ...
FROM ...
DIALOG_FLOW DLG,
DLG_DATA_SENT DDS,
BUS_PROC_STEP PSF, – from procedure step
VIEW_SET VSF, – from view set
ENTITY_VIEW EVF, – from entity view
ENTITY_TYPE ETF, – from entity type
BUS_PROC_STEP PST, – to procedure step
VIEW_SET VST, – to view set
ENTITY_VIEW EVT, – to entity view
ENTITY_TYPE ETT, – to entity type
WHERE ...
AND DDS.DIALOG_FLOW_ID = DLG.ID
AND EVF.ID = DDS.SRC_DATA_VIEW_ID
AND ETF.ID = EVF.ENTITY_ID
AND VSF.ID = EVF.VIEW_SET_ID
AND PSF.ID = VSF.ACTIVITY_ID
AND EVT.ID = DDS.DST_DATA_VIEW_ID
AND ETT.ID = EVT.ENTITY_ID
AND VST.ID = EVT.VIEW_SET_ID
AND PST.ID = VST.ACTIVITY_ID;
LNK_DATA_RTND (Link Data Returned)
```

The Link Data Returned table represents the data that is returned from a linked-to procedure step when it returns to a linked-from procedure step. It is identified by the dialog flow and identifies the source of the data returned (in an export view of the linked-to procedure step that is returning) and the destination of the data returned (in an import view of the linked-from procedure step). It is created in the Dialog Flow Diagram, when you select a dialog flow line representing a link, Detail, and Data Returned.

LOCAL_PF_KEY (Local PF Key Definition)

The local PF Key Definition table represents the use of PF keys by a specific procedure step. The table represents a three-way intersection between procedure step, command and system PF key.

Task

Determine the local PF key definitions for a specific procedure step.

```
LOCAL_PF_KEY.BUS_PROC_STEP_ID = BUS_PROC_STEP.ID
```

Task

Retrieve the command set by a specific local PF Key.

```
LOCAL_PF_KEY.COMMAND_ID = COMMAND.ID
```

Screen Design Tool Objects

The following table defines various design objects.

Object	Definition
SCREEN_DEF	Screen Definition
SCREEN_TMPLT	Screen Template
TMPLT_USAGE	Template Usage
SCRN_FLD_LIT	Screen Field, Literal
SCRN_SYS_DEF	Screen System-defined Variable
SCRN_VAR_DEF	Screen Variable Definition
SCRN_VAR_IO	Screen Variable Input/Output
SCRNFLD_VAR	Screen Field, Variable
SCRN_FLD_VARP	Screen Field, Variable, Normal Video Attrs
SCRN_FLD_VARE	Screen Field, Variable, Error Video Attrs

Object	Definition
CSTM_EDT_PTRN	Custom Edit Pattern
SCRN_RP_GRP	Screen Repeating Group
SCRN_RG_OCC	Screen Repeating Group Occurrence
SCRN_FLD_PRMT	Screen Field, Prompt
PROMPT	Prompt
UNFORMAT_INPUT	Unformatted Input Definition
UNFRMT_INP_USAGE	Unformatted Input Usage

The PI tables in this section are primarily maintained by Screen Design. For a diagram showing the associations between screen definition objects, see the appendix "Relationships Between PI Tables.".

Design Tasks Related to Screen Design Objects

The following table directs you to the PI table explanation that contains SQL for Design tasks related to Screen Design objects:

If You Want To	See
Select the attribute views that appear on a screen	SCRN_VAR_IO
Select all literal fields that appear on a screen	SCRN_FLD_LIT
Select all literal fields that appear in a template used by a screen	SCRN_FLD_LIT
Select all prompts that appear on a specific screen	SCRN_FLD_PRMT
List all screens that use templates	TMPLT_USAGE
Select all templates for a specific business system	SCREEN_TMPLT
Select the attribute that each parameter of the unformatted input inputs to for a procedure step	UNFORMAT_INPUT
Select the prompts used as keywords for each parameter for a procedure step	UNFRMT_INP_USAGE

SCREEN_DEF (Screen Definition)

Screen Definitions are created:

- Automatically when the user invokes the Screen Design with a selected procedure step.

- When the user builds templates.

Screens are always associated with one procedure step.

SCREEN_TMPLT (Screen Template)

Screen Templates are created when the user adds a new template within the Screen Design Tool. They are always associated with a single Business System.

Task

Select all templates for a particular Business System.

```
SCREEN_TMPLT.BUSINESS_SYS_ID = BUSINESS_SYS.ID
```

TMPLT_USAGE (Screen Template Usage)

Screen Template Usages are created automatically when the user uses a template definition within a screen layout.

The Screen Template Usage table forms an intersection between Screen Definitions and Template Definitions and records the templates used in the various screens.

Task

List all screens that use templates.

```
SELECT PS.NAME, TMPLT.NAME
  FROM
  MODEL,
  BUSINESS_SYS BSYS,
  BUSINESS_PROC BP,
  BUS_PROC_STEP PS,
  SCREEN_DEF SCRN,
  SCREEN_TMPLT TMPLT,
  TMPLT_USAGE
  WHERE
  MODEL.NAME = 'my model name'
  AND BSYS.NAME = 'my business system name'
  AND BSYS.MODEL_ID = MODEL.ID
  AND BP.BUSINESS_SYS_ID = BSYS.ID
  AND PS.BUS_PROCEDURE_ID = BP.ID
  AND SCRN.BUS_PROC_STEP_ID = PS.ID
  AND SCRN.ID = TMPLT_USAGE.SCREEN_DEF_ID
  AND TMPLT.ID = TMPLT_USAGE.SCREEN_TMPLT_ID
ORDER BY PS.NAME, TMPLT.NAME;
```

SCRN_FLD_LIT (Screen Field Literal)

Screen field literals are created by selecting ADD and then the Literal action. The literal text string represents the string specified by the designer and the row and column values, the coordinates where the literal was placed on the screen.

The literal text string can be updated by detailing the field. The row and column values are updated when the field is moved to another position on the screen.

Task

Select all literal fields that appear on a specific screen.

```
SCRN_FLD_LIT.SCREEN_DEF_ID = SCREEN_DEF.ID
```

Note: Literals can also be included on screens indirectly through the use of a template. For example, see the next task.

Select all literal fields that appear on a screen.

```
SELECT PS.NAME, FL.ROW, FL.COL, FL.TEXT
  FROM
  MODEL,
  BUS_PROC_STEP PS,
  SCREEN_DEF SCRN,
  SCRN_FLD_LIT FL
  WHERE
  MODEL.NAME = 'my model name'
  AND PS.MODEL_ID = MODEL.ID
  AND SCRN.BUS_PROC_STEP_ID = PS.ID
  AND FL.SCREEN_DEF_ID = SCRN.ID;
```

Task

Select all the literals from a template that the screen uses.

```
SELECT PS.NAME, FL.ROW, FL.COL, FL.TEXT
  FROM
  MODEL,
  BUS_PROC_STEP PS,
  SCREEN_DEF SCRN,
  SCREEN_TMPLT TMPLT,
  TMPLT_USAGE TU,
  SCRN_FLD_LIT FL,
  WHERE
  MODEL.NAME = 'my model name'
  AND PS.MODEL_ID = MODEL.ID
  AND SCRN.BUS_PROC_STEP_ID = PS.ID
  AND SCRN.ID = TU.SCREEN_DEF_ID
  AND TMPLT.ID = TU.SCREEN_TMPLT_ID
  AND FL.SCREEN_DEF_ID = TMPLT.ID;
```

SCRN_SYS_DEF (Screen System-Defined Variables)

Screen system-defined variables are created automatically when placing a special field on the screen.

The Screen System-defined Variables table identifies system-defined attributes and the screens or templates on which they appear.

SCRN_VAR_DEF (Screen Variable Definition)

Screen variable definitions are created automatically when a field is defined as a sub-command of the field command.

There is only one variable definition for each attribute view that appears on the screen. If an attribute is part of a repeat group, and hence appears on the screen many times, it is represented as multiple entries in the screen field variable table (SCRN_FLD_VAR).

Variable definitions are associated with attribute views indirectly using the screen variable I/O table (SCRN_VAR_IO). Each variable definition can act on behalf of one input or one output attribute view.

SCRN_VAR_IO (Screen Variable Input/Output)

Screen variable input/outputs are created automatically when the user:

- Defines a variable as appearing on the screen.
- Details the field and changes its input or output information view.

The Screen Variable Input/Output table represents the mapping of screen fields to attribute views. A field can act as any of these:

- An input field for an input attribute view
- An output field for an output attribute view
- Both

Task

List the attribute views that appear on a screen.

```
SELECT PS.NAME, EV.NAME, ET.NAME, A.NAME
  FROM
  MODEL,
  BUS_PROC_STEP PS,
  SCREEN_DEF SCRN,
  SCRN_VAR_DEF SVD,
  SCRN_VAR_IO SVIO,
  ATTR_VIEW AV,
  ATTRIBUTE A,
  ENTITY_VIEW EV,
  ENTITY_TYPE ET
  WHERE
  MODEL.NAME = 'my model name'
  AND PS.MODEL_ID = MODEL.ID
  AND SCRN.BUS_PROC_STEP_ID = PS.ID
  AND SVD.SCREEN_DEF_ID = SCRN.ID
  AND SVIO.SCRN_VAR_DEF_ID = SVD.ID
  AND SVIO.ATTR_VIEW_ID = AV.ID
  AND AV.ATTRIBUTE_ID = A.ID
  AND AV.ENTITY_VIEW_ID = EV.ID
  AND EV.ENTITY_ID = ET.ID
ORDER BY PS.NAME, ET.NAME, EV.NAME, A.NAME;
```

SCRN_FLD_VAR (Screen Field Variable)

Screen field variables are created automatically by positioning attributes on a screen. The variable is associated with a variable definition that was previously created by defining a field before positioning it.

The row and column values are updated when the field is moved to another position on the screen.

SCRN_FLD_VARP (Screen Variable Properties)

The Screen Variable Properties table contains the video properties for screen variables. It can be viewed simply as an extension of the Screen Field Variable (SCRN_FLD_VAR) table. The columns of this table are updated by specifying that a field should have customized video properties.

If default properties are required, the appropriate values are stored in the system default table (DFLTV_AR_VDAT).

SCRN_FLD_VARE (Screen Variable Error Properties)

The Screen Variable Error Props table contains the error video properties for screen variables. It can be viewed simply as an extension of the Screen Field Variable (SCRN_FLD_VAR) table. The columns of this table are updated by specifying that a field should have customized error video properties.

If default properties are required, the appropriate values are stored in the system default table (DFLT_VARE_VDAT).

CSTM_EDT_PTRN (Custom Edit Pattern)

The Custom Edit Pattern table contains the text of custom edit patterns. It is created when a custom edit pattern is typed as a field is added or the properties of a field are detailed.

SCRN_RP_GRP (Repeating Group Definition)

A repeating group definition is created automatically when the user positions a repeating group information view on the screen.

Each repeating group placed on the screen has only one repeating group definition. This is true, even though the highest-level repeating group contains other repeating groups in the view definition.

SCRN_RP_OCC (Repeating Group Occurrence)

The Repeating Group Occurrence table represents an occurrence of a repeating group on the screen.

SCRN_FLD_PRMT (Screen Field Prompt)

Screen field prompts are created by selecting a prompt to be associated with a variable field. They represent the usage of a prompt on a particular screen or template.

The PROMPT_ID represents the prompt selected by the user. The row and column values are the coordinates where the prompt was placed on the screen.

The prompt string can be changed by detailing the field. The row and column values are updated when the prompt is moved to another position on the screen.

Task

Select all prompts used in a specific screen.

```
SCRN_FLD_PRMT.SCREEN_DEF_ID = SCREEN_DEF.ID
```

PROMPT (Prompt Definition)

Prompts are associated with particular attributes but can be used in many screens or templates. Each prompt is directly associated with the attribute for which it is a prompt and with many Screen Field Prompts (SCRN_FLD_PRMT) that represent the usage of a prompt on a particular Screen or Template.

UNFORMAT_INPUT (Unformatted Input)

The UNFORMAT_INPUT table represents each occurrence of a parameter in the list of parameters for unformatted input for a procedure step. Each parameter has input to one attribute view or system attribute and is always used by one procedure step.

Task

Select the attribute that each parameter of the unformatted input inputs to a procedure step.

```
UNFORMAT_INPUT.ATTR_VIEW_ID = ATTR_VIEW.ID
AND UNFORMAT_INPUT.BUS_PROC_STEP_ID = BUS_PROC_STEP.ID
AND ATTR_VIEW.ATTRIBUTE_ID = ATTRIBUTE.ID
```

UNFRMT_INP_USAGE (Unformatted Input Usage)

The UNFRMT_INP_USAGE table represents the usage of prompts by unformatted input. Each parameter may use many prompts as keywords to identify the various parameters in the parameter list.

Task

Select the prompts used as keywords for each parameter for a procedure step.

```
UNFRMT_INP_USAGE.PROMPT_ID = PROMPT.ID
AND UNFRMT_INP_USAGE.UNFORMAT_INP_ID = UNFORMAT_INPUT.ID
```

BSD_ACTN_BLK (Design Action Block)

Design Action Blocks are used in the same way as Analysis Action Blocks in the Process Action Diagram.

Procedure Action Diagram Objects

Procedure Action Diagram objects are added during Business System Design. The ACTION_BLOCK table contains Analysis Action Blocks and Design Action Blocks.

The PAD_CREATE, PAD_READ, PAD_UPDATE, PAD_DELETE, and PAD_SET_ATTR tables discussed under Process Action Diagrams can also be used with Procedure Action Diagrams.

The following table defines various action diagram objects.

Object	Definition
ACTN_BLK_USE	Action Block Usage
BSD_ACTN_BLK	Design Action Block

Business System Defaults

The tables in this section represent the default options that are associated with a Business System.

The following table defines various default objects.

Object	Definition
COMMAND	Command
CMD_SYNONYM	Command Synonym
EXIT_STATE	Exit State
MESSAGE	Message for an Exit State
DFLT_LIT_VDAT	Default Literal Video Attributes
DFLT_PRM_VDAT	Default Prompt Video Attributes
DFLT_VAR_VDAT	Default Variable Field Video Attributes
DFLT_VARE_VDAT	Default Variable Error Field Video Attributes
DFLT_EDT_PTRN	Default Edit Pattern
SYSTEM_PF_KEY	System Program Function (PF) Key Definition
PARM_DELIMITER	Parameter Delimiter
PARM_STRING_DEL	Parameter String Delimiter
DIALECT	Dialect

Object	Definition
DIALECT_TEXT	Dialect Text
SCROLL_AMOUNT	Scroll amount values for a Dialect

Design Tasks Related to System Default Objects

The following table directs you to the PI table explanation that contains SQL for Design tasks related to system default objects:

If You Want To	See
List all commands defined for a business system	COMMAND
List all command synonyms for a specific business system	CMD_SYNONYM
List all the dialect text values for commands	DIALECT_TEXT
List all exit states defined for a business system	EXIT_STATE
Select the message for an exit state	MESSAGE
List all parameter delimiters for a business system	PARM_DELIMITER
List all parameter delimiters for a procedure step	PARM_DELIMITER
List all string separators for a business system	PARM_STRING_DEL
List all string separators for a procedure step	PARM_STRING_DEL

COMMAND

Each Business System processes many commands. Commands can be input through the use of a command variable on a screen definition or set implicitly by a Dialog Flow.

Task

List all the commands defined for a specific Business System.

```
SELECT CMD.NAME
  FROM
  MODEL,
  BUSINESS_SYS BSYS,
  COMMAND CMD
  WHERE
  MODEL.NAME = 'my model name'
  AND BSYS.MODEL_ID = MODEL.ID
  AND BSYS.NAME = 'my business system name'
  AND CMD.BUSINESS_SYS_ID = BSYS.ID
ORDER BY CMD.NAME;
```

CMD_SYNONYM (Command Synonym)

Each command defined in a Business System can have many synonyms. Synonyms can be used as input to a screen command field.

Task

List all the command synonyms defined for a specific Business System.

```
SELECT CMDSYN.CMD_SYNONYM
  FROM
  MODEL,
  BUSINESS_SYS BSYS,
  COMMAND CMD,
  CMD_SYNONYM CMDSYN
  WHERE
  MODEL.NAME = 'my model name'
  AND BSYS.MODEL_ID = MODEL.ID
  AND BSYS.NAME = 'my business system name'
  AND CMD.BUSINESS_SYS_ID = BSYS.ID
  AND CMDSYN.COMMAND_ID = CMD.ID
ORDER BY CMDSYN.CMD_SYNONYM;
```

EXIT_STATE

Each Business System can have many defined exit states.

Task

List all the exit states defined for a specific Business System.

```
SELECT EXS.NAME
  FROM
  MODEL,
  BUSINESS_SYS BSYS,
  EXIT_STATE EXS
  WHERE
  MODEL.NAME = 'my model name'
  AND BSYS.MODEL_ID = MODEL.ID
  AND BSYS.NAME = 'my business system name'
  AND EXS.BUSINESS_SYS_ID = BSYS.ID
ORDER BY EXS.NAME
```

Joins to Retrieve Exit State Messages

The following table describes tasks to be performed for retrieving exit state messages.

To Retrieve Exit State Messages For	Join
Default dialect	EXIT_STATE with TEXT
Non-default dialect	EXIT_STATE with MESSAGE and DIALECT_TEXT

MESSAGE (Exit State Message)

Each Exit State for non-default dialects can have a default message. The message type can be informational, warning, error or normal.

Task

Select the message for an Exit State.

```
MESSAGE.ID = EXIT_STATE.MESSAGE_ID
```

DFLT_LIT_VDAT (Default Literal Video Attribute)

The Default Literal Video Attribute table represents the default video attributes used to display literal values on screens unless they are explicitly overridden by customized values.

Each Business System has one set of default video attributes for literals.

DFLT_PRM_VDAT (Default Prompt Video Attribute)

The Default Prompt Video Attribute table represents the default video attribute used to display prompt values on screens unless they are explicitly overridden by customized values. Each Business System has one set of default video attributes for prompts.

DFLT_VAR_VDAT (Default Variable Video Attribute)

The Default Variable Video Attribute table represents the default video attributes used to display variable fields on screens unless they are explicitly overridden by customized values. Each Business System has one set of default video attributes for variables.

DFLT_VARE_VDAT (Default Error Video Attribute)

The Default Error Video Attribute table represents the default video attributes used to display error fields on screens unless they are explicitly overridden by customized values.

Each Business System has one set of default video attributes for error fields.

DFLT_EDT_PTRN (Default Edit Pattern)

The Default Edit Pattern table represents the default edit patterns that have been defined. Each edit pattern is associated with one Business System.

SYSTEM_PF_KEY (System PF Key Definition)

The SYSTEM_PF_KEY table represents the system-wide PF key definitions that are to be used within a Business System. The entries in this table are created automatically when a Business System is added to the model, but the column values are maintained using the System Defaults option on the Design tools menu.

PARM_DELIMITER (Parameter Delimiters for Unformatted Input)

The PARM_DELIMITER table contains the parameter delimiters for unformatted input. The parameter delimiters are defined by selecting SYSTEM DEFAULTS within the Business System Definition function.

Task

Select all the parameter delimiters used by a procedure step.

```
PARM_DELIMITER.BUS_PROC_STEP_ID = BUS_PROC_STEP.ID
```

Task

Select all parameter delimiters within a business system.

```
PARM_DELIMITER.BUSINESS_SYS_ID = BUSINESS_SYS.ID
```

PARM_STRING_DEL (Parameter String Delimiters for Unformatted Input)

The PARM_STRING_DEL table contains the string separators for unformatted input. The string delimiters are defined by selecting SYSTEM DEFAULTS within the Business System Definition function.

Task

Select all string separators used by a procedure step.

```
PARM_STRING_DEL.BUS_PROC_STEP_ID = BUS_PROC_STEP.ID
```

Task

Select all string separators within a business system.

```
PARM_STRING_DEL.BUSINESS_SYS_ID = BUSINESS_SYS.ID
```

DIALECT

Bilingual applications support provides the capability of designing an application system with end-user presentation in more than one language. Bilingual support is implemented using dialects. A dialect would exist for each language. Every model has an implied default dialect. The user may define multiple dialects for bilingual support.

DIALECT_TEXT

For bilingual support, each dialect (except the default dialect) may have text values for the following:

- Prompts
- Literals
- Commands
- Command synonyms
- Exit states
- Messages
- Edit patterns
- Scroll amounts

Task

List all the dialect text values for commands.

```
SELECT DLECT.NAME, CMD.NAME, TEXT.TEXT_VALUE
  FROM
  MODEL M,
  DIALECT DLECT,
  DIALECT_TEXT TEXT,
  COMMAND CMD
  WHERE
  M.NAME = 'my model name'
  AND DLECT.MODEL_ID = M.ID
  AND TEXT.DIALECT_ID = DLECT.ID
  AND TEXT.OBJECT_ID = CMD_ID
ORDER BY DLECT.NAME, CMD.NAME;
```

SCROLL_AMOUNT (Scroll Amount Values for Dialect)

Each dialect has its own set of scroll amount values. The following are the illustrations:

- Page
- Half
- Csr
- Max

Internal Design Objects

For the purposes of Export, Internal Design objects are those created in the Data Store List. The Data Store List is used to implement a conceptual data model (ERD) in a physical data environment (DBMS). For example, entity types are implemented as records; attributes are implemented as fields; relationships are implemented as linkages; identifiers are implemented as entry points, and so forth.

Physical data objects such as databases, data stores, indexes, and data sets are defined or identified for use, and the Data Definition Language (DDL) statements to define them are created.

Data Store List objects are entered into the Host Encyclopedia using the Data Store List tool and the ERD–DSD transformation. The ERD represent the conceptual schema of the ANSI/SPARC three-schema architecture. The DSD captures the external schema and internal schema, which together implement the conceptual schema. For illustrations representing the relationships of internal and external schema objects, see the appendix "Relationships Between PI Tables."

Data Structure Diagram Objects

Data Structure Diagram objects include:

- Internal schema definition objects
- External schema definition objects
- Construction DB2 objects
- Construction objects for Cascade Delete

Internal Schema Definition Objects

The following table defines various definition objects.

Object	Definition
TECHNICAL_DESIGN	Technical Design
STORAGE_GROUP	Storage Group
DATA_BASE	Database
DATA_BASE_USAGE	Database Usage
DATA_STORE_TBLSP	Tablespace Data Store
DATA_STORE_INDEX	Index Data Store
DATASET_TBLSP	Tablespace Data Set
DATASET_INDEX	Index Data Set
DASD_VOLUME	DASD Volume
DASD_VOL_USAGE	DASD Volume Usage
STG_DVOL_USAGE	Storage Group/DASD volume usage
TECHNICAL_SYSTEM	Technical System (see the following note)
LIBRARY	Library (see the following note)
LIBRARY_USAGE	Library Usage (see the following note)
LIB_USAGE_SCOPE	Library Usage Scope (see the following note)
DATACOM_COLUMN	DATACOM Column
DATACOM _CONSTRNT	DATACOM Constraint
DATACOM _DATABASE	DATACOM Database
DATACOM _INDEX	DATACOM Index
DATACOM _TABLE	DATACOM Table

Object	Definition
DATACOM _TD	Datacom Technical Design
DB2_MVS_COLUMN	DB2 Column
DB2_MVS_CONSTRNT	DB2 Constraint
DB2_MVS_DATABASE	DB2 Database
DB2_MVS_INDEX	DB2 Index
DB2_MVS_INDEXSPC	DB2 Indexspace
DB2_MVS_TABLE	DB2 Table
DB2_MVS_TABLESPC	DB2 Tablespace
DB2_MVS_TD	DB2 Technical Design

Note: These are objects created during Construction Stage.

External Schema Definition Objects

External Schema Definition Objects for Internal Design are:

- Data implementation objects
- Access and connection objects

Data Implementation Objects

The following table defines various implementation objects.

Object	Definition
RECORD	Record Definition
FIELD	Field Definition
ENTITY_REC_IMPL	Entity to Record Implementation

Access and Connection Objects

The following table defines various access and connection objects.

Object	Definition
ENTRY_POINT	Entry Point Definition
REC_ENTRY_PT_USE	Record Entry Point Usage

Object	Definition
FLD_ENTRY_PT_USE	Field Entry Point Usage
FLD_ENTPT_VALUE	Field Entry Point Value
REL_PART_IMPL	Relationship or Partition Implementation
LINKAGE	Linkage Definition
FLD_LINK_USE	Foreign Key Field Linkage Usage

Construction DB2 Objects

Internal Design objects include Construction DB2 objects, which are defined in the following table:

Object	Definition
DB2_DDL_DB	Member name containing generated DDL for database definition
DB2_DDL_INDEX	Member name containing generated DDL for index definition
DB2_DDL_TABLE	Member name containing generated DDL for table definition
DB2_DDL_TBLSP	Member name containing generated DDL for tablespace definition
DB2_DEF_CLUSTER	Member name containing VSAM define cluster for data store

Construction Objects for Cascade Delete

The following table defines various cascade delete objects.

Object	Definition
CD_ACTN_BLK	CD Action Block for Cascade Delete
IMPLEMENT_LOGIC	Implementation Logic
XT_IMPL_LOGIC	External Implementation Logic
DB2_RESRC_MODULE	DB2 Resource Module
RECORD_REFERENCE	Record Reference by an Implementation Logic
IMPL_LOGIC_USAGE	Implementation Logic Usage

Object	Definition
TD_LIBRARY_USAGE	Technical Design/Library Usage

Internal Schema Objects

Internal schema, or Technical Design objects, are created by using the database design part of the Design toolset.

Design Tasks Related to Internal Schema Definition Objects

The following table directs you to the PI table explanation that contains SQL for Design tasks related to database design objects:

If You Want To	See
Join Technical Design to the model that contains it	TECHNICAL_DESIGN
Select the DASD volumes defined for a storage group	STG_DVOL_USAGE
Select the DASD volume on which a data set resides	DASD_VOLUME
Select the DASD volumes on which a specific tablespace resides	DASD_VOL_USAGE
Select the data set that an indexspace resides on	DATASET_INDEX
Select the data sets that a tablespace resides on	DATASET_TBLSP
Select the default storage group for a database	DATA_BASE
Select the default storage group for databases within a Technical Design	STORAGE_GROUP
Select the default storage group for indexspaces	STORAGE_GROUP
Select the default storage group for tablespaces	STORAGE_GROUP
Select the indexspace datasets that are held by a storage group	DATASET_INDEX
Select the indices contained in a storage group	DATA_STORE_INDEX
Select the indices grouped within a database	DATA_STORE_INDEX
Select the tablespaces contained in a storage group	DATA_STORE_TBLSP
Select the tablespace datasets that are held by a storage group	DATASET_TBLSP
Select the tablespaces grouped within a database	DATA_STORE_TBLSP
Select all uses of databases within a Technical Design	DATA_BASE_USAGE

TECHNICAL_DESIGN

Technical Design objects are created by using the Data Structure Diagram in the Design toolset. The Technical Design Table is referenced by all technical design objects.

Task

Join the Technical Design to the model that contains it.

```
TECHNICAL_DESIGN.MODEL_ID = MODEL.ID
```

STORAGE_GROUP

The Storage Group table represents the DB2 storage groups. They are used to define a set of volumes on which storage may be allocated for tablespaces and indexes.

Task

Select the default storage group for databases within a technical design.

```
STORAGE_GROUP.ID = TECHNICAL_DESIGN.DEF_STG_DB_ID
```

Task

Select the default storage group for tablespaces.

```
STORAGE_GROUP.ID = TECHNICAL_DESIGN.DEF_STG_TBLSP_ID
```

DATA_BASE

The Data Base table represents the DB2 or Datacom databases. It records the name (NAME) of the database and the default bufferpool (BUFFERPOOL) to be used by tablespaces in the database. (BUFFERPOOL is not used with the Datacom database.)

This table is maintained by the DSD Group command. Databases may be created or referenced by the DSD. They represent a grouping of data stores.

DATA_BASE_USAGE

The Data Base Usage table represents the connection between a database and a technical design. It records only the connection.

Task

Select all the uses of databases within a technical design.

```
DATA_BASE_USAGE.TECH_DESIGN_ID = TECHNICAL_DESIGN.ID
AND DATA_BASE_USAGE.DATABASE_ID = DATABASE.ID
```

DATA_STORE_TBLSP (Tablespace Data Store)

The Tablespace Data Store table represents DB2 tablespaces. Each tablespace must have at least one data set (DATASET_TBLSP) but may have more than one. Most tablespaces will have one or more tables stored in them.

The Tablespace Data Store table records the name (NAME), bufferpool (DB2 BUFFERPOOL), VSAM catalog name (VSAM_CAT NAME), unit of locking (DB2_LOCKSIZE), and the close-data-sets-when-not-in-use option (DB2_CLOSE).

This table is maintained by the DSD Group command.

Task

Select the tablespaces grouped within a database.

```
DATA_STORE_TBLSP.DATA_BASE_ID = DATA_BASE.ID
```

Select the tablespaces contained in a storage group.

```
DATA_STORE_TBLSP.STORAGE_GROUP_ID = STORAGE_GROUP.ID
```

DATA_STORE_INDEX (Index Data Store)

The Index Data Store table represents DB2 indexspaces. Each indexspace contains exactly one index (ENTRY_POINT), and each index resides in exactly one indexspace. An indexspace must have at least one data set (DATASET_INDEX) and may have more than one.

This table records the name (NAME), bufferpool (DB2_BUFFERPOOL), VSAM catalog name (VSAM_CAT NAME), number of subpages per index page (DB2_SUBPAGES), and the close-data-sets-when-not-in-use option (DB2_CLOSE).

This table is maintained by the DSD Detail Entry Point Data Store option.

Task

Select the indices grouped within a database.

```
DATA_STORE_INDEX.DATA_BASE_ID = DATA_BASE.ID
```

Task

Select all the indices contained in a storage group.

```
DATA_STORE_INDEX.STORAGE_GRP_ID = STORAGE_GROUP.ID
```

DATASET_TBLSP (Tablespace Data Set)

The Tablespace Data Set table represents a physical disc allocation used by a tablespace (DATA_STORE_TBLSP).

It records:

- Free page frequency (FREE_PAGE_FREQ)
- Free space percentage (FREE_SPACE_PCT)
- Primary allocation quantity (PRIME_ ALLOC)
- Secondary allocation quantity (SECOND_ALLOC)
- Allocation unit (UNIT_ALLOC)
- Partitioning data set number (PART_NUMBER)
- Sequence (SEQ) within the tablespace (DATA_STORE_TBLSP)

This table is maintained by the DSD Group command.

Task

Select the data sets that a tablespace resides on.

```
DATASET_TBLSP.DATA_STORE_ID = DATA_STORE TBLSP.ID
```

Task

Select all the tablespace data sets that are held by a storage group.

```
DATASET_TBLSP.STORAGE_GROUP_ID = STORAGE_GROUP.ID
```

DATASET_INDEX (Index Data Set)

The Indexspace Data Set table represents a physical disk allocation used by an indexspace (DATA_STORE_INDEX).

It records:

- Free page frequency (FREE_PAGE_FREQ)
- Free space percentage (FREE_SPACE_PCT)
- Primary allocation quantity (PRIME ALLOC)
- Secondary allocation quantity (SECOND ALLOC)
- Allocation unit (UNIT_ALLOC)

- Partitioning data set number (PART_NUMBER)
- Sequence (SEQ) within the indexspace (DATA_STORE_INDEX)

The Indexspace Data Set table is maintained by the DSD Detail Entry Point Data Store option.

Task

Select the data sets that an indexspace resides on.

```
DATASET_INDEX.DATA_STORE_ID = DATA_STORE_INDEX.ID
```

Task

Select all the indexspace data sets that are held by a storage group.

```
DATASET_INDEX.STORAGE_GRP_ID = STORAGE_GROUP.ID
```

DASD_VOLUME

The DASD Volume table represents DASD volumes that contain data sets. It records the volume's identifying volume serial number (Volser) and type (Type).

A data set resides on a volume. Each volume may contain many data sets. A data set may reside on many volumes.

Task

Select the DASD volume on which a data set resides.

```
DASD_VOL_USAGE.DATASET_ID = DATASET_ZZZZ.ID
AND DASD_VOL_USAGE.DASD_VOLUME_ID = DASD_VOLUME.ID
```

when ZZZZ = INDEX or TBLSP.

DASD_VOL_USAGE (DASD Volume Usage)

The DASD Volume Usage table represents the usage of a particular DASD VOLUME to store a particular data set (DATASET INDEX, DATASET_TBLSP). It records the usage only and is maintained by the Data Structure Diagram Data Set Properties Panel.

There is a DASD volume usage for each volume on which a data set resides.

Task

Select the DASD volumes on which tablespace ABCD resides.

```
DATA_STORE_TBLSP.NAME = 'ABCD'
AND DATASET_TBLSP.DATA_STORE_ID = DATA_STORE_TBLSP.ID
AND DASD_VOL_USAGE.DATASET_ID = DATASET_TBLSP.ID
AND DASD_VOL_USAGE.DASD_VOLUME_ID = DASD_VOLUME.ID
```

STG_DVOL_USAGE (Storage Group/ DASD Volume Usage)

The Storage Group/DASD Volume Usage table represents the usage of a particular DASD volume to a storage group.

Task

Select the DASD volumes defined for a storage group.

```
STG_DVOL_USAGE.DASD_VOLUME_ID = DASD_VOLUME.ID AND
STG_DVOL_USAGE.STORAGE_GRP_ID = STORAGE_GROUP.ID
```

External Schema Definition Objects

External schema definition objects are created using the database design component of the Design toolset.

Design Tasks for Data Implementation Objects

The following table directs you to the PI table explanation that contains SQL for Design tasks related to data implementation objects:

If You Want To	See
Find all the fields in a record	RECORD
Find the attribute that a field implements	FIELD
Select the records that implement an entity type	ENTITY_REC_IMPL

RECORD (Record Definition)

All records are database tables. A record is either a data record or a link record depending on the value in the ROLE field.

A data record represents an implementation of an entity type or subtype. An entity type or subtype can be implemented by more than one record. Conversely, a record may implement more than one entity type or subtype. The Entity to Record Implementation defines the entity type to record intersection. For more information, see ENTITY_REC_IMPL.

A link record implements a many-to-many relationship between two entity types. A many-to-many relationship between Entity Type A and Entity Type B is equivalent to a one-to-many relationship between A and LINK and a one-to-many relationship between B and LINK, where LINK is an intersection entity type.

A link record is an implementation of such an intersection entity type. Link records have no data fields, only foreign key fields.

Task

Find all fields in a record.

```
RECORD.NAME = 'the_name'
AND FIELD.RECORD_IS_IN_ID = RECORD.ID
ORDER BY FIELD.SEQ
```

FIELD (Field Definition)

The Field Definition table represents the fields or columns in database tables. It includes all fields of the three possible types:

- Data fields
- Denormalized fields
- Foreign key fields

In the table, the three types are distinguished by the ROLE field.

Fields are implementations of attributes.

Task

Determine the attribute that the field implements.

```
FIELD.ATTRIBUTE_ID = ATTRIBUTE.ID
```

Determine the record that contains this field.

```
FIELD.RECORD_IS_IN_ID = RECORD.ID
```

Data fields are primary implementations of attributes defined in the ERD. Denormalized fields are redundant implementations of an attribute. (For performance reasons, it may be advantageous to implement an attribute in more than one table in more than one record.) The denormalized field is associated with a relationship to the entity type from which it comes.

Identify the relationship associated with a denormalized field.

```
FIELD.REL_DENORM_ID = RELATIONSHIP.ID
```

Foreign key fields are part of a linkage. They are introduced into records as part of the relationship implementation process. For example, assume a relationship between Entity Type A and Entity Type B. When A is implemented, the identifier of B is made a foreign key in A. Foreign key fields are value-based pointers, copies of the target fields.

FIELD records the following:

- Name (Name)
- Macro name (Macro_Name)
- Sequence (Seq) within the table (RECORD)
- Role:
  - **P**—Data
  - **F**—Foreign Key
  - **D**—Denormalized

  Role is always P for primary implementation data fields.
- Format (Format)
- Number of occurrences (Occurs) (not allowed for DB2 or Datacom)
- Length in digits or characters (Length)
- Number of decimal places (Dec_Places)
- Name of fieldproc (Fieldproc_Name)
- Optionality (Opt)

The table (RECORD_IS_IN_ID) containing the field and the attribute (Attribute_ID) implemented by the field are also recorded.

This table is maintained by the DSDs Detail Record Layout Detail Field options.

ENTITY_REC_IMPL (Entity to Record Implementation)

The Entity to Record Implementation table represents the entity to record intersection. An entity type may be implemented by one or more records. A record may implement one or more entity types.

Task

Determine the records that implement an entity type.

```
ENTITY_TYPE.NAME= 'the_name'
AND ENTITY_REC_IMPL.ENTITY_ID = ENTITY_TYPE.ID
AND ENTITY_REC_IMPL.RECORD_ID = RECORD.ID
```

Access and Connection Objects

Access and connection objects are entry points, linkages, field link usages, and related objects.

Design Tasks Related to Access and Connection Objects

The following table directs you to the PI table explanation that contains SQL for Design tasks related to database access and connection objects:

If You Want To	See
Select the entry points defined for record	REC_ENTRY_PT_USE
Select the entry points used for each entry point implementation technique, the record on which the entry point is defined, and the foreign key fields and their usage in the entry point	Entry Point Technique
Select the fields that participate in an index	FLD_ENTRY_PT_USE
Select all the limit values for a partitioned entry point.	FLD_ENTPT_VALUE
Select all the limit values for a partitioned indexspace dataset.	FLD_ENTPT_VALUE
Select the foreign keys in the from record and the corresponding target field of the to record	Foreign Key Technique
Select the foreign key fields in the link record and the corresponding target field of the to record	Many-to-Many Technique
Select the field that a foreign key is pointing to	FLD_LINK_USE

ENTRY_POINT

The Entry Point table represents the indexes defined using the DSD.

It records the index name (Name), whether it is clustered or not (Clustered), whether it is partitioned or not (Partitioned), the indexspace (Data_Str_Ndx_ID) that contains the index, the identifier (Identifier_ID) implemented by the index, if any, and the relationship or partitioning implementation (Rel_Impl_ID) that uses the index, if any.

The Entry Point table is maintained by the DSD Detail Entry Point Name and Properties option.

A record entry point may be created as a result of implementing an identifier, relationship, or partitioning, or it may be added for performance reasons. If the entry point is added for performance reasons, both the identifier (Identifier_ID) and relationship or partitioning implemented (Rel_Impl_ID) columns are zero.

If the entry point represents an identifier, the Identifier_ID is present. If the entry point is used in the implementation of a relationship or partitioning, the Rel_Impl_ID is present.

REC_ENTRY_PT_US (Record Entry Point Usage)

The Record Entry Point Usage table represents the definition of an index (Entrypoint_ID) on a record (Record_ID). It identifies whether the index is unique or non-unique (Unique) and the role the record plays in defining the index (Role).

The Record Entry Point Usage table is maintained by the DSD Add Entry Point option and Detail Entry Point Properties option.

Task

Select the entry points defined for a record.

```
REC_ENTRY_PT_USE.RECORD_ID = RECORD.ID
AND REC_ENTRY_PT_USE.ENTRY_POINT_ID = ENTRY_POINT.ID
```

FLD_ENTRY_PT_US (Field Entry Point Usage)

The Field Entry Point Usage table represents the usage of a field (Data, Denormalized, or Foreign Key) by an index (ENTRY_POINT).

It records the order (Sequence), ascending or descending, of values of fields that appear in the index and the order (Seq) where this field appears relative to other fields in the same index.

The Field Entry Point Usage table is maintained by the DSD Detail Entry Point Contents option.

Task

Select the fields that participate in an index.

```
FLD_ENTRY_PT_USE.FIELD_ID = FIELD.ID
AND FLD_ENTRY_PT_USE.ENTRY_POINT_ID = ENTRY_POINT.ID
ORDER BY FLD_ENTRY_PT_USE.ENTRY_POINT_ID,
FLD_ENTRY_PT_USE.SEQ
```

REL_PART_IMPL (Relationship or Partition Implementation)

When either a relationship or a partitioning is implemented, a REL_PART_IMPL is created. Rel_Part_ID identifies the relationship or partitioning that is implemented.

A relationship or partitioning implementation uses one of three techniques:

- Foreign Key (FK)
- Entry Point (EP)
- Many-to-Many (MN)

The joins for each technique follow.

Foreign Key (FK) Technique.

Task

Select a list of foreign key fields in the from record and the corresponding target field (part of the primary key) of the to record.

```
SELECT IMP.ID, LINK.ID, LINK.RECORD_FROM_ID, FLK.FIELD_FROM_ID,
  LINK.RECORD_TO_ID, FLK.FIELD_TO_ID
  FROM
  REL_PART_IMPL IMP,
  LINKAGE LINK,
  FLD_LINK_USE FLK
  WHERE
  IMP.TECHNIQUE = 'F'
  AND LINK.IMPLEMENTATON_ID = IMP.ID
  AND FLK.LINKAGE_ID = LINK.ID;
```

The entry point technique implements the inverse of a foreign key technique implementation. The foreign key fields used in the foreign key technique implementation become the high order part of some entry point. This allows for efficient joins following the relationship in either direction.

Task

Retrieve the entry points used for each entry point implementation technique, the record on which the entry point is defined, and the foreign key fields and their usage in the entry point.

```
SELECT IMP.ID, EP.NAME, REC.NAME, FK.NAME, EPF.SEQUENCE,
  EPF.SEQ
  FROM
  REL_PART_IMPL IMP,
  ENTRY POINT EP,
  RECORD REC,
  FIELD FK,
  FLD_ENTRY PT_USE EPF
  WHERE
  IMP.TECHNIQUE = 'E'
  AND EP.REL_IMPL_ID = IMP.ID
  AND EPF.ENTRY_POINT_ID = EP.ID
  AND FK.ID = EPF.FIELD_ID
  AND REC.ID = FK.RECORD_IS_IN_ID
ORDER BY IMP.ID, EP.NAME, EPF.SEQ;
```

Only many-to-many relationship memberships are implemented using the MN technique.

Task

Retrieve a list of foreign key fields in the link record (Record From ID) and the corresponding target field (part of the primary key) of the to record.

```
SELECT IMP.ID, LINK.ID, LINK.RECORD_FROM_ID, FLK.FIELD_FROM_ID,
LINK.RECORD_TO_ID, FLK.FIELD_TO_ID
FROM
REL_PART_IMPL IMP,
LINKAGE LINK,
FLD_LINK_USE FLK
WHERE
IMP.TECHNIQUE = ‘M’
AND LINK IMPLEMENTATON_ID = IMP.ID
AND FLK.LINKAGE_ID = LINK.ID;
```

To get the full picture, this join should be used for the implementation of each of the inverse relationship memberships in the many-to-many relationship.

FLD_ENTPT_VALUE (Field Entry Point Value)

The Field Entry Point Value table represents the limit values for a partition. Each field in a partitioned entry point may have a limit value for each partition.

Task

Select all the limit values for a partitioned entry point.

```
SELECT EP.NAME, FIELD.NAME, EPF.SEQ, VAL.VALUE
  FROM
  MODEL M,
  ENTRY_POINT EP,
  FLD_ENTRY_PT_USE EPF,
  FIELD FLD,
  FLD_ENTPT_VALUE VAL
  WHERE
  M.NAME = 'my model name'
  AND EP.ID = M.ID
  AND EP.PARTITIONED = 'Y'
  AND EPF.ENTRY_POINT_ID = EP.ID
  AND EPF.FIELD_ID = FLD.ID
  AND VAL.FLD_ENTPT_US_ID = EPF.ID
ORDER BY EP.NAME, EPF.SEQ;
```

Task

Select all the partitioning limit values for a partitioned indexspace data set.

```
FLD_ENTPT_VALUE.DATASET_INDX_ID = DATASET_INDEX.ID
```

LINKAGE (Linkage Between Records)

The Linkage table represents all linkages and directed connections between two tables (RECORDs), not necessarily distinct.

It records the type of the linkage (Pointer_Type = F for foreign key), the record that the linkage comes from (Record_From_ID), the record that the linkage points to (Record_To_ID), and the identifier (Identifier_ID) of the to record being used to choose the foreign key fields. The referential constraint option (REF_CONSTRT_OPT) represents the referential integrity option for the relationship that the linkage implements. The reason for the linkage is indicated using the Implementation_ID (a relationship or partitioning implementation). A linkage is used in the implementation of a relationship or partitioning.

FLD_LINK_USE (Foreign Key Field Linkage Usage)

The Foreign Key Field Linkage Usage (FLD_LINK_USE) table represents the usage of a foreign key field indicated by the Field From ID, by a linkage (LINKAGE) to join to a field (data or foreign key) in the linked-to record.

It records only the correspondence between the from and to fields and their use by a linkage.

The FLD_LINK_USE table is maintained by the DSD Detail Linkage Identifier option.

Task

Select the field that a foreign key field is pointing to.

```
FLD_LINK_USE.FIELD_FROM_ID = FIELD.ID (Pointing)
AND FLD_LINK_USE.FIELD_TO_ID = FIELD.ID (Target)
```

Construction DB2 Objects

Construction DB2 objects identify objects that contain Data Definition Language (DDL) used to define database objects.

DSD Tasks Related to DB2 Objects

The following table directs you to the PI table explanation that contains SQL for Construction tasks related to databases and their physical storage:

If You Want To	See
Find the database associated with a data set member	DB2_DDL_DB
Find the entry point associated with a data set member	DB2_DDL_INDEX
Find the tablespace associated with a data set member	DB2_DDL_TABLE
Find the data set associated with a data set member that contains the VSAM define cluster text	DB2_DEF_CLUSTER

DB2_DDL_DB

The DB2_DDL_DB table contains the name of the partitioned data set member that contains the DDL that defines a database. The database defined is specified by Data_Base_ID.

Task

Select the database defined by this DDL.

```
DATA_BASE.ID = DB2_DDL_DB.DATA_BASE_ID
```

DB2_DDL_INDEX

The DB2_DDL_INDEX table contains the name of the partitioned data set member that contains the DDL that defines an entry point. The entry point defined is specified by Entry_Point_ID.

Task

Select the entry point defined.

```
ENTRY_POINT.ID = DB2_DDL_INDEX.ENTRY_POINT_ID
```

DB2_DDL_TABLE

The DB2_DDL_TABLE contains the name of the partitioned data set member that contains the DDL that defines a table associated with a record. The record defined is specified by Record_ID.

Task

Select the record defined.

```
RECORD.ID = DB2_DDL_TABLE.RECORD_ID
```

DB2_DDL_TBLSP

The DB2_DDL_TBLSP table contains the name of the partitioned data set member that contains the DDL that defines a tablespace data store. The tablespace data store defined is specified by Data_Str_Tblsp ID.

Task

Select the tablespace defined.

```
DATA_STORE_TBLSP.ID = DB2_DDL_TBLSP.DATA_STR_TBSP_ID
```

DB2_DEF_CLUSTER

The DB2_DEF_CLUSTER table contains the name of the partitioned data set member that contains the VSAM define cluster text. This text defines either an indexspace data set or a tablespace data set. The defined data set is specified by DATA_SET_ID.

Task

Select the data set defined.

```
DATASET_zzzz.ID = DB2_DEF_CLUSTER.DATA_SET_ID
```

where *zzzz* is INDEX or TBLSP.

Construction Stage Objects

This section discusses the construction stage objects.

Construction Objects

The Construction objects are created on the mainframe during the Construction stage.

Construction Tasks for Technical Systems

The following table directs you to the PI table explanation that contains SQL for Construction tasks related to technical systems, libraries, and library usage:

If You Want To	See Explanation of This Table
Select the Technical System for a business system	TECHNICAL_SYSTEM
Select the name of a data set for a library	LIBRARY
Select the library usages for a technical system	LIBRARY_USAGE
Select the libraries implemented by a library usage	LIB_USAGE_SCOPE

TECHNICAL_SYSTEM

The Technical System table represents the technical system for a business system.

Task

Select the technical system for a business system.

```
TECHNICAL_SYSTEM.BUSINESS_SYS_ID = BUSINESS_SYS.ID
```

LIBRARY

The Library table contains the name of the dataset for a library.

LIBRARY_USAGE

The Library Usage table represents the library usages implemented by a technical system.

Task

Select the library usages for a technical system.

```
LIBRARY_USAGE.TECHSYS_ID = TECHNICAL_SYSTEM.ID
```

LIB_USAGE_SCOPE (Library Usage Scope)

The Library Usage Scope table represents the intersection between libraries and library usages.

Task

The Library Usage Scope table represents the intersection between libraries and library usages.

Construction Tasks For Cascade Delete

The following table directs you to the PI table explanation that contains SQL for Construction tasks related to cascade delete:

If You Want To	See Explanation of This Table
Select the cascade delete routine for an entity	CD_ACTN_BLK
Select the cascade delete routine for a linkage	CD_ACTN_BLK
Select the implementation logic for an action block	IMPLEMENT_LOGIC
Select the implementation logic for a cascade delete action block	IMPLEMENT_LOGIC
Select the external implementation logic for an external action block	XT_IMPL_LOGIC
Select the DBRM for an implementation logic	DB2_RESRC_MODULE
Select the DBRM for an external implementation logic	DB2_RESRC_MODULE
Select the records referenced by an implementation logic	RECORD_REFERENCE
Select the action blocks used to cascade delete an entity and select the action blocks it calls directly	IMPL_LOGIC_USAGE
Select the library usages for a technical design	TD_LIBRARY_USAGE

CD_ACTN_BLK (Action Block for Cascade Delete)

The Cascade Delete Action Block table represents the action blocks created for cascade delete. The action block may be a cascade delete routine for entity or linkage.

Task

Select the cascade delete routine for an entity type.

CD_ACTN_BLK.ENTITY_ID = ENTITY_TYPE.ID

Task

Select the cascade delete routine for a linkage.

CD_ACTN_BLK.LINKAGE.ID = LINKAGE.ID

IMPLEMENT_LOGIC (Implementation Logic)

The Implementation Logic table represents the implementation of an action block. It contains information such as member name, source language, generation date, generation time, generated by user ID, date compiled, time compiled, and compiled by user ID.

Task

Select the implementation logic for an action block.

IMPLEMENT_LOGIC.ACTION_BLOCK_ID = ACTION_BLOCK.ID

Task

Select the implementation logic for a cascade delete action block.

IMPLEMENT_LOGIC.ACTION_BLOCK_ID = CD_ACTN_BLK.ID

XT_IMPL_LOGIC (External Implementation Logic)

The External Implementation Logic represents the implementation of an external action block. It contains information such as: member name, source language, generation date, generation time and generated by user ID.

Task

The External Implementation Logic represents the implementation of an external action block. It contains information such as: member name, source language, generation date, generation time and generated by user ID.

DB2_RESRC_ MODULE (DB2 Resource Module)

The DB2 Resource Module table represents the DBRM for an implementation logic. It contains information such as member name, generation date, generation time, and generated by user ID.

Task

Select the DBRM for an implementation logic table.

```
DB2_RESRC_MODULE.IMPL_LOGIC_ID = IMPLEMENT_LOGIC.ID
```

Task

Select the DBRM for an external implementation logic.

```
DB2_RESRC_MODULE.IMPL_LOGIC_ID = XT_IMPL_LOGIC.ID
```

RECORD_REFERENCE (Record Reference by Implementation Logic)

The Record Reference table represents the reference of a record by the implementation logic.

Task

Select the records referenced by the implementation logic.

```
RECORD_REFERENCE.IMPL_LOGIC_ID = IMPLEMENT_LOGIC.ID
AND RECORD_REFERENCE.RECORD_ID = RECORD_ID
```

IMPL_LOGIC_USAGE (Implementation Logic Usage)

The Implementation Logic table represents the usage between implementation logic tables. The table contains:

- The ID of the implementation logic that calls the usage
- The ID of the implementation logic that is called by the usage

Task

Select the action used to cascade delete an entity and select the action

blocks it calls directly.

```
SELECT ENT.NAME, ACBLK1.NAME, ACBLK2.NAME
FROM
MODEL M,
ENTITY_TYPE ENT,
CD_ACTN_BLK ACBLK1
IMPLEMENT_LOGIC IMP1
IMPL_LOGIC_USAGE USE,
IMPLEMENT_LOGIC IMP2,
CD_ACTN_BLK ACBLK2
WHERE
M.NAME = ‘my model name‘
AND ENT.MODEL_ID = M.ID
AND ACBLK1.ENTITY_ID = ENT.ID
AND IMP1.ACTION_BLOCK_ID = ACBLK1.ID
AND USE.IMPL_LOGIC_ID = IMP1.ID
AND USE.CALLED_IMPL_ID = IMP2.ID
AND IMP2.ACTION_BLOCK_ID = ACBLK2.ID
ORDER BY ENT.NAME, ACBLK1.NAME, ACBLK2.NAME;
```

This will only select the action blocks called directly from the action block but not any other action blocks called by them.

TD_LIBRARY_USAGE (Technical Design/Library Usage)

The Technical Design/Library Usage table represents the library usages implemented by a technical design.

Task

Select the library usages for a technical design.

```
TD_LIBRARY_USAGE.TECH_DESIGN_ID = TECHNICAL_DESIGN.ID
```

Model Management Objects

Each time a model is checked in from the workstation to the Host Encyclopedia, a session for model maintenance is created. The session is associated with objects that have been changed.

Change Occurrence

The Change Occurrence table (CHANGE_OCCUR) represents the intersection between sessions and objects.

Session of Model Maintenance

The session table (SESSION) contains the date and time of the change session and the user ID of the person who made the change during the session.

Note: For more information about this date/timestamp, see the *Host Encyclopedia Version Control User Guide.*

Public Interface Functions

CA Gen provides the following Public Interface functions:

- **Export Model to Public Interface Tables**—Loads a model into the Public Interface Tables or replaces the model if it has already been loaded. Before the PI tables can be used, they must be loaded from the Host Encyclopedia.
- Delete Model from Public Interface Tables—Removes a model from the Public Interface Tables.
- **Import Model into Host Encyclopedia**—Imports models from sequential files into the Host Encyclopedia. Import Model is documented in the chapter on the model import function.
- **Public Interface Reporting**—Creates a report about models in the Public Interface. This KWIC (keyword-in-context) Report shows an index of entity types, subtypes, attributes, and aliases in the models.

Export Model to PI Tables

Export Model loads a model into the Public Interface Tables or replaces a model already in the tables. Before the extract tables can be used, the models must be loaded from the Host Encyclopedia.

1. Choose Option **2** from the Host Main menu.
2. Choose **1**. Export Model to PI Tables. The Export Model to PI Tables Panel appears.
3. Enter or select a model name. Enter the name, or press F4 for a list of models from which you can choose.
4. Choose to generate online or in batch.

5. Choose whether to remove formatting characters in descriptions.

   If you choose to remove formatting characters, special formatting characters for new line, new paragraph, and tab characters are removed from descriptions.

Delete Model from PI Tables

Delete Model removes a model from the Public Interface Tables. Any copy of the model in the Host Encyclopedia is not affected.

1. Choose Option 2 from the Host Main menu.
2. Choose **2** Delete Model from PI Tables. The Delete Model from PI Tables Panel appears.
3. Enter or select a model name. Enter the name, or press F4 prompt for a list of models from which you can choose.
4. Choose to generate online or in batch.

   After the model has been deleted from the tables, the message MODEL PROCESSED appears.

KWIC Index Report

The Public Interface Functions provides the KWIC Index Report about models that have been exported to the Public Interface.

The KWIC Index Report creates a keyword-in-context index from the entity types, entity subtypes, attributes, aliases, and, optionally, descriptions of the models loaded into the Public Interface. The report is available with a brief description (the previous 20 and next 40 characters on either side of the keyword).

Steps to Generate the KWIC Index Report

1. Choose Option 2 from the Host Main menu.
2. Choose **4** Public Interface Reports. The PI Reports Panel appears.
3. Enter or select a model name. Enter the name, or press F4 prompt for a list of models from which to choose.
4. Choose option **1** for KWIC Index.
5. Specify whether you want to include descriptions in the report. Press Enter.

   This message appears:

   The requested model action completed successfully.

   The KWIC Index Report appears online and in a data set named userid.IEF.KWIC that you can print.

Steps to Print the KWIC Index Report

1. Press Exit. The Report Print Options Panel appears. The system automatically supplies the report's data set name on the request panel.
2. Choose to print batch or local.
3. Enter a sysout class or printer ID and press Enter. If a system printer is being used, enter a Job Statement.
4. Edit print job JCL if necessary.
5. On the command line, select the print option:
   - **PK**—Print and keep the data set
   - **PD**—Print the data set and delete it
   - **K**—Keep the data set without printing it

Chapter 2: Model Import Function

This chapter lists Entity Relationship Diagram (ERD) and External Design (DSD) object types contained in the Object Definition file. It also describes the rules and record formats for both record types.

Records in the Object Definition File

The Object Definition file contains records that define the following ERD object types:

- Subject areas
- Entity types
- Entity subtypes
- Attributes
- Aliases
- Permitted values
- Relationships
- Identifiers
- Functions
- Processes
- Expected Effects
- Group views
- Entity views
- Dependencies
- Work attribute sets

The Object Definition File also contains records for the following External Design (DSD) object types:

- Storage groups
- Databases
- Tablespaces
- Indexspaces
- Tablespace data sets

- Indexspace data sets
- Records
- Entity record implementations
- Fields
- Entry points
- Field entry point usages
- Field entry point values
- Relationship implementations
- Linkages
- Field linkage usages
- DASD volumes
- DASD volume usages
- Storage group/DASD volume usages

The following sections describe the rules and record formats for both ERD and DSD objects.

Subject Areas

Consider the following rules:

- Subject area names must be unique within a model.
- Except for the root subject area, each subject area definition record must include the name of its parent subject area.
- Each model has one root subject area. The root subject area definition record must appear before any other subject area.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	SUBJECT_AREA
OBJECT_NAME	Subject area name
REF_OBJ_1	Name of parent subject area, or blank for root subject area

Field Name	Content
REF_OBJ_2	Unused
REF_OBJ_3	Unused
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unused
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Entity Types

Consider the following rules:

- Entity type names must be unique among all entity types, subtypes, and work attribute sets within a model.
- Each entity type definition record must include the name of the subject area that contains this entity type.
- The DSD names must be unique among all DSD names for all entity types and subtypes.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	ENTITY_TYPE
OBJECT_NAME	Entity type name
REF_OBJ_1	Name of subject area that contains this entity type
REF_OBJ_2	DSD name (data structure name)
REF_OBJ_3	Unused
NUM_PROP_1	Minimum number of occurrences
NUM_PROP_2	Maximum number of occurrences
NUM_PROP_3	Average number of occurrences
NUM_PROP_4	Growth rate as a percentage
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Growth rate period: ■ Y (Year) ■ M (Month) ■ W (Week) ■ D (Day)
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Entity Subtypes

Consider the following rules:

- Subtype names must be unique among all entity types, subtypes, and work attribute sets within a model.
- Each subtype definition record must include the name of the parent entity or subtype that contains this subtype.
- The DSD names must be unique among all DSD names for all entity types and subtypes.
- The classifying attribute must be an attribute of the subtype's parent. Subtypes with no classifying attribute will be grouped together in one partitioning.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	ENTITY_SUBTYPE
OBJECT_NAME	Entity subtype name
REF_OBJ_1	Name of the parent entity type that contains this subtype
REF_OBJ_2	Classifying attribute name
REF_OBJ_3	Highest level entity type name that includes this subtype
NUM_PROP_1	Minimum number of occurrences
NUM_PROP_2	Maximum number of occurrences
NUM_PROP_3	Average number of occurrences
NUM_PROP_4	Growth rate (percent)
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused

Field Name	Content
CHAR_PROP_1	Growth rate period: ■ Y (Year) ■ M (Month) ■ W (Week) ■ D (Day)
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	DSD name (data structure name)
NAME_2	Unused

Attributes

Consider the following rules:

- Names of attributes must be unique among all attributes in the entity type's hierarchy
- Each attribute definition record must include the name of the parent entity or subtype that contains this attribute.
- The DSD names must be unique among all DSD names for all attributes in the entity type's hierarchy.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	ATTRIBUTE
OBJECT_NAME	Attribute name

Field Name	Content
REF_OBJ_1	Name of the parent entity type or subtype that contains this attribute
REF_OBJ_2	Highest level entity type name that includes this attribute
REF_OBJ_3	DSD name (data structure name)
NUM_PROP_1	Number of characters (length)
NUM_PROP_2	Number of decimal places
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Optionality: ■ M (Mandatory) ■ O (Optional)
CHAR_PROP_2	Type of attribute: ■ B (Basic) ■ D (Derived) ■ S (Designed)
CHAR_PROP_3	Domain of attribute: ■ T (Text) ■ D (Date) ■ M (Time) ■ N (Number) ■ Q (Timestamp) ■ G (Graphic [DBCS]) ■ Z (Mixed) ■ B (BLOB)
CHAR_PROP_4	Case sensitive? ■ Y (Yes) ■ N (No)

Field Name	Content
CHAR_PROP_5	Varying length string? ■ Y (Yes) ■ N (No)
CHAR_PROP_6	Units (for BLOB only): ■ K (KB) ■ M (MB) ■ G (GB) ■ Space (Bytes)
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Aliases

Consider the following rules:

- You may define alias names for attributes, subtypes, and entity types.
- Alias names must be unique within the parent entity type, subtype, or attribute.
- Each alias definition record must include the name of the parent entity or subtype for which it is an alias.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	ALIAS
OBJECT_NAME	Alias name
REF_OBJ_1	Attribute name, if alias is for attribute
REF_OBJ_2	Entity type or subtype name
REF_OBJ_3	Unused
NUM_PROP_1	Unused

Field Name	Content
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Abbreviation? Y (Yes) N (No)
CHAR_PROP_2	Acronym? Y (Yes) N (No)
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	First 32 characters of long name for alias
NAME_2	Next 32 characters of long name for alias

Permitted Values

Permitted values define valid inputs for an attribute and may be used to classify subtypes of the attribute's entity types.

Also be aware of the following rules:

- The permitted value should be valid for the owning attribute. For example, do not enter a character value for a numeric attribute or a value that exceeds the number of characters for an attribute.
- If several ranges exist for a permitted value, enter a definition record for each range. The ranges cannot overlap. Permitted values for character attributes cannot have ranges.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	PERMIT_VALUE
OBJECT_NAME	Unused
REF_OBJ_1	Name of entity or work attribute set described by attribute. Ex: CUSTOMER
REF_OBJ_2	Name of owning attribute. Ex: TYPE
REF_OBJ_3	Name of classified subtype. Ex: FOREIGN (optional)
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unused
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Classifying value (or low value of range)
NAME_2	Blank (or high value of range)

Relationships

The Relationship Definition names both the from and to entity types (or subtypes) and both the forward and backward relationship memberships. The combination of these four items must be unique but can be in any order.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	RELATIONSHIP
OBJECT_NAME	Relationship ID1
REF_OBJ_1	Name of from entity. Ex: CUSTOMER
REF_OBJ_2	Name of entity Ex: ORDER
REF_OBJ_3	Unused
NUM_PROP_12	Forward rel minimum number of occurrences
NUM_PROP_2	Forward rel maximum number of occurrences
NUM_PROP_3	Forward rel average number of occurrences
NUM_PROP_4	Forward rel percent optional
NUM_PROP_5	Backward rel minimum number of occurrences
NUM_PROP_6	Backward rel maximum number of occurrences
NUM_PROP_7	Backward rel average number of occurrences
NUM_PROP_8	Backward rel percent optional
CHAR_PROP_12	Forward rel transferable: Y (Yes) N (No)
CHAR_PROP_2	Forward rel optionality: M (Always) O (Sometimes)

Field Name	Content
CHAR_PROP_3	Forward rel cardinality: 1 (One) M (Many)
CHAR_PROP_4	Abs or est flag for min occurrences: A (Abs) E (Est)
CHAR_PROP_5	Backward rel transferable: Y (Yes) N (No)
CHAR_PROP_6	Backward rel optionality: M (Always) O (Sometimes)
CHAR_PROP_7	Backward rel cardinality: 1 (One) M (Many)
CHAR_PROP_8	Abs or est flag for max occurrences: A (Abs) E (Est)
NAME_1	Name of forward relationship. Ex: PLACES
NAME_2	Name of backward relationship: Ex: PLACED BY

You must include the relationship ID if the relationship:

- Is used to form part of an identifier (the relationship ID would match NAME_1 or NAME_2 of the IDENTIFIER record).
- Is used for a denormalized field (the relationship ID would match NAME_1 of the Field record).
- Is implemented by a relationship implementation (the relationship ID would match REF_OBJ_1 of the REL_IMPLEMENT record). If used, this ID must be unique within the model.
- All NUM_PROP and CHAR_PROP fields are optional.

Identifiers

Consider the following rules:

- An identifier must have at least one attribute or relationship name
- An entity can have only one primary identifier

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	IDENTIFIER
OBJECT_NAME	Ident ID1
REF_OBJ_1	Name of entity for which this is an identifier
REF_OBJ_2	Name of attribute 1 (optional)
REF_OBJ_3	Name of attribute 2 (optional)
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	If Relationship 1 (NAME_1) is used for a recursive relationship, which relationship membership should be used: ■ F (Forward) ■ B (Backward)
CHAR_PROP_2	If Relationship 2 (NAME_2) is used for a recursive relationship, which relationship membership should be used: ■ F (Forward) ■ B (Backward)

Field Name	Content
CHAR_PROP_3	Is this the primary identifier for the entity? ■ Y (Yes) ■ N (No)
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unique ID of relationship 1 (optional)
NAME_2	Unique ID of relationship 2 (optional)

Specify an Ident ID if:

- You cannot fully specify the IDENT in a single record (that is, if more than two attributes or two relationships are involved). When this occurs, specify the remainder of the identifier in consecutive IDENT records that have the same IDENT ID.
- The identifier is implemented by an entry point (the IDENT ID would match REF_OBJ_3 of the ENTRY_POINT record).
- The identifier is targeted by a linkage (the IDENT ID would match REF_OBJ_3 of the LINKAGE record).

Functions

Consider the following rules:

- Names of functions and processes must be unique within a model. That is, no function can have the same name as any other function or process.
- Except for the root function, each function or process definition record must include the name of its parent.
- Each model has one root function, the name of which must differ from the model name. The root function definition record must appear before any other function or process.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model

Field Name	Content
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	FUNCTION
OBJECT_NAME	Function name
REF_OBJ_1	Name of parent function (blank for root function)
REF_OBJ_2	Unused
REF_OBJ_3	Unused
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unused
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Processes

A process can be subordinate to another function or process, but an elementary process cannot have a subordinate.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	PROCESS
OBJECT_NAME	Name of the process
REF_OBJ_1	Name parent function or process
REF_OBJ_2	Unused
REF_OBJ_3	Unused
NUM_PROP_1	Usage props: average frequency
NUM_PROP_2	Usage props: maximum frequency
NUM_PROP_3	Usage props: minimum frequency
NUM_PROP_4	Usage props: percentage of growth
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Usage props: unit of frequency: ■ Y (Year) ■ M (Month) ■ W (Week) ■ D (Day)
CHAR_PROP_2	Usage props: growth unit: ■ Y (Year) ■ M (Month) ■ W (Week) ■ D (Day)
CHAR_PROP_3	Unused

Field Name	Content
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Elementary process indicator: ■ Y (Yes) ■ N (No)
CHAR_PROP_7	Repetition indicator: ■ Y (Yes) ■ N (No)
CHAR_PROP_8	Suggested mechanism: ■ L (Online) ■ M (Manual) ■ B (Batch) ■ O (Other)
NAME_1	Unused
NAME_2	Unused

Expected Effects

Expected effect records are optional. If more than one entity or entity subtype is affected by a function or process, create this record for each entity type.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	EXPECTED_EFFECT
OBJECT_NAME	Unused
REF_OBJ_1	Name of function or process that produces this effect
REF_OBJ_2	Name of entity or entity subtype affected by previous process
REF_OBJ_3	Unused
NUM_PROP_1	Unused

Field Name	Content
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	CREATE expected indicator: Y (Yes) N (No)
CHAR_PROP_2	DELETE expected indicator: Y (Yes) N (No)
CHAR_PROP_3	UPDATE expected indicator: Y (Yes) N (No)
CHAR_PROP_4	READ expected indicator: Y (Yes) N (No)
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Group Views

Consider the following rules:

- A group view may be subordinate to another group view. It may consist of several other group views or entity views.

- Within a function or process, group view names and entity view names must be unique in the following ways:
  - A group view name must be different from any other group view name or entity view name.
  - The entity view name and entity type combination must be unique.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	GROUP_VIEW
OBJECT_NAME	Name of the group view
REF_OBJ_1	Name of parent group view or blanks
REF_OBJ_2	Name of function or process for which this view is used
REF_OBJ_3	IMPORT or EXPORT
NUM_PROP_1	Expected average cardinality (Optional)
NUM_PROP_2	Expected maximum cardinality (Optional)
NUM_PROP_3	Expected minimum cardinality (Optional)
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Cardinality: 1 (One) M (Many)
CHAR_PROP_2	Abs or est flag for max cardinality: A (Absolute) E (Estimated)
CHAR_PROP_3	Abs or est flag for min cardinality: A (Absolute) E (Estimated)

Field Name	Content
CHAR_PROP_4	Horizon: (Optional) L (Logical) I (ID) N (None)
CHAR_PROP_5	Required input indicator: Y (Yes) N (No)
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Entity Views

Consider the following rules:

- An entity view may or may not belong to a group view
- If more than one attribute of a given entity is part of the same entity view, create one of these records for each attribute

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	ENTITY_VIEW
OBJECT_NAME	Name of the entity view
REF_OBJ_1	Name of parent group view (if any) or blanks
REF_OBJ_2	Name of function or process for which this view is used
REF_OBJ_3	IMPORT EXPORT ENTITY_ACTION

Field Name	Content
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Required input indicator: ■ Y (Yes) ■ N (No)
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Name of entity or entity subtype seen by this view
NAME_2	Name of attribute contained in the previous entity (see the previous rules)

Dependencies

Dependency records are optional. If supplied, they are used to create the portion of the model associated with the Process Dependency Diagram.

Note: Both the FROM and TO processes or functions must belong to the same parent.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model

Field Name	Content
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	DEPENDENCY
OBJECT_NAME	Name of dependency between two processes or functions (Optional)
REF_OBJ_1	Name of from process or function
REF_OBJ_2	Name of to process or function
REF_OBJ_3	Unused
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unused
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Work Attribute Sets

Names of work attribute sets must be unique among all entity types, entity subtypes, and work attribute sets.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	WORK_ATTR_SET
OBJECT_NAME	Name of the work attribute set
REF_OBJ_1	Unused
REF_OBJ_2	Unused
REF_OBJ_3	Unused
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unused
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Storage Groups

Consider the following rules:

- The storage group name must be unique within the model
- The storage group name contains a maximum of eight characters

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	STORAGE_GROUP
OBJECT_NAME	Name of the storage group
REF_OBJ_1	Password
REF_OBJ_2	Unused
REF_OBJ_3	Unused
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unused
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Name of VSAM catalog

Field Name	Content
NAME_2	DB2 authorization user ID

Databases

The database name must be unique within the model. The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	DATA_BASE
OBJECT_NAME	Name of the database
REF_OBJ_1	Name of the storage group that is used as a default for the database (Optional)
REF_OBJ_2	Unused
REF_OBJ_3	Unused
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Default bufferpool: 0 (BP0) 1 (BP1) 2 (BP2) K (BP32K)
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused

Field Name	Content
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Tablespaces

Consider the following rules:

- The data store tablespace must be grouped by a database
- The data store tablespace must contain a data set
- The data store tablespace name must be unique within the model

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	TABLE_SPACE
OBJECT_NAME	Name of the data store tablespace
REF_OBJ_1	Name of database that the tablespace is grouped by.
REF_OBJ_2	Name of storage group that contains the tablespace (Optional)
REF_OBJ_3	Name of VSAM catalog
NUM_PROP_1	Segment size
NUM_PROP_2	Free page frequency

Field Name	Content
NUM_PROP_3	Free space percentage
NUM_PROP_4	Primary allocation quantity
NUM_PROP_5	Secondary allocation quantity
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Bufferpool assignment: ■ 0 (BP0) ■ 1 (BP1) ■ 2 (BP2) ■ K (BP32K)
CHAR_PROP_2	Close when not in use? ■ Y (Yes) ■ N (No)
CHAR_PROP_3	Lock size: ■ P (Page) ■ A (Any) ■ T (Table)
CHAR_PROP_4	Erase old data during Create: ■ Y (Yes) ■ N (No)
CHAR_PROP_5	Unit for allocation: ■ C (Cyl) ■ T (Trk) ■ R (Rec) ■ K (Kbytes)
CHAR_PROP_6	Use default data set properties? ■ Y (Yes) ■ N (No)

Field Name	Content
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Password
NAME_2	Unused

Indexspaces

Consider the following rules:

- The data store indexspace must be grouped by a database
- The data store indexspace name must be unique within the model
- The indexspace name must have exactly the same name (8 characters) as the index (entry point) stored in it
- The indexspace must contain a data set

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	INDEX_SPACE
OBJECT_NAME	Name of the data store indexspace
REF_OBJ_1	Name of database that the indexspace is grouped by
REF_OBJ_2	Name of storage group that contains the indexspace (Optional)
REF_OBJ_3	Name of VSAM catalog
NUM_PROP_1	Free page frequency
NUM_PROP_2	Free space percentage
NUM_PROP_3	Primary allocation quantity
NUM_PROP_4	Secondary allocation quantity

Field Name	Content
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Bufferpool assignment: ■ 0 (BP0) ■ 1 (BP1) ■ 2 (BP2) ■ K (BP32K)
CHAR_PROP_2	Close when not in use? ■ Y (Yes) ■ N (No)
CHAR_PROP_3	Number of subpages per page (1, 2, 4, 8, S-16)
CHAR_PROP_4	Erase old data during Create? ■ Y (Yes) ■ N (No)
CHAR_PROP_5	Unit for allocation quantity: ■ C (Cyl) ■ T (Trk) ■ R (Rec) ■ K (Kbytes)
CHAR_PROP_6	Use default data set properties? ■ Y (Yes) ■ N (No)
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Password
NAME_2	Unused

Tablespace Data Sets

The tablespace data set must be contained by a database and a tablespace.

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	DATASET_TABLE
OBJECT_NAME	Dataset_table ID1
REF_OBJ_1	Name of data store tablespace that contains this data set
REF_OBJ_2	Name of storage group that holds the tablespace data set (Optional)
REF_OBJ_3	Name of VSAM catalog
NUM_PROP_1	Free page frequency
NUM_PROP_2	Free space percentage
NUM_PROP_3	Primary allocation quantity
NUM_PROP_4	Secondary allocation quantity
NUM_PROP_5	Partition sequence number within tablespace2
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unit for allocation quantity: C (Cyl) T (Trk) R (Rec) K (Kbytes)
CHAR_PROP_2	Erase old data during Create? Y (Yes) N (No)
CHAR_PROP_3	Use default data set properties? Y (Yes) N (No)
CHAR_PROP_4	Unused

Field Name	Content
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Name of the database
NAME_2	Unused

The dataset_table ID is required if the data set uses DASD volumes (the dataset_table ID would match REF_OBJ_2 of the DASD_VOL_USAGE record).

The partition sequence number is required if tablespace contains multiple data sets. The DATASET_TABLE records must be specified with the partition numbers in consecutive order within a tablespace.

Records

Consider the following rules:

- Record names and table names must be unique within a model.
- A data record must be contained by a data store tablespace.
- A data record must have an entity record implementation.
- Each linkage from a linkage record must connect the records that implement the entities that participate in the many-to-many relationship being implemented.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	RECORD
OBJECT_NAME	Name of the record
REF_OBJ_1	Name of data store tablespace that contains this record
REF_OBJ_2	Table name for the record (macro name)
REF_OBJ_3	Name of the edit proc routine (optional)

Field Name	Content
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Role (mandatory): ■ D (Data record) ■ L (Linkage record)
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Name of the validation proc routine (Optional)
NAME_2	Owner ID

Entity Record Implementations

Consider the following rules:

- A data record must have at least one entity record implementation.
- Each entity type can be implemented by only one data record.
- The entity implemented by the data record can be an entity type or subtype but all subtypes of the same entity type must use the same data record.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	ENTITY_REC_IMP
OBJECT_NAME	Unused
REF_OBJ_1	Name of the entity type or subtype implemented by the record
REF_OBJ_2	Name of the data record defined by this entity or subtype
REF_OBJ_3	Unused
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unused
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Fields

Consider the following rules:

- Names of fields and columns must be unique within a record.
- The attribute implemented by a data field must belong to the entity implemented by the data record.
- The attribute implemented by a denormalized field must be an attribute (or inherited attribute) of the entity participating in the relationship the field is denormalized along.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	FIELD
OBJECT_NAME	Name of the field
REF_OBJ_1	Name of the record that contains this field
REF_OBJ_2	Name of the attribute that this field implements
REF_OBJ_3	Name of the entity or subtype (parent) described by attribute
NUM_PROP_1	Length
NUM_PROP_2	Number of decimal places
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Role: P (Primary data implementation) F (Foreign key) D (Denormalized)

Field Name	Content
CHAR_PROP_2	Format: P (Packed) X (Text) S (Small) I (Integer) V (Varchar) Q (Timestamp) F (Floating point) D (Date) T (Time) L (Long varchar) B (BLOB)
CHAR_PROP_3	Database field optionality: O (Optional) M (Mandatory)
CHAR_PROP_4	If this is a denormalized field, which relationship membership should be used? F (Forward) B (Backward)
CHAR_PROP_5	Units (for BLOB only): ■ K (KB) ■ M (MB) ■ G (GB) ■ Space (Bytes)
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unique ID of relationship (if this field is denormalized)
NAME_2	Column name for field (macro name)

Entry Points

Consider the following rules:

- Names of entry points must be unique within a model.
- The name of the entry point must be the same as that of the data store indexspace.
- A record can have only one primary key.
- A record can have only one clustered entry point.
- A record can have only one partitioned entry point.
- If an entry point implements a many-to-many relationship implementation, then the entry point must be on a link record.
- An entry point that implements an identifier must match the identifier.
- If the entry point was added for performance reasons, then the identifier ID and the relationship ID fields should be blank.
- For an entry point to be partitioned, the data store indexspace must contain multiple data sets.
- For a non-partitioned entry point, the data store indexspace cannot contain multiple data sets.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	ENTRY_POINT
OBJECT_NAME	Name of the entry point
REF_OBJ_1	Name of the data store indexspace that stores this entry point
REF_OBJ_2	Name of the record referenced by this entry point
REF_OBJ_3	Unique ID of identifier, if entry point implements an identifier
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused

Field Name	Content
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Is the entry point clustered? ■ Y (Yes) ■ N (No)
CHAR_PROP_2	Is this entry point unique? ■ Y (Yes) ■ N (No)
CHAR_PROP_3	Is this the primary key for the record? ■ Y (Yes) ■ N (No)
CHAR_PROP_4	Is this entry point partitioned? ■ Y (Yes) ■ N (No)
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unique ID of the relationship implementation (entry point or many-to-many), if the entry point is associated with a relationship implementation
NAME_2	Unused

Field Entry Point Usages

Consider the following rules:

- Each entry point must use at least one field.
- If more than one field exists for an entry point, specify the remainder of the fields in consecutive FLD_ENTRYPT_US records that have the same entry point.

- The field must be contained in the same record that is referenced by the entry point.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	FLD_ENTRYPT_US
OBJECT_NAME	Unused
REF_OBJ_1	Name of the entry point
REF_OBJ_2	Name of the field used by this entry point
REF_OBJ_3	Unused
NUM_PROP_1	Sequence number of field within entry point1
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Sequence type: ■ A (Ascending) ■ D (Descending)
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Note: The sequence number is required if the entry point contains more than one field. The sequence number must be specified in consecutive order within an entry point.

Field Entry Point Values

Consider the following rules:

- If the entry point contains more than four fields, specify the remainder of the field entry point values in consecutive FLD_ENTPT_VALUE records that have the same entry point name.
- Field entry point values cannot be given for non-partitioned entry points.
- All partitions for an entry point except the last partition must have field entry point values (partitioning limit values).
- The order of the values for the fields must be in the same order that the FLD_ENTRYPT_US records are specified for the entry point. For example, the value for field 1 would be for the field entry point usage with sequence number 1.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	FLD_ENTPT_VALUE
OBJECT_NAME	Unused
REF_OBJ_1	Name of the entry point
REF_OBJ_2	Partitioning limit value for field 1 of entry point
REF_OBJ_3	Partitioning limit value for field 2 of entry point
NUM_PROP_1	Partition sequence number
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unused

Field Name	Content
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Partitioning limit value for field 3 of entry point
NAME_2	Partitioning limit value for field 4 of entry point

Relationship Implementations

Consider the following rules:

- Only one relationship implementation is allowed per relationship membership.
- A many-to-many relationship implementation must be associated with one linkage and one entry point.
- A foreign key relationship implementation must be associated with one linkage.
- An entry point implementation must be associated with one entry point.
- A many-to-many relationship implementation technique is valid only for many-to-many relationships.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	REL_IMPLEMENT
OBJECT_NAME	REL_IMPLEMENT ID (mandatory, must be unique)
REF_OBJ_1	Unique ID of the relationship implemented
REF_OBJ_2	Unused
REF_OBJ_3	Unused

Field Name	Content
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Technique (mandatory): ■ F (Foreign key) ■ E (Entry point) ■ M (Many-to-many)
CHAR_PROP_2	Which relationship membership should be used? ■ F (Forward) ■ B (Backward)
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Linkages

Consider the following rules:

- The linkage must connect records implementing the entities that participate in the relationship that the relationship implementation represents.
- A linkage must target an identifier that identifies the to record.
- The relationship implementation must be many-to-many technique or foreign key technique.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	LINKAGE
OBJECT_NAME	Linkage ID (mandatory, must be unique)
REF_OBJ_1	Name of the record this linkage is from (mandatory)
REF_OBJ_2	Name of the data record this linkage is to (mandatory)
REF_OBJ_3	Unique ID of the identifier that this linkage targets (mandatory) (The identifier of the to record is used to choose the foreign key fields.)
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Database or system-enforced: ■ I (System) ■ D (Database)
CHAR_PROP_2	Referential constraint option: ■ D (Cascade delete) ■ R (Restrict) ■ N (Nullify)
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused

Field Name	Content
CHAR_PROP_8	Unused
NAME_1	Unique ID of the relationship implementation this linkage is used in (mandatory)
NAME_2	Name of the linkage

Field Link Usages

Consider the following rules:

- REF_OBJ_1 must be a foreign key field in the from record.
- REF_OBJ_2 must be a data field or foreign key field in the to record.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	FIELD_LINK_USE
OBJECT_NAME	Unused
REF_OBJ_1	Name of the foreign key field in the from record (see note 1)
REF_OBJ_2	Name of the foreign key field in the to record (see note 2)
REF_OBJ_3	Unique ID of the linkage that uses this usage
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Unused

Field Name	Content
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

DASD Volumes

Consider the following rules:

- The DASD_volume ID and volume serial number are required.
- The volume serial number can have a maximum of 6 characters.

The following table describes the content of each field type:

Field Name	Content
MODEL_NAME	Host Encyclopedia name of the model
RECORD_TYPE	O (Object Record)
OBJECT_TYPE	DASD_VOLUME
OBJECT_NAME	Dasd_volume ID
REF_OBJ_1	Volume serial number
REF_OBJ_2	Unused
REF_OBJ_3	Unused
NUM_PROP_1	Unused
NUM_PROP_2	Unused
NUM_PROP_3	Unused
NUM_PROP_4	Unused
NUM_PROP_5	Unused

Field Name	Content
NUM_PROP_6	Unused
NUM_PROP_7	Unused
NUM_PROP_8	Unused
CHAR_PROP_1	Type of DASD device: ■ 8 (3380) ■ 5 (3350)
CHAR_PROP_2	Unused
CHAR_PROP_3	Unused
CHAR_PROP_4	Unused
CHAR_PROP_5	Unused
CHAR_PROP_6	Unused
CHAR_PROP_7	Unused
CHAR_PROP_8	Unused
NAME_1	Unused
NAME_2	Unused

Order of Input Records

Records in the Object Definition File must be ordered so that each object definition precedes any other object definitions that refer to it.

Records in the Description File must appear in the same order as the objects they describe. It is not necessary to have a description for each object or for any objects at all, but if descriptions are supplied, they must be in the same order as the corresponding object definition records.

A change in the model name causes the import program to create another model with the new name.

To order the records, you must use one of two sequences. Either sequence ensures that objects are loaded before they are referenced.

Import Model into Host Encyclopedia

Import Model creates a CA Gen model from files formatted, named and arranged according to the directions in this chapter. The import function is used only to create a new CA Gen model, not to modify or delete an existing model.

1. Choose Option **2** from the Host Main menu.
2. Choose **3** to Import Model into Host Encyclopedia. The Import Model into Host Encyclopedia Panel appears.
3. Enter name of the Object Definition file (IEF.PIOBJ).
4. Enter name of the description file (IEF.PIDESC).

Appendix A: Public Interface Tables List

This appendix contains a list of the public interface tables.

Name	Remarks
ACTION_BLOCK	Action block
ACTIV_USAGE	Activity usage (functions and processes)
ACTIVITY_CLUSTER	Activity cluster
ACTN_BLK_USE	Action block usage
ADD_ATOM_DEP	Activity Dependency Diagram (ADD) atomic dependency
ADD_CLOSURE	ADD closure
ADD_DEPENDNCY	ADD dependency
ADD_EXT_FLOW	ADD external flow
ADD_MUTL_EXCL	ADD mutually exclusive construct
ADD_PARALLEL	ADD parallel construct
ALIAS	Alias or alternate name
ATTR_IDENT	Attribute identifier
ATTR_VIEW	Attribute view definition
ATTRIBUTE	Attribute of an entity type, subtype, or work attribute set
BAA_ACTN_BLK	Analysis action block definition
BSD_ACTN_BLK	Design action block definition
BUS_AREA	Business area
BUS_GOAL	Business goal
BUS_LOCATION	Location of business assets
BUS_OBJECTIVE	Business objective
BUS_PROC_SCOPE	Procedure scoping
BUS_PROC_STEP	Procedure step
BUS_SYS_SCOPE	Business system scoping
BUSINESS_PROC	Procedure

Name	Remarks
BUSINESS_SYS	Business system
CD_ACTN_BLK	Cascade delete action block
CELL_VALUE	Cell value in matrix
CHANGE_OCCUR	Change occurrence between session and object
CLASSIFIER	Classifier of a partitioning
CMD_SYNONYM	Command synonym
COMMAND	Command
COMPONENT_IMPLEM	Component implementation
COMPONENT_SPEC	Component specification
CRITICAL_SUCCESS	Critical success factor
CSTM_EDT_PTRN	Custom edit pattern
CURRENT_DATA	Current database or store
CURRENT_EFFECT	Current system/data effect
CURRENT_INFO_SYS	Current information system
DASD_VOL_USAGE	DASD volume usage
DASD_VOLUME	DASD volume
DATA_BASE	Database
DATA_BASE_USAGE	Database usage
DATA_CLUSTER	Data cluster (natural data store)
DATA_STORE_INDEX	Indexspace
DATA_STORE_TBLSP	Tablespace
DATASET_INDEX	Indexspace data set
DATASET_TBLSP	Tablespace data set
DB2_DDL_DB	Member name containing generated DDL for database
DB2_DDL_INDEX	Member name containing generated DDL for indexspace
DB2_DDL_TABLE	Member name containing generated DDL for table
DB2_DDL_TBLSP	Member name containing generated DDL for tablespace

Name	Remarks
DB2_DEF_CLUSTER	Member name containing VSAM define cluster for tablespace in index space
DB2_MVS_COLUMN	Field DB2 Extension (FIELDB)
DATACOM_COLUMN	Field DATACOM Extension (FIELDA)
DATACOM _CONSTRNT	Linkage DATACOM Extension (LINKFA)
DATACOM _DATABASE	Database DATACOM Extension (DATBASA)
DATACOM _INDEX	Entry Point DATACOM Extension (ENTPNTA)
DATACOM _TABLE	Record DATACOM Extension (RECIEFA)
DATACOM _TD	Technical Design DATACOM Extension (TCHDSNA)
DB2_MVS_CONSTRNT	Linkage DB2 Extension (LINKFB)
DB2_MVS_DATABASE	Database DB2 Extension (DATBASB)
DB2_MVS_INDEX	Entry Point DB2 Extension (ENTPNTB)
DB2_MVS_INDEXSPC	Indexspace DB2 Extension (DATSTBI)
DB2_MVS_TABLE	Record DB2 Extension (RECIEFB)
DB2_MVS_TABLESPC	Tablespace DB2 Extension (DATSTBT)
DB2_MVS_TD	Technical Design DB2 Extension (TCHDSNB)
DB2_RESRC_MODULE	DB2 resource module
DERIVATION_ALGOR	Derivation algorithm action block
DESC	Description
DFLT_EDT_PTRN	Default edit pattern
DFLT_LIT_VDAT	Default literal video attributes
DFLT_PRM_VDAT	Default prompt video attributes
DFLT_VAR_VDAT	Default variable video attributes
DFLT_VARE_VDAT	Default variable error field attributes
DIALECT	Dialect
DIALECT_TEXT	Dialect-specific text
DIALOG_FLOW	Dialog flow
DLG_DATA_SENT	Dialog flow - data sent
DLG_FLWS_EXST	Dialog flow - Flows on exit state
DLG_SETS_CMD	Dialog flow - Sets command

Name	Remarks
ENTITY_ST_TRANS	Entity state transition
ENT_ST_TRANS_USE	Entity state transition usage
ENTITY_REC_IMPL	Entity to record implementation
ENTITY_SUBTYP	Entity subtype
ENTITY_TYPE	Entity type
ENTITY_VIEW	Entity view definition
ENTRY_POINT	Record entry point
ENVIRONMENT	Hardware/software environment
EXIT_STATE	Exit state
EXIT_STATE_US	Exit state usage
EXPECT_EFFECT	Expected effect
FACILITY	Computing/communication facility
FIELD	Field definition
FLD_ENTPT_VALUE	Field entry point value
FLD_ENTRY_PT_USE	Field entry point usage
FLD_LINK_USE	Foreign key field linkage usage
FUNCTION_DEF	Function definition
GROUP_VIEW	Group view definition
IDENTIFIER	Attribute or relationship identifier
IMPLEMENT_LOGIC	Implementation logic
IMPL_LOGIC_USAGE	Implementation logic usage
INFORMATION_NEED	Information need
INTERFACE_TYPE	Interface type
INTRFCE_TYPE_MDL	Interface type model
LIB_USAGE_SCOPE	Library usage scope
LIBRARY	Library
LIBRARY_USAGE	Library usage
LINKAGE	Linkage between records
LNK_DATA_RTND	Link dialog flow data returned
LNK_RTNS_CMD	Link dialog flow returns command

Name	Remarks
LNK_RTNS_EXST	Link dialog flow returns on exit state
LOCAL_PF_KEY	Local PF key
MATRIX	Matrices
MATRIX_USAGE_X	Matrix usage on the X axis
MATRIX_USAGE_Y	Matrix usage on the Y axis
MESSAGE	Message
MODEL	Model definition
OBJECT_CLASS	System-defined class or user-defined class
ORGANIZAT_UNIT	Organizational unit
PAD_CREATE	Action diagram Create
PAD_DELETE	Action diagram Delete
PAD_FUNCTION	System-defined function
PAD_READ	Action diagram Read
PAD_SET_ATTR	Action diagram Set attribute
PAD_UPDATE	Action diagram Update
PARM	Parm
PARM_DELIMITER	Parameter delimiter
PARM_STRING_DEL	Parameter string delimiter
PARTITIONING	Partitioning of entity type or subtype
PDD_ATOM_DEP	PDD atomic dependency
PDD_CLOSURE	PDD closure of a mutually exclusive construct
PDD_DEPENDNCY	PDD dependency
PDD_EXT_FLOW	PDD external flow
PDD_MUTL_EXCL	PDD mutually exclusive construct
PDD_PARALLEL	PDD parallel construct
PERFORM_MEASURE	Performance measure
PERMIT_VALUE	Permitted values for an attribute
PERMIT_VALUE_HI	Permitted value, high end
PERMIT_VALUE_LOW	Permitted value, low end
PROCESS_DEF	Process definition

Name	Remarks
PROMPT	Attribute prompt
REC_ENTRY_PT_USE	Record entry point usage
RECORD	Record definition
RECORD_REFERENCE	Record reference
REL_IDENT	Relationship identifier
REL_MUTL_EXCL	Mutually exclusive relationship
REL_PART_IMPL	Relationship partition implementation
REL_VIEW	Relationship view
RELATIONSHIP	Relationship of an entity type or subtype. Another row represents the inverse (destination) of this relationship. The properties are for the source relationship.
SCREEN_DEF	Screen definition
SCREEN_TMPLT	Screen template
SCRN_FLD_LIT	Literal screen field
SCRN_FLD_PRMT	Prompt screen field
SCRN_FLD_VAR	Variable screen field
SCRN_FLD_VARE	Variable screen field, error video attributes
SCRN_FLD_VARP	Variable screen field, video attributes
SCRN_HELP	Screen Help identifier
SCRN_RG_OCC	Screen repeating group occurrence
SCRN_RP_GRP	Screen repeating group definition
SCRN_SYS_DEF	Screen system-defined field definition
SCRN_VAR_DEF	Screen variable definition
SCRN_VAR_IO	Screen variable input/output
SCROLL_AMOUNT	Scroll amount value
SESSION	Session
SPEC_TYPE	Specification type
STG_DVOL_USAGE	Storage group/DASD volume usage
STORAGE_GROUP	Storage group
STRATEGY	Strategy

Name	Remarks
SUBJECT_AREA	Subject area
SYS_ATTRIBUTE	System-defined attribute
SYS_ENT_TYPE	System-defined entity type (Work attribute set)
SYSTEM_PF_KEY	System PF key
TACTIC	Tactic
TD_LIBRARY_USAGE	Technical Design/Library usage
TECHNICAL_DESIGN	Technical Design
TECHNICAL_SYSTEM	Technical System
TEXT	Descriptions
TMPLT_USAGE	Screen template usage
UNFORMAT_INPUT	Unformatted input
UNFRMT_INP_USAGE	Unformatted input usage
USE_DATA_RTND	Action block usage, data returned
USE_DATA_SENT	Action block usage, data, sent
USER_DEF_OBJECT	User-defined object for matrices
VIEW_SET	View set
XT_EVNT_USAGE	External event usage
XT_IMPL_LOGIC	External implementation logic
XT_OBJ_USAGE	External object usage
XTERNL_EVENT	External event
XTERNL_OBJECT	External object

Appendix B: Relationships Between Public Interface Tables

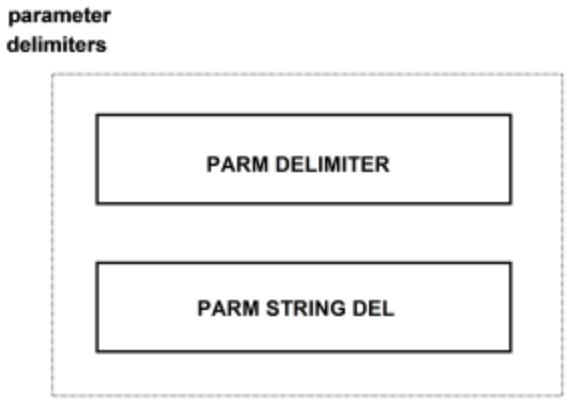
This appendix contains design and construction data models that show the interrelationships between Public Interface (PI) tables.

This section contains the following topics:

Interpreting Entity Type Names

In the data models, if an entity type's name is uppercase, the entity type represents a Public Interface table.

If an entity type's name is lowercase, the entity type represents a generalization, not a PI table. For example, in the illustration for business system defaults, the tables PARM_DELIMITER and PARM_STRING_DEL are subtypes of the generalization, parameter delimiters. Parameter delimiters is not a table name and appears in lowercase, as shown in the following illustration:



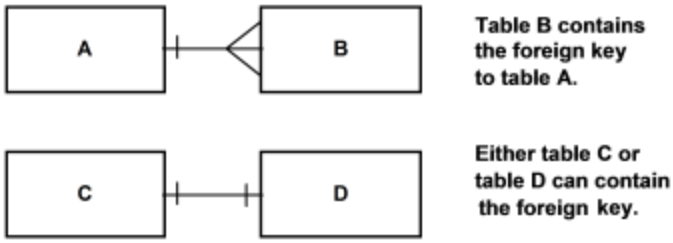
The dashed line represents a partitioning into subtypes.

Interpreting Relationships

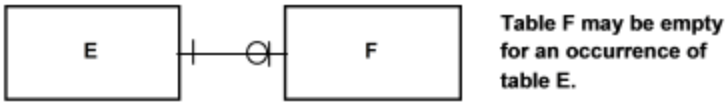
Two concepts underlie the representation of relationships between entity types: cardinality and optionality. For data models:

- *Cardinality* is the number of times an entity can participate in a relationship.
- *Relationship* optionality is membership of an entity type in a relationship such that entities of the type can exist without participating in a pairing under the relationship.

Each relationship in the data models represents an SQL join. Relationships in the data models show both cardinality and optionality. When a relationship has a one-to-many cardinality, the foreign key is on the many side. In a one-to-one relationship, the foreign key can be on either side. The following graphic is a pictorial depiction.



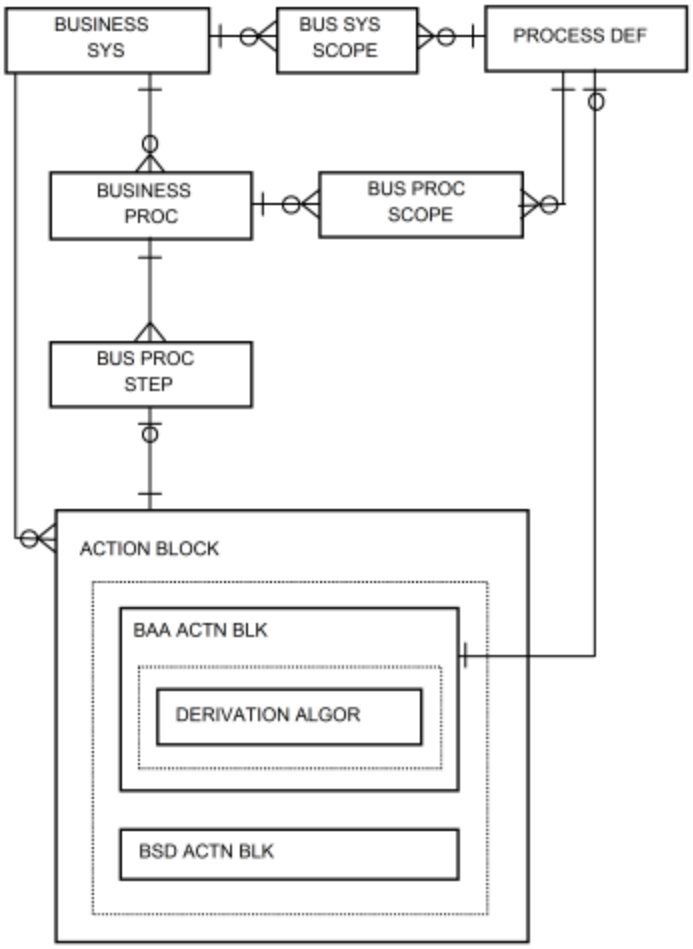
The optionality symbol (a circle on the relationship line) shows whether a table can be empty for a particular occurrence and therefore return no data in an SQL query. The following graphic is a pictorial depiction.



When a relationship is optional on one side, the foreign key is on the optional side. In the example, F contains the foreign key.

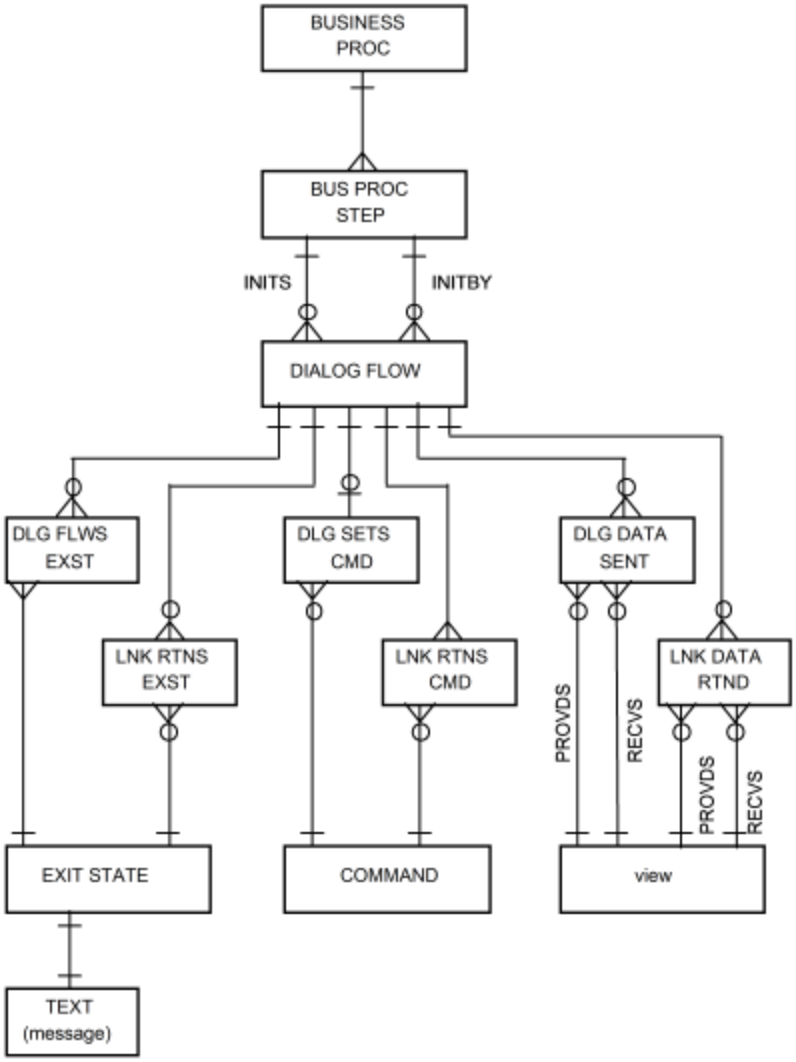
Business System Definition

The following graphic depicts a business system definition.



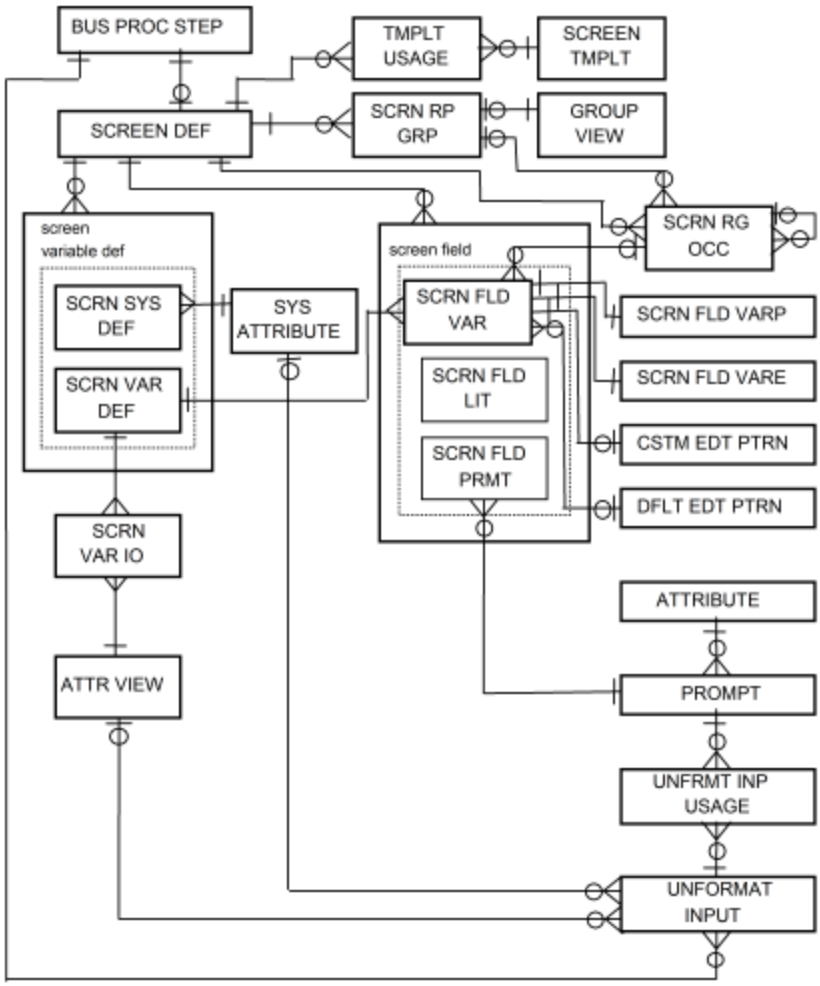
Dialog Flow Model

The following graphic depicts a dialog flow model.



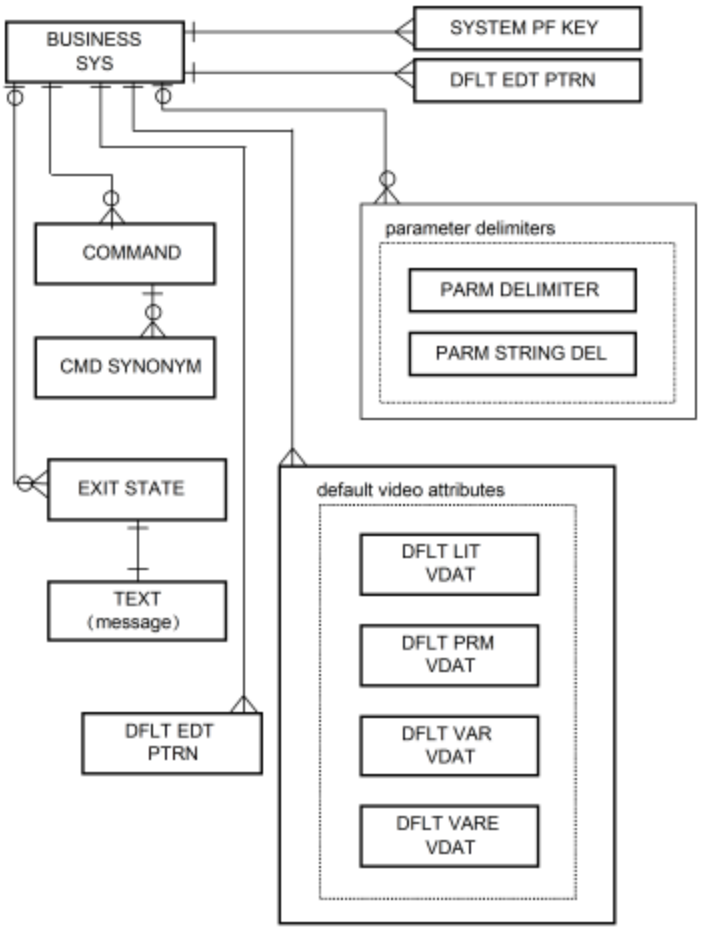
Screen Definition

The following graphic depicts a screen definition.



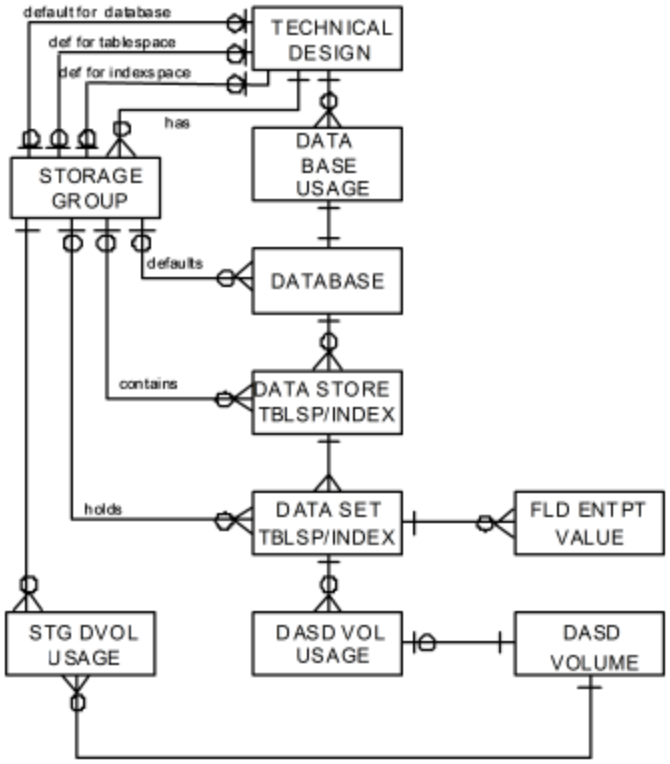
Business System Defaults

The following graphic depicts business system defaults.



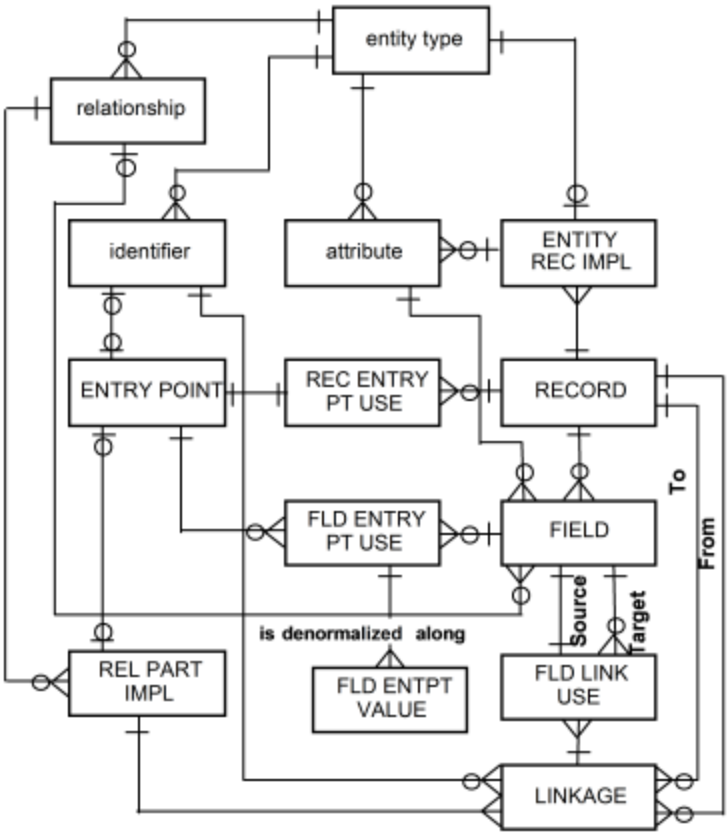
Data Structure Diagram: Internal Schema Definition Objects

The following graphic depicts internal schema definition objects.



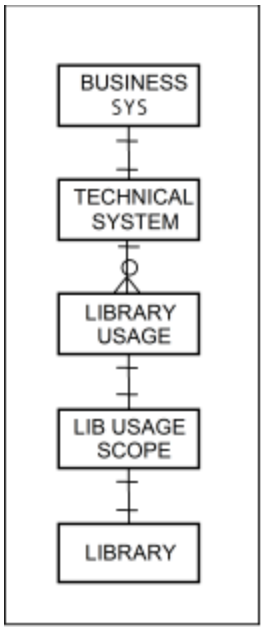
Data Structure Diagram: External Schema Definition Objects

The following graphic depicts external schema definition objects.



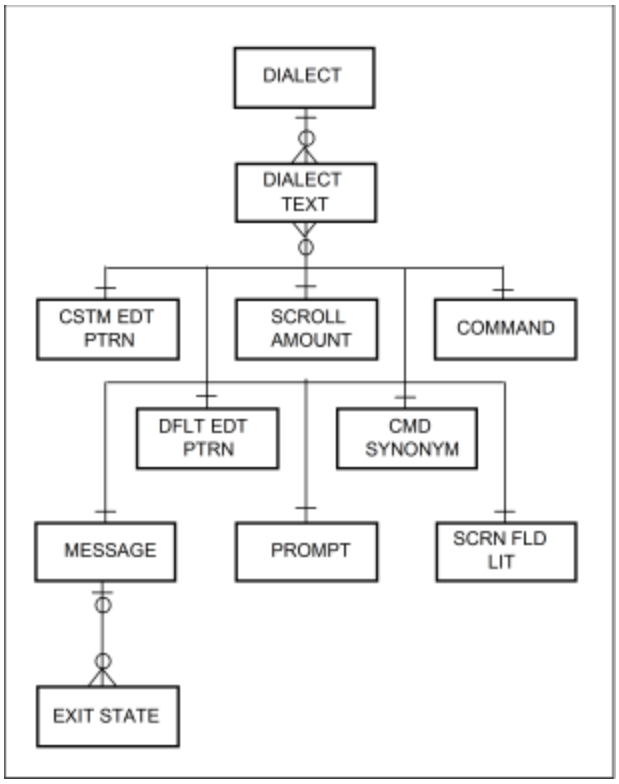
Construction Management

The following graphic depicts construction management.



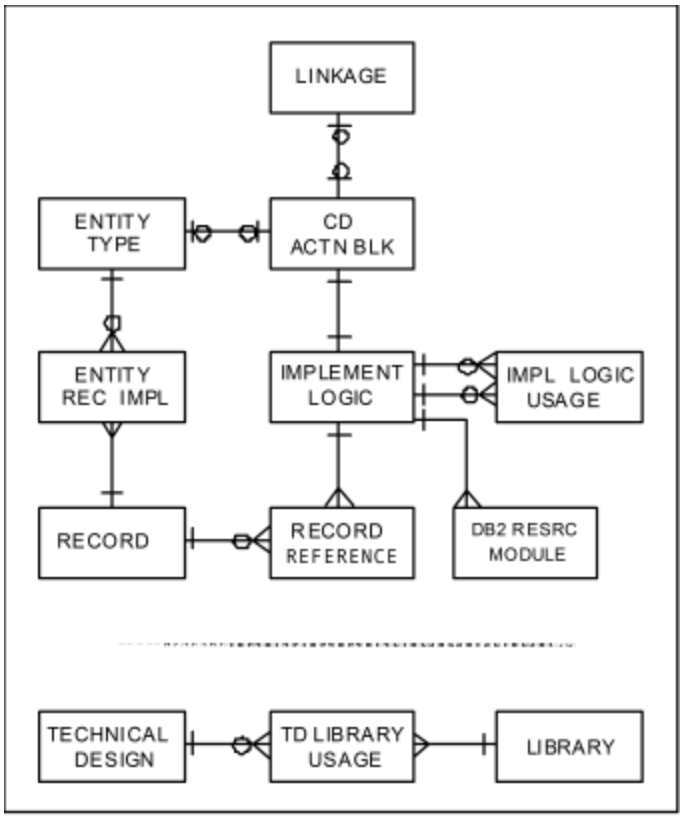
Dialect

The following graphic depicts a dialect.



Cascade Delete Support

The following graphic depicts a cascade delete support.



Appendix C: Public Interface Table Definitions

This appendix lists the Public Interface Table definitions alphabetically.

ACTION_BLOCK

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ACTION_BLOCK
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Action block name
6	INTEXT	1	CHAR	I (Internal) E (External)
7	BUSINESS_SYS_ ID	4	INTEGER	Business system that directly owns this action block. (Zero if not directly owned by a business system or if it is a procedure action block.)

ACTIV_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
2	TBNAME	16	CHAR	ACTIV_USAGE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier

Col	Column Name	Len	Coltype	Description
5	SEQH	4	INTEGER	Sequence of child within activity hierarchy. Use ORDER_BY_SEQH to return the activity hierarchy as specified in toolset.
6	SEQ	2	SMALLINT	Sequence within parent activity. Use ORDER_BY_SEQ to return children in the same order as specified in toolset.
7	MIN_FREQ	4	INTEGER	Minimum frequency of execution
8	MAX_FREQ	4	INTEGER	Maximum frequency of execution
9	AVG_FREQ	4	INTEGER	Average frequency of execution
10	FREQ_UNITS	1	CHAR	Frequency units: Y (Year) M (Month) W (Week) D (Day)
11	GROWTH_RATE	4	INTEGER	Growth rate (percent)
12	GROWTH_RATE_PER	1	CHAR	Growth rate period: Y (Year) M (Month) W (Week) D (Day)
13	PARENT_ACTIV_ID	4	INTEGER	Parent activity ID. Join with: FUNCTION_DEF PROCESS_DEF Zero for root function
14	CHILD_ACTIV_ID	4	INTEGER	Child activity ID. Join with: FUNCTION_DEF PROCESS_DEF

ACTIVITY_CLUSTER

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ACTIVITY_CLUSTER
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the activity cluster

ACTN_BLK_USE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ACTN_BLK_USE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ACTN_BLK_ID	4	INTEGER	Action Block that contains the USE action. Join with ACTION_BLOCK.
6	USED_ACTN_BLK_ID	4	INTEGER	The used Action Block. Join with ACTION_BLOCK.

ADD_ATOM_DEP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ADD_ATOM_DEP

Col	Column Name	Len	Coltype	Description
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ADD_DEPENDNCY_ID	4	INTEGER	Dependency ID
6	EXPORT_USAGE_ ID	4	INTEGER	From side of dependency. Join with: ACTIV_USAGE XT_EVNT_USAGE XT_OBJ_USAGE ADD_MUTL_EXCL ADD_PARALLEL ADD_CLOSURE
7	IMPORT_USAGE_ ID	4	INTEGER	To side of dependency. Join with: ACTIV_USAGE XT_OBJ_USAGE ADD_MUTL_EXCL ADD_PARALLEL ADD_CLOSURE

ADD_CLOSURE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ADD_CLOSURE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARENT_ ACTIV_ID	32	VARCHAR	Parent activity ID. Join with: FUNCTION_DEF PROCESS_DEF
6	PDD_MUTL_ EXC_ID	4	INTEGER	ADD mutually exclusive construct being closed

ADD_DEPENDNCY

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ADD_DEPENDNCY
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Dependency name

ADD_EXT_FLOW

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ADD_EXT_FLOW
3	ADD_DEPENDNCY_ID	4	INTEGER	ID of the dependency
4	DATA_VIEW_ID	32	VARCHAR	Data view ID that flows on this flow. Join with: ■ GROUP_VIEW ■ ENTITY_VIEW

Note: For ADD_MUTL_EXCL, see the PDD_MUTL_EXCL (see page 284) table. For ADD_PARALLEL, see the PDD_PARALLEL (see page 284) table.

ALIAS

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model

Col	Column Name	Len	Coltype	Description
2	TBNAME	16	CHAR	ALIAS
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of alias
6	ABBREVIATION	1	CHAR	Abbreviation? Y (Yes) N (No)
7	ACRONYM	1	CHAR	Acronym? Y (Yes) N (No)
8	USE_IN_DSD	1	CHAR	Use alias name in DSD? Y (Yes) N (No)
9	DATA_ITEM_ID	4	INTEGER	ID of data item. Join with: ENTITY_TYPE ENTITY_SUBTYP ATTRIBUTE

ATTR_IDENT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	D of the containing model
2	TBNAME	16	CHAR	ATTR_IDENT
3	ID	4	INTEGER	Identifier (not unique)
4	ORG_ID	4	INTEGER	Original object ID
5	SEQ	2	SMALLINT	Sequence within entity type or subtype. Use ORDER_BY_SEQ to return attribute identifiers in the same order as specified in the toolset.
6	NAME	32	VARCHAR	Name of identifier

Col	Column Name	Len	Coltype	Description
7	PRIMARY_KEY	1	CHAR	Primary identifier for entity type? Y (Yes) N (No)
8	ENTITY_ID	4	INTEGER	Entity type or subtype for which this attribute is an identifier. Join with: ENTITY_TYPE ENTITY_SUBTYP
9	ATTRIBUTE_ ID	4	INTEGER	Attribute ID

ATTR_VIEW

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ATTR_VIEW
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence within entity view. Use ORDER_BY_SEQ to return attributes in the same order as specified in the toolset.
6	REQUIRED_ INPUT	1	CHAR	Is this a required import view? Y (Yes) N (No)
7	ENTITY_VIEW_ ID	4	INTEGER	Parent entity view
8	ATTRIBUTE_ID	4	INTEGER	Attribute included in this entity view

ATTRIBUTE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ATTRIBUTE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Attribute name
6	DSD_NAME	32	VARCHAR	Data Structure Illustration name
7	SEQ	2	SMALLINT	Sequence within entity type or subtype. Use ORDER_BY_SEQ to return attributes in the same order as specified in the toolset.
8	OPT	1	CHAR	Optionality: M (Mandatory) O (Optional)
9	TYPE	1	CHAR	Type of attribute: B (Basic) D (Derived) S (Designed) A (Auto Number)
10	DOMAIN	1	CHAR	Domain of attribute: T (Text) D (Date) M (Time) N (Number) Q (Timestamp) G (Graphic [DBCS]) Y (Text | Mixed) (for system-supplied functions only) Z (Mixed) B (BLOB)

Col	Column Name	Len	Coltype	Description
11	VARYING_LENGTH	1	CHAR	Varying length string? Y (Yes) N (No)
12	LENGTH	4	INTEGER	Number of characters
13	DEC_PLACES	4	INTEGER	Number of decimal places.
14	CASE_SENSITIVE	4	CHAR	Case sensitive? Y (Yes) N (No)
15	UNITS	1	CHAR	Units (for BLOB only): ■ K (KB) ■ M (MB) ■ G (GB) ■ Space (Bytes)
16	PARENT_ENTITY_ID	4	INTEGER	Entity type or subtype for which this is an attribute. Join with: ENTITY_TYPE ENTITY_SUBTYP SYS_ENT_TYPE
17	ACTION_BLOCK_ID	4	INTEGER	ID of algorithm action block for an attribute. (zero if algorithm does not exist).
18	DEF_PERM_VAL_ID	4	INTEGER	ID of default permitted value. Join with PERMIT_VALUE. (zero if default value does not exist)
19	ENCAP_LEVEL	1	CHAR	Level of encapsulation: space (Public) P (Protected)

BAA_ACTN_BLK

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BAA_ACTN_BLK
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Action block name
6	INTEXT	1	CHAR	■ I (Internal) ■ E (External)
7	BUSINESS_SYS_ ID	4	INTEGER	Business system that contains this action block. (Zero if not contained by a business system)

BSD_ACTN_BLK

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BSD_ACTN_BLK
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Action block name
6	INTEXT	1	CHAR	I (Internal) E (External)
7	BUSINESS_SYS_ID	4	INTEGER	Business System that contains this action block

BUS_AREA

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BUS_AREA
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Business area name

BUS_GOAL

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BUS_GOAL
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the business goal
6	SEQ	2	SMALLINT	Sequence within business goal. Use ORDER_BY_SEQ to return business goals in the same order as specified in the toolset.
7	AMOUNT	4	INTEGER	Goal amount.
8	PRIORITY	4	INTEGER	Priority The possible range is 0 (Low) to 9 (High)
9	TARGET_DATE	4	INTEGER	Target date of goal. The format is *YYYYMMDD*
10	UNIT_OF_ MEASURE	32	VARCHAR	Goal amount, unit of measure

Col	Column Name	Len	Coltype	Description
11	PARENT_GOAL_ID	4	INTEGER	ID of the parent business goal. Join with BUS_GOAL. Zero for root business goal (will be used in future release).

BUS_LOCATION

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BUS_LOCATION
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the Business Location
6	TYPE	1	CHAR	Location type: ■ S (Site) ■ T (Type)

BUS_OBJECTIVE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BUS_OBJECTIVE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the Business objective.

Col	Column Name	Len	Coltype	Description
6	SEQ	2	SMALLINT	Sequence within business objective. Use ORDER_BY_SEQ to return business objectives in the same order as specified in the toolset.
7	PRIORITY	4	INTEGER	Priority The possible range is 0 for low, to 9 for high.
8	PARENT_BUSOBJ_ ID	4	INTEGER	ID of the parent business objective. Join with BUS_OBJECTIVE. Zero for root business objective (will be used in future release).

BUS_PROC_SCOPE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BUS_PROC_SCOPE
3	BUSINESS_PROC_ ID	4	INTEGER	ID of business procedure this implements
4	ELEM_PROCESS_ID	4	INTEGER	ID of elementary process implemented. Join with PROCESS_DEF.

BUS_PROC_STEP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BUS_PROC_STEP
3	ID	4	INTEGER	Unique identifier

Col	Column Name	Len	Coltype	Description
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the procedure step
6	DSPLY_1ST	1	CHAR	Display first? Y (Yes) N (No)
7	NON_SCREENED	1	CHAR	Non-screened? S (Screened) N (Non-screened)
8	STEP_TYPE	1	CHAR	Procedure step type: O (Online) B (Batch)
9	BUS_PROCEDURE_ ID	4	INTEGER	Procedure that contains the procedure step. Join with BUSINESS_PROC.
10	SEQ	2	SMALLINT	Sequence of steps within a procedure
11	ACTION_BLOCK_ID	4	INTEGER	ID of the action block that defines the procedure step.

BUS_SYS_SCOPE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BUS_SYS_SCOPE
3	BUSINESS_SYS_ID	4	INTEGER	ID of business system this scopes
4	ELEM_PROCESS_ ID	4	INTEGER	ID of elementary process implemented. Join with PROCESS_DEF.

BUSINESS_PROC

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BUSINESS_PROC
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	The name of the procedure
6	PROCEDURE_TYPE	1	CHAR	Procedure type: ■ O (Online) ■ B (Batch)
7	BUSINESS_SYS_ID	4	INTEGER	The business system that contains the procedure
8	SEQ	2	SMALLINT	Sequence of procedures within business system

BUSINESS_SYS

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	BUSINESS_SYS
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the business system
6	SHOW_EX_ITEMS	1	CHAR	Show extraordinary items first? Y (Yes) N (No)
7	WHAT_PAGE_RTN	1	CHAR	Show what page on return? T (Top) L (Last)

CD_ACTN_BLK

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	CD_ACTN_BLK
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Cascade Delete (CD) action block name
6	ENTITY_ID	4	INTEGER	ID of the entity for which the CD action block is the cascade delete routine. Zero if the cd action block is not a cascade delete routine for an entity type.
7	LINKAGE_ID	4	INTEGER	ID of the linkage for which that the cd action block is the cascade delete routine. Zero if the cd action block is not a cascade delete routine for a linkage.

CELL_VALUE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	CELL_VALUE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	CELL_VALUE	1	CHAR	Cell value
6	MATRIX_ID	4	INTEGER	ID of the matrix that uses this cell value

Col	Column Name	Len	Coltype	Description
7	X_OBJECT_ID	4	INTEGER	ID of the object that uses this cell value on the X axis.
8	Y_OBJECT_ID	4	INTEGER	ID of the object that uses this cell value on the Y axis.

CHANGE_OCCUR

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	CHANGE_OCCUR
3	TYPE	1	CHAR	Type of change: ■ D (Direct) ■ I (Indirect) ■ C (Changed by migration)
4	SESSION_ID	4	INTEGER	ID of session
5	OBJECT_ID	4	INTEGER	ID of object

CLASSIFIER

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	CLASSIFIER
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARTITIONING_ID	4	INTEGER	ID of the partition for which this is a classifier
6	ATTRIBUTE_ID	4	INTEGER	Classifying attribute ID

CMD_SYNONYM

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID		INTEGER	ID of the containing model
2	TBNAME	16	CHAR	CMD_SYNONYM
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	CMD_ SYNONYM	32	VARCHAR	Synonym of the command
6	COMMAND_ID	4	INTEGER	ID of the command for which this is a synonym

COMMAND

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	COMMAND
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the command
6	BUSINESS_SYS_ID	4	INTEGER	ID of the business system that contains the command

COMPONENT_IMPLEM

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model

Col	Column Name	Len	Coltype	Description
2	TBNAME	16	CHAR	COMPONENT_IMPLEM
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Interface type name
6	TYPE	1	CHAR	Type of entity: ■ B (Persistent) ■ D (Transient)
7	NO_INSTANCE	1	CHAR	No instances? ■ space (Instances supported) ■ Y (No instances)
8	CATEGORY	1	CHAR	Category: space (Generic) ■ B (Business object type) ■ T (Task object type)

COMPONENT_SPEC

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	COMPONENT_SPEC
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Interface type name
6	TYPE	1	CHAR	Type of entity: ■ B (Persistent) ■ D (Transient)
7	NO_INSTANCE	1	CHAR	No instances? space (Instances supported) Y (No instances)

Col	Column Name	Len	Coltype	Description
8	CATEGORY	1	CHAR	Category: space (Generic) ■ B (Business object type) ■ T (Task object type)
9	VIEWABLE	1	CHAR	Viewable? ■ space (No) ■ Y (Yes)
10	ENCAP_LEVEL	1	CHAR	Level of encapsulation: ■ space | O (Open) ■ R (Restricted) ■ E (Encapsulated)
11	SUBJECT_AREA_ID	4	INTEGER	ID of the subject area that contains this entity type.

CRITICAL_SUCCESS

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	CRITICAL_SUCCESS
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the critical success factor
6	SEQ	2	SMALLINT	Sequence with critical success factor. Use ORDER_BY_SEQ to return cell values in the same order as specified in the toolset.
7	INHIBITOR	1	CHAR	Inhibitor? ■ Y (Yes) ■ N (No)

Col	Column Name	Len	Coltype	Description
8	PRIORITY	4	INTEGER	Priority The possible range is 0, for low to 9, for high.
9	PARENT_CSF_ID	4	INTEGER	ID of the parent critical success factor. Join with CRITICAL_SUCCESS. Zero for root critical success, will be used in a future release.

CSTM_EDT_PTRN

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	CSTM_EDT_PTRN
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	CLASS	1	CHAR	Class of pattern: ■ T (Text) ■ D (Date) ■ M (Time) ■ N (Number)
6	SCRN_FLD_VAR_ID	4	INTEGER	Screen variable that uses the edit pattern
7	TEXT	2000	LONGVAR	Text of the edit pattern

CURRENT_DATA

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model

Col	Column Name	Len	Coltype	Description
2	TBNAME	16	CHAR	CURRENT_DATA
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the current database or data store
6	SEQ	2	SMALLINT	Sequence within current database or store. Use ORDER_BY_SEQ to return current databases or stores in the same order as specified in the toolset.
7	PARENT_DATA_ID	4	INTEGER	ID of the parent current database or data store. Join with CURRENT_DATA. Zero for root current database or data store (will be used in a future release).

CURRENT_EFFECT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	CURRENT_EFFECT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	CREATE	1	CHAR	Create expected
6	READ	1	CHAR	Read expected
7	UPDT	1	CHAR	Update expected
8	DLET	1	CHAR	Delete expected
9	MATRIX_ID	4	INTEGER	ID of the matrix that uses this current effect.

Col	Column Name	Len	Coltype	Description
10	CUR_DATA_ID	4	INTEGER	ID of current database or data store. Join with CURRENT_DATA.
11	CUR_INFO_SYS_ ID	4	INTEGER	ID of the current information system. Join with CURRENT_INFO_SYS.

CURRENT_INFO_SYS

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	CURRENT_INFO_SYS
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the current information system
6	SEQ	2	SMALLINT	Sequence within current information system. Use ORDER_BY_SEQ to return information systems in the same order as specified in the toolset.
7	STATUS	1	CHAR	System Status: ■ C (Current) ■ P (Planned)
8	PARENT_SYS_ID	4	INTEGER	ID of the parent current information system. Join with CURRENT_INFO_SYS. Zero for root current information system (will be used in future release).

DASD_VOL_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DASD_VOL_USAGE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	DATASET_ID	4	INTEGER	ID of the data set. Join with: ■ DATASET_INDEX ■ DATASET_TBLSP
6	DASD_VOLUME_ ID	4	INTEGER	ID of the DASD volume

DASD_VOLUME

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DASD_VOLUME
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	TYPE	1	CHAR	Type of DASD device: ■ 8 (3380) ■ 5 (3350)
6	VOLSER	32	VARCHAR	Volume serial number

DATA_BASE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATA_BASE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the database
6	DBMS	1	CHAR	Database management system: ■ 2 (DB2 z/OS) ■ A (Datacom) ■ B (ODBC/ADO.NET) ■ C (MSSQL) ■ E (DB2 UDB) ■ H (NONE) ■ J (JDBC) ■ O (Oracle)
7	TYPE	1	CHAR	Type of database: ■ P (Physical) ■ L (Logical)
8	DFLT_ BUFFERPOOL	1	CHAR	Default bufferpool
9	DEF_STG_GRP_ ID	4	INTEGER	ID of storage group that is used as default for this database. Join with STORAGE_GROUP. Zero if default storage group does not exist.

DATA_BASE_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATA_BASE_USAGE
3	TECH_DESIGN_ ID	4	INTEGER	ID of the technical design that uses this database. Join with TECHNICAL_DESIGN.
4	DATABASE_ID	4	INTEGER	ID of the database used by this technical design. Join with DATA_BASE.

DATA_CLUSTER

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATA_CLUSTER
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the data cluster

DATA_STORE_INDEX

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATA_STORE_INDEX
3	ID	4	INTEGER	Unique identifier

Col	Column Name	Len	Coltype	Description
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	8	CHAR	Name of the data store
6	DB2_BUFFERPOOL	1	CHAR	Buffer pool assignment: ■ 0 (BP0) ■ 1 (BP1) ■ 2 (BP2) ■ K (BP32K)
7	VSAM_CAT_NAME	8	CHAR	Name of the VSAM catalog
8	DB2_CLOSE	1	CHAR	Close when not in use? ■ Y (Yes) ■ N (No)
9	DB2_SUBPAGES	1	CHAR	Number of subpages per page, 1, 2, 4, 8, S - 16
10	ERASE_OLD_ DATA	1	CHAR	Erase old data during Create? ■ Y (Yes) ■ N (No)
11	UNIT_ALLOC	1	CHAR	Unit for allocation quantity: ■ C (CYL) ■ T (Trk) ■ R (Rec) ■ K (Kbytes)
12	USE_DEFAULT_ PROP	1	CHAR	Use default data set properties? ■ Y (Yes) ■ N (No)
13	FREE_PAGE_FREQ	4	INTEGER	Free page frequency
14	FREE_SPACE_PCT	4	INTEGER	Free space percentage
15	PRIME_ALLOC	4	INTEGER	Primary allocation quantity
16	SECOND_ALLOC	4	INTEGER	Secondary allocation quantity
17	DATABASE_ID	4	INTEGER	ID of the database. Join with DATA_BASE.

Col	Column Name	Len	Coltype	Description
18	STORAGE_GRP_ID	4	INTEGER	ID of the storage group that contains the indexspace. Join with STORAGE_GROUP. Zero if the indexspace is not contained by a storage group.

DATA_STORE_TBLSP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATA_STORE_TBLSP
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	8	CHAR	Name of the data store
6	DB2_BUFFERPOOL	1	CHAR	Buffer pool assignment: ■ 0 (BP0) ■ 1 (BP1) ■ 2 (BP2) ■ K (BP32K)
7	VSAM_CAT_NAME	8	CHAR	Name of the VSAM catalog
8	DB2_CLOSE	1	CHAR	Close when not in use? ■ Y (Yes) ■ N (No)
9	DB2_LOCKSIZE	1	CHAR	Lock size: ■ P (Page) ■ A (Any) ■ T (Table)
10	ERASE_OLD_DATA	1	CHAR	Erase old data during Create? ■ Y (Yes) ■ N (No)

Col	Column Name	Len	Coltype	Description
11	UNIT_ALLOC	1	CHAR	Unit for allocation quantity: ■ C (Cyl) ■ T (Trk) ■ R (Rec) ■ K (Kbytes)
12	USE_DEFAULT_ PROP	1	CHAR	Use default data set properties: ■ Y (Yes) ■ N (No)
13	SEGMENT_SIZE	4	INTEGER	Segment size
14	FREE_PAGE_FREQ	4	INTEGER	Free page frequency
15	FREE_SPACE_PCT	4	INTEGER	Free space percentage
16	PRIME_ALLOC	4	INTEGER	Primary allocation quantity
17	SECOND_ALLOC	4	INTEGER	Secondary allocation quantity
18	DATABASE_ID	4	INTEGER	ID of the database. Join with DATA_BASE.
19	STORAGE_GRP_ID	4	INTEGER	ID of the storage group that contains this tablespace. Join with STORAGE_GROUP. Zero if the tablespace is not contained by a storage group.

DATACOM_COLUMN

The following table details various column names along with their descriptions.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	ID	4	INTEGER	ID of Datacom column
3	TBNAME	16	CHAR	DATACOM_COLUMN
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name

Col	Column Name	Len	Coltype	Description
6	FORMAT	1	CHAR	Format of data
7	LENGTH	4	INTEGER	Length of field
8	NUM_DEC_PLC	4	INTEGER	Number of decimal places
9	OPTIONALITY	1	CHAR	Optionality for DBMS fields
10	DFTL_VALUE_SPEC	1	CHAR	Default value specified
11	FIELD_ID	1	INTEGER	Base column this object is extended from. Join with FIELD.

DATACOM_CONSTRNT

The following table details various column names along with their descriptions.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATACOM_CONSTRNT
3	ID	4	INTEGER	ID of Datacom constraint
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name
6	RI_ENF_PROD	1	CHAR	RI enforcement for production
7	RI_ENF_TEST	1	CHAR	RI enforcement for test
8	RI_ENF_TRIGGER	1	CHAR	RI enforced trigger
9	LINKAGE_ID	4	INTEGER	Base linkage this object is extended from. Join with LINKAGE.

DATACOM_DATABASE

The following table details various column names along with their descriptions.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATACOM_DATABASE
3	ID	4	INTEGER	ID of Datacom database
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name
6	DATABASE_ID	4	INTEGER	Base database this object is extended from. Join with DATA_BASE.

DATACOM _INDEX

The following table details various column names along with their descriptions.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATACOM_TABLE
3	ID	4	INTEGER	ID of Datacom index
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name
6	ENTRY_POINT_ID	4	INTEGER	Base index this object is extended from. Join with ENTRY_POINT.

DATACOM _TABLE

The following table details various column names along with their descriptions.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATACOM_TABLE
3	ID	4	INTEGER	ID of Datacom table
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name
6	RI_TRIG_PRFX	32	CHAR	RI trigger and procedure name prefix
7	OWNER_ID_EXT	8	CHAR	Owner identifier extension
8	PV_READ_ENF	1	CHAR	Permitted value READ enforcement
9	PV_DBMS_ENF	1	CHAR	Permitted value DBMS enforcement
10	RECORD_ID	4	INTEGER	Base record this object is extended from. Join with RECORD.

DATACOM _TD

The following table details various column names along with their descriptions.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATACOM_TD
3	ID	4	INTEGER	ID of Datacom technical design
4	ORG_ID	4	INTEGER	Original object identifier
5	OWNER_ID	8	CHAR	Owner identifier

Col	Column Name	Len	Coltype	Description
6	DFLT_DBNAME	32	CHAR	Default DB name
7	RI_ENFRCMNT	1	CHAR	Referential integrity enforcement
8	RES_WORD_CHK	1	CHAR	Reserved work check?
9	DBMS_REL_VER	1	CHAR	DBMS release version
10	PV_READ_ENF	1	CHAR	Permitted value READ enforcement
11	PV_DBMS_ENF	1	CHAR	Permitted value DBMS enforcement
12	DEF_VALUE_ENF	1	CHAR	Default value enforced?
13	NULL_OR_DEF	1	CHAR	Create NULL or default value?
14	GEN_ODBC_INT	1	CHAR	Generate ODBC interface?
15	TECH_DSN_ID	4	INTEGER	Base technical design this object is extended from. Join with TECHNICAL_DESIGN.

DATASET_INDEX

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATASET_INDEX
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	VSAM_CAT_NAME	8	CHAR	Name of the VSAM catalog
6	FREE_PAGE_FREQ	4	INTEGER	Free page frequency
7	FREE_SPACE_PCT	4	INTEGER	Free space percentage
8	PRIME_ALLOC	4	INTEGER	Primary allocation quantity
9	SECOND_ALLOC	4	INTEGER	Secondary allocation quantity

Col	Column Name	Len	Coltype	Description
10	UNIT_ALLOC	1	CHAR	Unit for allocation quantity: ■ C (Cyl) ■ T (Trk) ■ R (Rec) ■ K (Kbytes)
11	ERASE_OLD_DATA	1	CHAR	Erase old data during Create? ■ Y (Yes) ■ N (No)
12	USE_DEFAULT_ PROP	1	CHAR	Use default data set properties? ■ Y (Yes) ■ N (No)
13	PART_NUMBER	4	INTEGER	Data set number for DB2 partitioning
14	SEQ	2	SMALLINT	Sequence within data store
15	DATA_STORE_ID	4	INTEGER	ID of the containing data store. Join with DATA_STORE_INDEX.
16	STORAGE_GRP_ID	4	INTEGER	ID of the storage group that holds this indexspace data set. Join with STORAGE_GROUP. Zero if the data set is not held by a storage group.

DATASET_TBLSP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DATASET_TBLSP
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	VSAM_CAT_NAME	8	CHAR	Name of the VSAM catalog

Col	Column Name	Len	Coltype	Description
6	FREE_PAGE_FREQ	4	INTEGER	Free page frequency
7	FREE_SPACE_PCT	4	INTEGER	Free space percentage
8	PRIME_ALLOC	4	INTEGER	Primary allocation quantity
9	SECOND_ALLOC	4	INTEGER	Secondary allocation quantity
10	UNIT_ALLOC	1	CHAR	Unit for allocation quantity: ■ C (Cyl) ■ T (Trk) ■ R (Rec) ■ K (Kbytes)
11	ERASE_OLD_DATA	1	CHAR	Erase old data during Create? ■ Y (Yes) ■ N (No)
12	USE_DEFAULT_ PROP	1	CHAR	Use default data set properties: ■ Y (Yes) ■ N (No)
13	PART_NUMBER	4	INTEGER	Data set number for DB2 partitioning
14	SEQ	2	SMALLINT	Sequence within data store
15	DATA_STORE_ID	4	INTEGER	ID of the containing data store. Join with DATA_STORE_TBLSP.
16	STORAGE_GRP_ID	4	INTEGER	ID of the storage group that holds this tablespace index. Join with STORAGE_GROUP. Zero if data set is not held by a storage group.

DB2_DDL_DB

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DB2_DDL_DB

Col	Column Name	Len	Coltype	Description
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	MEMBER_NAME	8	CHAR	Name of member containing generated DDL for DB2 database definition
6	DATE_GENERATED	4	INTEGER	Generation date
7	TIME_GENERATED	4	INTEGER	Generation time
8	GEN_BY_USER	8	CHAR	Generated by user ID
9	DATA_BASE_ID	4	INTEGER	ID of the database with which this DB2 database definition is built. Join with DATA_BASE.

DB2_DDL_INDEX

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DB2_DDL_INDEX
3	ID	4	CHAR	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	MEMBER_NAME	8	CHAR	Name of member containing generated DDL for DB2 index definition
6	DATE_GENERATED	4	INTEGER	Generation date
7	TIME_GENERATED	4	INTEGER	Generation time
8	GEN_BY_USER	8	CHAR	Generated by user ID
9	ENTRY_POINT_ID	4	INTEGER	ID of the entry point which this DB2 index definition is built from. Join with ENTRY_POINT.

DB2_DDL_TABLE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DB2_DDL_TABLE
3	ID	4	CHAR	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	MEMBER_NAME	8	CHAR	Name of member containing generated DDL for DB2 table definition
6	DATE_GENERATED	4	INTEGER	Generation date
7	TIME_GENERATED	4	INTEGER	Generation time
8	GEN_BY_USER	8	CHAR	Generated by user ID
9	RECORD_ID	4	INTEGER	ID of the record that is used to build this DB2 table definition. Join with RECORD.

DB2_DDL_TBLSP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DB2_DDL_TBLSP
3	ID	4	CHAR	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	MEMBER_NAME	8	CHAR	Name of member containing generated DDL for DB2 tablespace definition
6	DATE_GENERATED	4	INTEGER	Generation date
7	TIME_GENERATED	4	INTEGER	Generation time
8	GEN_BY_USER	8	CHAR	Generated by user ID

Col	Column Name	Len	Coltype	Description
9	DATA_STR_TBSP_ ID	4	INTEGER	ID of the data storage tablespace with which this DB2 tablespace is built. Join with DATA_STORE_TBLSP.

DB2_DEF_CLUSTER

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DB2_DEF_CLUSTER
3	ID	4	CHAR	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	MEMBER_NAME	8	CHAR	Name of member containing DB2 VSAM define cluster for data store
6	DATE_ GENERATED	4	INTEGER	Generation date
7	TIME_ GENERATED	4	INTEGER	Generation time
8	GEN_BY_USER	8	CHAR	Generated by user ID
9	DATA_SET_ID	4	INTEGER	ID of the data set that is implemented by this DB2 VSAM define cluster. Join with: ■ DATASET_INDEX ■ DATASET_TBLSP

DB2_MVS_COLUMN

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of DB2 column
2	MODEL_ID	4	INTEGER	ID of the containing model
3	TBNAME	16	CHAR	DB2_MVS_COLUMN
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name
6	FORMAT	1	CHAR	Format of data
7	LENGTH	4	INTEGER	Length of field
8	NUM_DEC_PLC	4	INTEGER	Number of decimal places
9	OPTIONALITY	1	CHAR	Optionality for DBMS fields
10	DFLT_VALUE_ SPEC	1	CHAR	Default value specified
11	FIELD_ID	1	INTEGER	Base column this object is extended from. Join with FIELD.

DB2_MVS_CONSTRNT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of DB2 constraint
2	MODEL_ID	4	INTEGER	ID of the containing model
3	TBNAME	16	CHAR	DB2_MVS_CONSTRNT
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name
6	RI_ENF_PROD	1	CHAR	RI enforcement for production
7	RI_ENF_TEST	1	CHAR	RI enforcement for test
8	RI_ENF_TRIGGER	1	CHAR	RI enforced trigger

Col	Column Name	Len	Coltype	Description
9	LINKAGE_ID	4	INTEGER	Base linkage this object is extended from. Join with LINKAGE.

DB2_MVS_DATABASE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of DB2 constraint
2	MODEL_ID	4	INTEGER	ID of the containing model
3	TBNAME	16	CHAR	DB2_MVS_DATABASE
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name
6	OWNER_IDENTIFIER	8	CHAR	Owner identifier
7	ROSHARE	1	CHAR	None/Owner/Read availability
8	DB_ALLOWS_ TABLES	1	CHAR	Database allows tables in system
9	BUFFERPOOL_NUM	4	INTEGER	Bufferpool number
10	DATABASE_ID	4	INTEGER	Base database this object is extended from. Join with DATA_BASE.

DB2_MVS_INDEX

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of DB2 index
2	MODEL_ID	4	INTEGER	ID of the containing model
3	TBNAME	16	CHAR	DB2_MVS_INDEX
4	ORG_ID	4	INTEGER	Original object identifier

Col	Column Name	Len	Coltype	Description
5	NAME	32	CHAR	Name
6	ENTRY_POINT_ ID	4	INTEGER	Base index this object is extended from. Join with ENTRY_POINT.

DB2_MVS_INDEXSPC

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of DB2 indexspace
2	MODEL_ID	4	INTEGER	ID of the containing model
3	TBNAME	16	CHAR	DB2_MVS_INDEXSPC
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name
6	DEFER_BUILD	1	CHAR	Defer index build?
7	INDEX_TYPE	1	CHAR	Index type
8	NOT_NULL_UNIQ	1	CHAR	Unique where not null?
9	BUFFERPOOL_NUM	4	INTEGER	Bufferpool number
10	INDEXSPACE_ID	4	INTEGER	Base indexspace this object is extended from. Join with DATA_STORE_INDEX.

DB2_MVS_TABLE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of DB2 tabel
2	MODEL_ID	4	INTEGER	ID of the containing model
3	TBNAME	16	CHAR	DB2_MVS_TABLE
4	ORG_ID	4	INTEGER	Original object identifier

Col	Column Name	Len	Coltype	Description
5	NAME	32	CHAR	Name
6	RI_TRIG_PRFX	32	CHAR	RI trigger and procedure name prefix
7	OWNER_ID_EXT	8	CHAR	Owner identifier extension
8	PV_READ_ENF	1	CHAR	Permitted value READ enforcement
9	PV_DBMS_ENF	1	CHAR	Permitted value DBMS enforcement
10	RSTR_DROP	1	CHAR	With restrict on drop?
11	DATA_CAPTURE	1	CHAR	Data capture?
12	RECORD_ID	4	INTEGER	Base record this object is extended from. Join with RECORD.

DB2_MVS_TABLESPC

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of DB2 tablespace
2	MODEL_ID	4	INTEGER	ID of the containing model
3	TBNAME	16	CHAR	DB2_MVS_TABLESPC
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Name
6	COMPRESS	1	CHAR	Compress?
7	LOCKMAX_SYS	1	CHAR	Lockmax by system?
8	LOCKMAX_VALUE	4	INTEGER	Lockmax value
9	SYSTEM_ TABLESPC	1	CHAR	System tablespace?
10	BUFFERPOOL_ NUM	4	INTEGER	Bufferpool number

Col	Column Name	Len	Coltype	Description
11	TABLESPACE_ID	4	INTEGER	Base tablespace this object is extended from. Join with DATA_STORE_TBLSP.

DB2_MVS_TD

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of DB2 technical design
2	MODEL_ID	4	INTEGER	ID of the containing model
3	TBNAME	16	CHAR	DB2_MVS_TD
4	ORG_ID	4	INTEGER	Original object identifier
5	OWNER_ID	8	CHAR	Owner identifier
6	DFLT_DBNAME	32	CHAR	Default DB name
7	RI_ ENFORCEMENT	1	CHAR	Referential integrity enforcement
8	RES_WORD_CHK	1	CHAR	Reserved work check?
9	DBMS_REL_VER	1	CHAR	DBMS release version
10	PV_READ_ENF	1	CHAR	Permitted value READ enforcement
11	PV_DBMS_ENF	1	CHAR	Permitted value DBMS enforcement
12	DEF_VALUE_ENF	1	CHAR	Default value enforced?
13	NULL_OR_DEF	1	CHAR	Create NULL or default value?
14	GEN_ODBC_INT	1	CHAR	Generate ODBC interface?
15	AVAILABILITY	1	CHAR	None/Owner/Read availability
16	COMPRESS_TBS	1	CHAR	Compress tablespace?
17	LOCKMAX_VAL	4	INTEGER	Lockmax value
18	LOCKMAX_SYS	1	CHAR	Lockmax by systerm?
19	DATA_CAPTURE	1	CHAR	Data capture?
20	RESTRICT_DROP	1	CHAR	With restrict on drop?

Col	Column Name	Len	Coltype	Description
21	DEFER_IDX_BLD	1	CHAR	Defer index build?
22	INDEX_TYPE	1	CHAR	Index type
23	QUALIFY_TABLE	1	CHAR	Qualify table references?
24	ASIS_FLAG	1	CHAR	ASIS naming flag on TD properties
25	ALLOWS_TABLES	1	CHAR	Database allows tables in system
26	BUFFERPOOL_DB	4	INTEGER	Bufferpool number database default
27	BUFFERPOOL_IN	4	INTEGER	Bufferpool number indexspace default
28	BUFFERPOOL_TB	4	INTEGER	Bufferpool number tablespace default
29	REC_DEF_TSPC	8	CHAR	Record default tablespace
30	TECH_DSN_ID	4	INTEGER	Base technical design this object is extended from. Join with TECHNICAL_DESIGN.

DB2_RESRC_MODULE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DB2_RESRC_MODULE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	MEMBER_NAME	8	CHAR	Member name
6	DATE_ GENERATED	4	INTEGER	Generation date. Not used.
7	TIME_ GENERATED	4	INTEGER	Generation time. Not used.
8	GEN_BY_USER	8	CHAR	Generated by user ID. Not used.

Col	Column Name	Len	Coltype	Description
9	IMPL_LOGIC_ID	4	INTEGER	ID of the implementation logic that uses the DB2 resource module

DERIVATION_ALGOR

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DERIVATION_ALGOR
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Action block name
6	INTEXT	1	CHAR	Internal or external action block? ■ I (Internal) ■ E (External)
7	BUSINESS_SYS_ID	4	INTEGER	Business system that contains this action block. Zero if not contained by a business system.

DESC

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of object for which this is a description
2	TEXT	2000	LONGVAR	Text of description

DFLT_EDT_PTRN

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DFLT_EDT_PTRN
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the default edit pattern
6	CLASS	1	CHAR	Class of pattern: ■ T (Text) ■ N (Num) ■ D (Date) ■ M (Time) ■ G (DBCS) ■ Q (Timestamp)
7	BUSINESS_SYS_ ID	4	INTEGER	ID of the business system that contains this edit pattern
8	TEXT	2000	LONGVAR	Text of the edit pattern

DFLT_LIT_VDAT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DFLT_LIT_VDAT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	COLOR	1	CHAR	Default color for literal fields
6	HIGHLIGHT	1	CHAR	Default highlighting for literal fields

Col	Column Name	Len	Coltype	Description
7	INTENSITY	1	CHAR	Default intensity for literal fields
8	BUSINESS_ SYS_ID	4	INTEGER	Business system for which the defaults apply.

DFLT_PRM_VDAT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DFLT_PRM_VDAT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	COLOR	1	CHAR	Default color for prompt fields
6	HIGHLIGHT	1	CHAR	Default highlighting for prompt fields
7	INTENSITY	1	CHAR	Default intensity for prompt fields
8	BUSINESS_SYS_ ID	4	INTEGER	Business system for which the defaults apply.

DFLT_VAR_VDAT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DFLT_VAR_VDAT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	COLOR	1	CHAR	Default color for variable fields

Col	Column Name	Len	Coltype	Description
6	HIGHLIGHT	1	CHAR	Default highlighting for variable fields
7	INTENSITY	1	CHAR	Default intensify for variable fields
8	PROTECTION	1	CHAR	Default protection for variable fields
9	JUSTIFICATION	1	CHAR	Default justification for variable fields
10	FILL_CHAR	1	CHAR	Default fill character for variable fields
11	VIDEO_CURSOR	1	CHAR	Insert cursor here? ■ Y (Yes) ■ N (No)
12	BLANK_WHEN_ ZERO	1	CHAR	Blank field when zero? ■ Y (Yes) ■ N (No)
13	BUSINESS_SYS_ ID	4	INTEGER	Business system for which the defaults apply

DFLT_VARE_VDAT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DFLT_VARE_VDAT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	COLOR	1	CHAR	Default color for variable fields
6	HIGHLIGHT	1	CHAR	Default highlighting for variable fields
7	INTENSITY	1	CHAR	Default intensify for variable fields

Col	Column Name	Len	Coltype	Description
8	PROTECTION	1	CHAR	Default protection for variable fields
9	VIDEO_CURSOR	1	CHAR	Insert cursor here? ■ Y (Yes) ■ N (No)
10	BUSINESS_SYS_ID	4	INTEGER	Business system for which the defaults apply.

DIALECT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	Dialect
3	ID	4	INTEGER	Unique Identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	8	CHAR	Dialect name
6	DEFAULT	1	CHAR	Is the dialect the default dialect? ■ Y (Yes) ■ N (No)
7	DEC_PT_CHAR	1	CHAR	Decimal point character
8	SEPARATOR_ CHAR	1	CHAR	Separator character
9	TRANSLAT_ TABLE1	8	CHAR	Translation table name ASCII/EBCDIC
10	TRANSLAT_ TABLE2	8	CHAR	Translation table name lower/upper. Not used.
11	MESSAGE_ TABLE	8	CHAR	Message table name

Col	Column Name	Len	Coltype	Description
12	NATL_LANG_ SUP	1	CHAR	National language support option: ■ Y (Yes) ■ N (No)

DIALECT_TEXT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DIALECT_TEXT
3	ID	4	INTEGER	Unique Identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence number. Use ORDER_BY_SEQ to return in same order as specified in toolset.
6	ACCEL_VALUE	1	CHAR	Accelerator value (for scroll amount only)
7	DIALECT_ID	4	INTEGER	ID of the dialect of the text
8	OBJECT_ID	4	INTEGER	ID of the text for the prompt, screen literal, command, command synonym, edit pattern, message, or scroll amount. Join with: ■ PROMPT ■ SCRN_FLD_LIT ■ COMMAND ■ CMD_SYNONYM ■ DFLT_EDT_PTRN ■ CSTM_EDT_PTRN ■ MESSAGE ■ SCROLL_AMOUNT

Col	Column Name	Len	Coltype	Description
9	TEXT_VALUE	2000	LONGVAR	Text value

DIALOG_FLOW

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DIALOG_FLOW
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	TYPE	1	CHAR	Type of Flow: ■ T (Transfer) ■ L (Link)
6	DSPLY_1ST	1	CHAR	Display first? ■ Y (Yes) ■ N (No)
7	DSPLY_1ST_ RTRN	1	CHAR	Display first on return from link? ■ Y (Yes) ■ N (No)
8	SEND_ COMMAND	1	CHAR	Send current value of command? ■ Y (Yes) ■ N (No)
9	RETURN_ COMMAND	1	CHAR	Return current value of command? ■ Y (Yes) ■ N (No)
10	INITBY_P_ STEP_ID	4	INTEGER	Initiated procedure step. Join with BUS_PROC_STEP.
11	INITS_P_STEP_ ID	4	INTEGER	Initiating procedure step. Join with BUS_PROC_STEP.

DLG_DATA_SENT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DLG_DATA_SENT
3	DIALOG_FLOW_ ID	4	INTEGER	ID of the dialog flow
4	SRC_DATA_ VIEW_ID	4	INTEGER	Source data view sent by the dialog flow. Join with: ■ GROUP_VIEW ■ ENTITY_VIEW
5	DST_DATA_ VIEW_ID	4	INTEGER	Destination data view sent by the dialog flow. Join with: ■ GROUP_VIEW ■ ENTITY_VIEW

DLG_FLWS_EXST

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DLG_FLWS_EXST
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SAVE_INPUT_ VIEWS	1	CHAR	Save input views on link? ■ Y (Yes) ■ N (No)
6	DIALOG_FLOW_ ID	4	INTEGER	ID of the dialog flow

Col	Column Name	Len	Coltype	Description
7	EXIT_STATE_ ID	4	INTEGER	ID of the exit state on which the dialog flows

DLG_SETS_CMD

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	DLG_SETS_CMD
3	DIALOG_FLOW_ ID	4	INTEGER	ID of the dialog flow
4	COMMAND_ID	4	INTEGER	ID of the command set by the dialog flow

ENTITY_ST_TRANS

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ENTITY_ST_TRANS
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SOURCE_ENTITY_ID	4	INTEGER	ID of the source entity
6	DEST_ENTITY_ID	4	INTEGER	ID of the destination entity

ENT_ST_TRANS_USE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ENT_ST_TRANS_USE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ENT_ST_TRANS_ ID	4	INTEGER	ID of the entity state transition
6	ACTIVITY_ID	4	INTEGER	ID of the activity (function or process)

ENTITY_REC_IMPL

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ENTITY_REC_IMPL
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ENTITY_ID	4	INTEGER	ID of the entity type implemented by this record. Join with: ■ ENTITY_TYPE ■ ENTITY_SUBTYP
6	RECORD_ID	4	INTEGER	ID of the record defined by this entity type

ENTITY_SUBTYP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ENTITY_SUBTYP
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Entity subtype name
6	DSD_NAME	32	VARCHAR	Data Structure Illustration name
7	MIN_OCCUR	4	INTEGER	Minimum number of occurrences
8	MAX_OCCUR	4	INTEGER	Maximum number of occurrences
9	AVG_OCCUR	4	INTEGER	Average number of occurrences
10	GROWTH_RATE	4	INTEGER	Growth rate (percent)
11	GROWTH_RATE_PER	1	CHAR	Growth rate period: ■ Y (Year) ■ M (Month) ■ W (Week) ■ D (Day)
12	SEQ	4	INTEGER	Sequence number for presentation
13	PARTITIONING_ID	4	INTEGER	ID of the partitioning on which the subtype appears
14	PERMIT_VALUE_ID	4	INTEGER	ID of classifying permitted value for entity subtypes (zero if classifying permitted value does not exist).

ENTITY_TYPE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ENTITY_TYPE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Entity type name
6	DSD_NAME	32	VARCHAR	Data Structure Illustration name
7	MIN_OCCUR	4	INTEGER	Minimum number of occurrences
8	MAX_OCCUR	4	INTEGER	Maximum number of occurrences
9	AVG_OCCUR	4	INTEGER	Average number of occurrences
10	GROWTH_RATE	4	INTEGER	Growth rate (percent)
11	GROWTH_RATE_PER	1	CHAR	Growth rate period: ■ Y (Year) ■ M (Month) ■ W (Week) ■ D (Day)
12	SUBJECT_AREA_ID	4	INTEGER	ID of subject area that contains this entity type
13	TYPE	1	CHAR	Type of entity: ■ B (Persistent) ■ D (Transient)
14	NO_INSTANCE	1	CHAR	No instances? ■ space (Instances supported) ■ Y (No instances)

Col	Column Name	Len	Coltype	Description
15	CATEGORY	1	CHAR	Category: ■ space (generic) ■ B (Business object type) ■ T (Task object type)
16	VIEWABLE	1	CHAR	Viewable? ■ space (No) ■ Y (Yes)
17	ENCAP_LEVEL	1	CHAR	Level of encapsulation: ■ space | O (Open) ■ R (Restricted) ■ E (Encapsulated)
18	CBD_TYPE	1	CHAR	CBD type: ■ space (Regular entity type) ■ I (Interface type) ■ S (Specification type) ■ P (Component specification) ■ M (Component implementation)

ENTITY_VIEW

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ENTITY_VIEW
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Entity view name

Col	Column Name	Len	Coltype	Description
6	REQUIRED_ INPUT	1	CHAR	Is this a required import view? ■ Y (Yes) ■ N (No)
7	ENT_ACT_ ALLOWED	1	CHAR	Is entity action allowed? ■ Y (Yes) ■ N (No)
8	USED_AS_I_O	1	CHAR	Used as input or output: ■ Y (Yes) ■ N (No)
9	SEQH	4	INTEGER	Sequence within viewset hierarchy. Use ORDER_BY_SEQH to return the views in the same order as specified in toolset.
10	SEQ	2	SMALLINT	Sequence within parent view or view set. Use ORDER_BY_SEQ to return children in the same order as specified in toolset.
11	PARENT_ID	4	INTEGER	Parent group view or view set. Join with: ■ GROUP_VIEW ■ VIEW_SET
12	VIEW_SET_ID	4	INTEGER	Viewset that contains this view. Join with VIEW_SET.
13	ENTITY_ID	4	INTEGER	Entity type or subtype for which this is an entity view. Join with: ■ ENTITY_TYPE ■ ENTITY_SUBTYP ■ SYS_ENT_TYPE

ENTRY_POINT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ENTRY_POINT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Entry point name
6	CLUSTERED	1	CHAR	Is the entry point clustered? ■ Y (Yes) ■ N (No)
7	PARTITIONED	1	CHAR	Is the entry point partitioned? ■ Y (Yes) ■ N (No)
8	ORGANIZATION	1	CHAR	Entry point organization
9	PRIMARY_KEY	1	CHAR	Primary key for table? ■ Y (Yes) ■ N (No)
10	DATA_STR_NDX_ ID	4	INTEGER	ID of the data store index wherethis entry point is stored. Join with DATA_STORE_INDEX.
11	IDENTIFIER_ID	4	INTEGER	ID of the identifier that this entry point implements (zero for entry points not associated with an identifier).
12	REL_IMPL_ID	4	INTEGER	ID of the relationship implementation associated with this entry point (zero for entry points not associated with an implementation). Join with REL_PART_IMPL.

ENVIRONMENT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ENVIRONMENT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the environment

EXIT_STATE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	EXIT_STATE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the exit state
6	TERMINAT_ ACTION	1	CHAR	Termination action: ■ M (Normal) ■ A (Abort) ■ R (Rollback)
7	TYPE	1	CHAR	Message type: ■ I (Informational) ■ W (Warning) ■ E (Error) ■ N (None)
8	BUSINESS_SYS_ ID	4	INTEGER	ID of the business system that contains the exit state (zero if not contained by a business system).

Col	Column Name	Len	Coltype	Description
9	MESSAGE_ID	4	INTEGER	ID of the message for non-default dialect Zero if it is the default dialect.

EXIT_STATE_US

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	EXIT_STATE_US
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SAVE_INPUT_ VIEWS	1	CHAR	Save input views on link? ■ Y (Yes) ■ N (No)
6	EXIT_STATE_ID	4	INTEGER	Exit state referenced by this usage
7	DIALOG_FLOW_ ID	4	INTEGER	Dialog flow that results from this usage
8	COMMAND_ID	4	INTEGER	Command that causes autoflow (zero if command does not cause autoflow).

EXPECT_EFFECT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	EXPECT_EFFECT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier

Col	Column Name	Len	Coltype	Description
5	CREATE	1	CHAR	Activity expected to create entity type? ■ Y (Yes) ■ N (No)
6	READ	1	CHAR	Activity expected to read entity type? ■ Y (Yes) ■ N (No)
7	UPDT	1	CHAR	Activity expected to update entity type? ■ Y (Yes) ■ N (No)
8	DLET	1	CHAR	Activity expected to delete entity type? ■ Y (Yes) ■ N (No)
9	ENTITY_ID	4	INTEGER	Entity type or subtype that this activity is expected to affect. Join with: ■ ENTITY_TYPE ■ ENTITY_SUBTYP
10	ACTIVITY_ID	4	INTEGER	Activity ID (function or process). Join with: ■ FUNCTION_DEF ■ PROCESS_DEF

FACILITY

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	FACILITY

Col	Column Name	Len	Coltype	Description
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the facility
6	TYPE	1	CHAR	Facility type: ■ H (Hardware) ■ S (Software)
7	CUR_CAPACITY	4	INTEGER	Current capacity
8	REQ_CAPACITY	4	INTEGER	Required capacity
9	UNIT_OF_ MEASURE	32	VARCHAR	Unit of measure

FIELD

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	FIELD
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Field name
6	MACRO_NAME	32	VARCHAR	Macro name (column name)
7	SEQ	2	SMALLINT	Sequence within record
8	ROLE	1	CHAR	Role: ■ P (Data) ■ F (Foreign key) ■ D (Denormalized)

Col	Column Name	Len	Coltype	Description
9	FORMAT	1	CHAR	Format: ■ P (Packed) ■ X (Text) ■ D (Date) ■ F (Float) ■ S (Small) ■ I (Integer) ■ V (Varchar) ■ Q (Timestamp) ■ G (Graphic) ■ U (Vargraphic) ■ M (Long Vargraphic) ■ L (Longvarchar) ■ B (BLOB)
10	OCCURS	4	INTEGER	Number of occurrences, if repeating. Not used.
11	LENGTH	4	INTEGER	Length
12	DEC_PLACES	4	INTEGER	Number of decimal places
13	OPT	1	CHAR	DB2 optionality (Optional) M (Mandatory)
14	UNITS	1	CHAR	Units (for BLOB only): ■ K (KB) ■ M (MB) ■ G (GB) ■ Space (Bytes)
15	FIELDPROC_ NAME	8	VARCHAR	Fieldproc routine name
16	RECORD_IS_IN_ ID	4	INTEGER	ID of the record that contains this field. Join with RECORD.
17	ATTRIBUTE_ID	4	INTEGER	ID of the attribute that this field implements

Col	Column Name	Len	Coltype	Description
18	REL_DENORM_ID	4	INTEGER	If this is a denormalized field, ID of the relationship this is denormalized along. Otherwise, zero. Join with RELATIONSHIP.

FLD_ENTPT_VALUE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	FLD_ENTPT_VALUE
3	ID	4	INTEGER	Unique Identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence within field entry point usage. Use ORDER_BY_SEQ to return the values in the same order as specified in the toolset.
6	SEQT	4	INTEGER	Sequence within tablespace data set. Use ORDER_BY_SEQT to return the values in the same order as specified in the toolset.
7	SEQI	4	INTEGER	Sequence within indexspace data set. Use ORDER_BY_SEQI to return the values in the same order as specified in the toolset.
8	FLD_ENTPT_US_ ID	4	INTEGER	ID of the field entry point usage that controls this partition
9	DATASET_TBSP_ ID	4	INTEGER	ID of the tablespace data set that this value determines the partitioning for

Col	Column Name	Len	Coltype	Description
10	DATASET_INDX_ ID	4	INTEGER	ID of the indexspace data set that this value determines the partitioning for
11	VALUE	2000	LONGVAR	Partitioning limit value

FLD_ENTRY_PT_USE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	FLD_ENTRY_PT_USE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQUENCE	1	CHAR	Sequence can be: ■ A (Ascending) ■ D (Descending)
6	SEQ	2	SMALLINT	Sequence within entry point
7	FIELD_ID	4	INTEGER	ID of the field used by this entry point
8	ENTRY_POINT_ ID	4	INTEGER	ID of the entry point defined by this field

FLD_LINK_USE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	FLD_LINK_USE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier

Col	Column Name	Len	Coltype	Description
5	FIELD_FROM_ID	4	INTEGER	ID of the foreign key field in the from record. Join with FIELD.
6	FIELD_TO_ID	4	INTEGER	ID of the target field in the to record. Join with FIELD.
7	LINKAGE_ID	4	INTEGER	ID of the linkage that uses this usage

FUNCTION_DEF

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	FUNCTION_DEF
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Function name

GROUP_VIEW

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	GROUP_VIEW
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Group view name

Col	Column Name	Len	Coltype	Description
6	SEQH	4	INTEGER	Sequence within viewset hierarchy. Use ORDER_BY_SEQH to return the views in the same order as specified in toolset.
7	SEQ	2	SMALLINT	Sequence within parent view or activity. Use ORDER_BY_SEQ to return children in the same order as specified in toolset.
8	CARD	1	CHAR	Cardinality: ■ 1 (One) ■ M (One or many)
9	MIN_CARD	4	INTEGER	Cardinality - at least
10	MAX_CARD	4	INTEGER	Cardinality - at most
11	AVG_CARD	4	INTEGER	Cardinality - on average
12	ABS_EST_MIN	1	CHAR	Absolute or estimated minimum cardinality: ■ E (Estimated) ■ A (Absolute)
13	ABS_EST_MAX	1	CHAR	Absolute or estimated maximum cardinality: ■ E (Estimated) ■ A (Absolute)
14	REQUIRED_ INPUT	1	CHAR	Is this a required import view? ■ Y (Yes) ■ N (No)
15	ENT_ACT_ ALLOWED	1	CHAR	Is entity action allowed? ■ Y (Yes) ■ N (No)
16	USED_AS_I_O	1	CHAR	Used as input or output: ■ Y (Yes) ■ N (No)

Col	Column Name	Len	Coltype	Description
17	PARENT_ID	4	INTEGER	Parent group view or view set. Join with: ■ GROUP_VIEW ■ VIEW_SET
18	VIEW_SET_ID	4	INTEGER	Viewset that contains this view. Join with VIEW_SET.

IDENTIFIER

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	IDENTIFIER
3	ID	4	INTEGER	Identifier (not unique)
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence within entity type or subtype. Use ORDER_BY_SEQ to return identifiers in the same order as specified in the toolset.
6	ROLE	1	CHAR	Role: ■ A (Attribute Identifier) ■ R (Relationship Identifier)
7	NAME	32	VARCHAR	Name of identifier
8	PRIMARY_KEY	1	CHAR	Primary identifier for entity type? ■ Y (Yes) ■ N (No)
9	ENTITY_ID	4	INTEGER	Entity type or subtype for which this attribute or relationship is identifier. Join with: ■ ENTITY_TYPE ■ ENTITY_SUBTYP

Col	Column Name	Len	Coltype	Description
10	ATTR_OR_REL_ ID	4	INTEGER	ID of attribute or relationship

IMPL_LOGIC_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	IMPL_LOGIC_USAGE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	IMPL_LOGIC_ID	4	INTEGER	ID of the implementation logic that calls this implementation logic usage. Join with IMPLEMENT_LOGIC.
6	CALLED_IMPL_ ID	4	INTEGER	ID of the implementation logic that is called by this implementation logic usage. Join with IMPLEMENT_LOGIC.

IMPLEMENT_LOGIC

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	IMPLEMENT_LOGIC
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	MEMBER_NAME	8	CHAR	Member name
6	SOURCE_ LANGUAGE	32	VARCHAR	Name of source language

Col	Column Name	Len	Coltype	Description
7	DATE_ GENERATED	4	CHAR	Generation date
8	TIME_ GENERATED	4	CHAR	Time generated
9	GEN_BY_USER	8	CHAR	Generated by user ID
10	DATE_COMPILED	4	INTEGER	Date compiled
11	TIME_COMPILED	4	INTEGER	Time compiled
12	COMPILED_BY_ USER	8	CHAR	Compiled by user ID. Not used.
13	OK_SESSION_ID	4	INTEGER	Session ID at which checked OK.
14	ACTION_BLOCK_ ID	4	INTEGER	ID of the action block that is implemented by the implementation logic. Join with: ■ ACTION_BLOCK ■ CD_ACTN_BLK

INFORMATION_NEED

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	INFORMATION_NEED
3	ID	4	INTEGER	Identifier (not unique)
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the information need
6	PRIORITY	4	INTEGER	Priority The range of possible values is 0 for low, to 9 for high.

Col	Column Name	Len	Coltype	Description
7	SATISFACT_ RATING	4	INTEGER	Satisfaction rating. The range of possible values is 0 to 3. ■ 0 (Fully supported) ■ 3 (Unsupported)
8	IMPORTANCE	4	INTEGER	Importance factor. The range of possible values is 1 to 5. ■ 5 (Supports a CSF) ■ 4 (Essential for goal objective achievement) ■ 3 (Essential for business activity) ■ 2 (Useful for goal/objective achievement) ■ 1 (Userful for any purpose)
9	REQUIRE_ WEIGHT	4	INTEGER	Requirement weight. Calculate Rating * Factor
10	REALTIME_OR_ SNAP	1	CHAR	■ R (Real time) ■ S (Snap shot)
11	CATEGORY	1	CHAR	Category: ■ S (Summary) ■ E (Exception) ■ D (Detail) ■ C (Correlation) ■ Q (other)

INTERFACE_TYPE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	INTERFACE_TYPE

Col	Column Name	Len	Coltype	Description
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Interface type name
6	MIN_OCCUR	4	INTEGER	Minimum number of occurrences
7	MAX_OCCUR	4	INTEGER	Maximum number of occurrences
8	AVG_OCCUR	4	INTEGER	Average number of occurrences
9	GROWTH_RATE	4	INTEGER	Growth rate (percent)
10	GROWTH_RATE_PER	1	CHAR	Growth rate period: ■ Y (Year) ■ M (Month) ■ W (Week) ■ D (Day)
11	TYPE	1	CHAR	Type of entity: ■ B (Persistent) ■ D (Transient)
12	NO_INSTANCE	1	CHAR	No instances? ■ space (Instances supported) ■ Y (No instances)
13	CATEGORY	1	CHAR	Category: ■ space (Generic) ■ B (Business object type) ■ T (Task object type)
14	VIEWABLE	1	CHAR	Viewable? ■ space (No) ■ Y (Yes)
15	ENCAP_LEVEL	1	CHAR	Level of encapsulation: ■ space or O (Open) ■ R (Restricted) ■ E (Encapsulated)
16	SUBJECT_AREA_ID	4	INTEGER	ID of subject area that contains this entity type.

Col	Column Name	Len	Coltype	Description
17	IT_MODEL_ID	4	INTEGER	ID of the scoping interface type model.

INTRFCE_TYPE_MDL

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	INTERFACE_TYPE_MDL
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Interface type model name

LIB_USAGE_SCOPE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model SCOPE
2	TBNAME	16	CHAR	LIB_USAGE_SCOPE
3	LIBRARY_USAGE_ID	4	INTEGER	ID of the library usage that is implemented
4	LIBRARY_ID	4	INTEGER	ID of the library that the library usage implements

LIBRARY

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model SCOPE
2	TBNAME	16	CHAR	LIBRARY
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	DATASET_NAME	2000	LONGVAR	Name of data set

LIBRARY_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	LIBRARY_USAGE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ROLE	1	CHAR	Role: ■ S (Generated system) ■ G (Generated items) ■ D (External DBRM) ■ E (External AB load)
6	SEQ	2	SMALLINT	Sequence within technical system. Use ORDER_BY_SEQ to return library usages in the same order as specified in the toolset (zero for generated items library usage).

Col	Column Name	Len	Coltype	Description
7	LIB_TYPE	32	VARCHAR	Library usage type for generated items library usage (zero for the other library usages).
8	TECHSYS_ID	4	INTEGER	ID of the technical system that implements the library usage. Join with TECHNICAL_SYSTEM.

LINKAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	LINKAGE
3	ID	4	INTEGER	Unique Identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the linkage.
6	POINTER_TYPE	1	CHAR	Pointer type: F (Foreign key)
7	DB2_OR_IEF_ENF	1	CHAR	Enforced by: ■ D (DB2) ■ I (System)
8	REF_CONSTRT_ OPT	1	CHAR	Referential constraint option for the relationship it implements: ■ D (Cascade delete) ■ R (Restrict) ■ N (Nullify)
9	RECORD_FROM_ ID	4	INTEGER	ID of the record this linkage is from. Join with RECORD.
10	RECORD_TO_ID	4	INTEGER	ID of the record this linkage is to. Join with RECORD.
11	IDENTIFIER_ID	4	INTEGER	ID of the identifier that this linkage targets

Col	Column Name	Len	Coltype	Description
12	IMPLEMENTATON_ID	4	INTEGER	ID of the implementation where this linkage is used. Join with REL_PART_IMPL.

LINK_DATA_RTND

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	LNK_DATA_RTND
3	DIALOG_FLOW_ID	4	INTEGER	ID of the dialog flow
4	SRC_DATA_VIEW_ID	4	INTEGER	Source data view returned by the link. Join with: ■ GROUP_VIEW ■ ENTITY_VIEW
5	DST_DATA_VIEW_ID	4	INTEGER	Destination data view set by the link. Join with: ■ GROUP_VIEW ■ ENTITY_VIEW

LNK_RTNS_CMD

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	LNK_RTNS_CMD
3	DIALOG_FLOW_ID	4	INTEGER	ID of the dialog flow

Col	Column Name	Len	Coltype	Description
4	COMMAND_ID	4	INTEGER	ID of the command returned by the dialog flow

LNK_RTNS_EXST

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	LNK_RTNS_EXST
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SAVE_INPUT_ VIEWS	1	CHAR	Save input views on link? ■ Y (Yes) ■ N (No)
6	DIALOG_FLOW_ ID	4	INTEGER	ID of the dialog flow
7	EXIT_STATE_ID	4	INTEGER	ID of the exit state on which the link returns

LOCAL_PF_KEY

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	LOCAL_PF_KEY
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	DISPLAY	1	CHAR	Display? ■ Y (Yes) ■ N (No)

Col	Column Name	Len	Coltype	Description
6	COMMAND_ID	4	INTEGER	Command set by this PF key (zero if command is not set by PF key).
7	BUS_PROC_ STEP_ID	4	INTEGER	Procedure step that accepts the local PF key
8	SYSTEM_PF_ KEY_ID	4	INTEGER	System PF key that is overwritten by local PF key

MATRIX

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	MATRIX
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the matrix
6	COEF_VAL_SET	1	CHAR	Coefficient value set: ■ N (Numeric (1-9)) ■ C (CRUD) ■ A (Alphanumeric) ■ X (X | blank | 1-9) ■ R (RAEW)
7	LEAF_ONLY_X	1	CHAR	Leaf objects only on X axis? ■ Y (Yes) ■ N (No)
8	LEAF_ONLY_Y	1	CHAR	Leaf objects only on Y axis? ■ Y (Yes) ■ N (No)
9	IEF_OR_USER_ SUP	1	CHAR	System-defined or user-defined: ■ U (User) ■ I (System)

Col	Column Name	Len	Coltype	Description
10	OBJ_CLASS_X_ID	4	INTEGER	ID of object class on X axis. Join with OBJECT_CLASS.
11	OBJ_CLASS_Y_ID	4	INTEGER	ID of object class on Y axis. Join with OBJECT_CLASS.

MATRIX_USAGE_X

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	MATRIX_USAGE_X
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence within matrix. Use ORDER_BY_SEQ to return the matrix usages on the X axis in the same order as specified in the toolset.
6	MATRIX_ID	4	INTEGER	ID of matrix which uses this matrix usage
7	OBJECT_ID	4	INTEGER	ID of the object that is referred to by this matrix usage

MATRIX_USAGE_Y

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	MATRIX_USAGE_Y
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier

Col	Column Name	Len	Coltype	Description
5	SEQ	2	SMALLINT	Sequence within matrix. Use ORDER_BY_SEQ to return the matrix usages on the Y axis in the same order as specified in the toolset.
6	MATRIX_ID	4	INTEGER	ID of matrix which uses this matrix usage
7	OBJECT_ID	4	INTEGER	ID of the object to which this matrix object refers.

MESSAGE (for Non-Default Dialect)

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	MESSAGE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name for non-default dialect message. Not used.
6	TYPE	1	CHAR	Type for non-default dialect message: ■ I (Informational) ■ W (Warning) ■ E (Error) ■ N (Normal)

MODEL

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model

Col	Column Name	Len	Coltype	Description
2	NAME	32	CHAR	The model name in the Host Encyclopedia
3	CR_DATE	4	INTEGER	Date this model was added to the PI
4	CR_TIME	4	INTEGER	Time this model was added to the PI
5	CR_USERID	8	CHAR	TSO user ID that added this model to the PI
6	ENCY_DATE	4	INTEGER	Date this model was last updated on the Host Encyclopedia
7	ENCY_TIME	4	INTEGER	Time this model was last updated on the Host Encyclopedia
8	ENCY_USERID	8	CHAR	TSO user ID that last updated this model on the Host Encyclopedia

OBJECT_CLASS

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TB_NAME	16	CHAR	OBJECT_CLASS
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the object class
6	IEF_SUPPL_NAME	8	CHAR	System-defined mnemonic
7	IEF_OR_USER_SUP	4	INTEGER	System-defined or user-defined: ■ U (User) ■ I (System)
8	PI_NAME	32	VARCHAR	Name of the Public Interface table that corresponds to this system-defined class. Blank for user-defined.

ORGANIZAT_UNIT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	ORGANIZAT_UNIT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the organizational unit
6	SEQ	2	SMALLINT	Sequence within matrix. Use ORDER_BY_SEQ to return the organizations in the same order as specified in the toolset.
7	WILLBE_SEQ	4	INTEGER	Sequence that will be within matrix. Use ORDER_BY_SEQ to return the organizational units in the same order as specified in the toolset (will be used in a future release).
8	MANAGER_NAME	32	VARCHAR	Name of manager
9	MANAGER_TITLE	32	VARCHAR	Title of manager
10	STATUS	1	CHAR	Organization Status: ■ C (Current) ■ P (Planned)
11	PARENT_ORG_ID	4	INTEGER	ID of the parent organizational unit. Join with ORGANIZAT_UNIT.
12	PARENT_ WILLBE_ID	4	INTEGER	ID of organizational unit. Join with ORGANIZAT_UNIT. Zero for the root organizational unit (will be used in future release).

PAD_CREATE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PAD_CREATE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARENT_ID	4	INTEGER	ID of the parent action block. Join with ACTION_BLOCK.
6	ENTITY_VIEW_ID	4	INTEGER	Entity view this Create acts on.

PAD_DELETE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PAD_DELETE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARENT_ID	4	INTEGER	ID of the parent action block. Join with ACTION_BLOCK.
6	ENTITY_VIEW_ID	4	INTEGER	Entity view upon which this Delete acts.

PAD_FUNCTION

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model

Col	Column Name	Len	Coltype	Description
2	TBNAME	16	CHAR	PAD_FUNCTION
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of function. (System-defined function)
6	INTEXT	1	CHAR	Internal or external function: ■ I (Internal) ■ E (External)
7	IEF_SUPPLIED	1	CHAR	Function supplied by system? ■ Y (Yes) ■ N (No)
8	DOMAIN	1	CHAR	Domain of function: ■ T (Text) ■ D (Date) ■ M (Time) ■ N (Number)
9	CODEGEN_NAME	32	VARCHAR	Codegen name
10	USE_SELECTION	1	CHAR	Usable in READ statement? ■ Y (Yes) ■ N (No)
11	INTRINSIC	1	CHAR	Intrinsic or not? ■ Y (Yes) ■ N (No)
12	DBMS_NAME	8	CHAR	Name of database management system
13	OPT	1	CHAR	Optionality: ■ M (Mandatory) ■ O (Optional)

PAD_READ

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PAD_READ
3	ID	4	INTEGER	Identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARENT_ID	4	INTEGER	ID of the parent action block. Join with ACTION_BLOCK.
6	ENTITY_VIEW_ID	4	INTEGER	Entity view upon which this Read acts.

PAD_SET_ATTR

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PAD_SET_ATTR
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARENT_ID	4	INTEGER	ID of the parent action block. Join with ACTION_BLOCK.
6	ATTR_VIEW_ID	4	INTEGER	Attribute view upon which this set acts.

PAD_UPDATE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PAD_UPDATE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARENT_ID	4	INTEGER	ID of the parent action block. Join with ACTION_BLOCK.
6	ENTITY_VIEW_ID	4	INTEGER	Entity view upon which this Update acts.

PARM

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the field for which this is a parameter
2	TEXT	2000	LONGVAR	Text of the parameter

PARM_DELIMITER

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PARM_DELIMITER
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier

Col	Column Name	Len	Coltype	Description
5	SEQ	2	SMALLINT	Sequence within procedure step. Use ORDER_BY_SEQ to return parameter delimiters in the same order as specified in the toolset.
6	VALUE	1	CHAR	Parameter delimiter value
7	BUS_PROC_ STEP_ID	4	INTEGER	ID of the procedure step that uses this parameter delimiter (zero for parameter delimiter that is not used by procedure step. For future release).
8	BUSINESS_SYS_ ID	4	INTEGER	ID of the business system that contains this parameter delimiter

PARM_STRING_DEL

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PARM_STRING_DEL
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence within procedure step. Use ORDER_BY_SEQ to return parameter delimiters in the same order as specified in the toolset.
6	INITIATOR	1	CHAR	Parameter string delimiter initiator
7	TERMINATOR	1	CHAR	Parameter string delimiter terminator
8	BUS_PROC_ STEP_ID	4	INTEGER	ID of the procedure step that uses this parameter string delimiter. Zero for parameter string delimiter that is not used by procedure step (for future release).

Col	Column Name	Len	Coltype	Description
9	BUSINESS_SYS_ ID	4	INTEGER	ID of the business system that contains this parameter delimiter.

PARTITIONING

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PARTITIONING
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	FULLY_ENUM	1	CHAR	Fully enumerated partitioning? ■ Y (Yes) ■ N (No)
6	LIFE_CYCLE	1	CHAR	Life cycle partitioning? ■ Y (Yes) ■ N (No)
7	SEQ	4	INTEGER	Sequence number for presentation
8	PARENT_ENTITY_ ID	4	INTEGER	Entity type or subtype for which this is a partitioning. Join with: ■ ENTITY_TYPE ■ ENTITY_SUBTYP

PDD_ATOM_DEP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model

Col	Column Name	Len	Coltype	Description
2	TBNAME	16	CHAR	PDD_ATOM_DEP
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PDD_DEPENDNCY_ID	4	INTEGER	Dependency ID
6	EXPORT_USAGE_ ID	4	INTEGER	From side of dependency. Join with: ACTIV_USAGE XT_OBJ_USAGE XT_EVNT_USAGE PDD_MUTL_EXCL PDD_PARALLEL PDD_CLOSURE
7	IMPORT_USAGE_ ID	4	INTEGER	To side of dependency. Join with: ACTIV_USAGE XT_OBJ_USAGE PDD_MUTL_EXCL PDD_PARALLEL PDD_CLOSURE

PDD_CLOSURE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PDD_CLOSURE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARENT_ACTIV_ ID	4	INTEGER	Parent activity ID. Join with: FUNCTION_DEF PROCESS_DEF

Col	Column Name	Len	Coltype	Description
6	PDD_MUTL_ EXCL_ID	4	INTEGER	PDD mutually exclusive construct being closed

PDD_DEPENDNCY

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PDD_DEPENDNCY
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Dependency name

PDD_EXT_FLOW

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PDD_EXT_FLOW
3	PDD_ DEPENDNCY_ID	4	INTEGER	ID of the dependency
4	DATA_VIEW_ID	4	INTEGER	Data view ID that flows on this flow. Join with: GROUP_VIEW ENTITY_VIEW

PDD_MUTL_EXCL

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PDD_MUTL_EXCL
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARENT_ACTIV_ ID	4	INTEGER	Parent activity ID. Join with: FUNCTION_DEF PROCESS_DEF

PDD_PARALLEL

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PDD_PARALLEL
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	PARENT_ACTIV_ ID	4	INTEGER	Parent activity ID. Join with: FUNCTION_DEF PROCESS_DEF

PERFORM_MEASURE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PERFORM_MEASURE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the performance measure

PERMIT_VALUE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PERMIT_VALUE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	LOW_VALUE	4	INTEGER	ID of permitted value or value at low end of the range Join with PERMIT_VALUE_LOW. (This column will be deleted in a future release).
6	HIGH_VALUE	4	INTEGER	ID of the high end of the range. Join with PERMIT_VALUE_HI. This column will be deleted in a future release.
7	ATTRIBUTE_ID	4	INTEGER	Attribute for which this is a permitted value range

PERMIT_VALUE_HI

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of permitted value for which this is a high end of the range (zero for permitted value that is not a range).
2	VALUE	2000	LONGVAR	Value of the upper limit

PERMIT_VALUE_LOW

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of permitted value for which this is a low end of the range.
2	VALUE	2000	LONGVAR	Value of permitted value (or the lower limit)

PROCESS_DEF

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PROCESS_DEF
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Process name
6	ELEMENTARY	1	CHAR	Elementary process? Y (Yes) N (No)

Col	Column Name	Len	Coltype	Description
7	REPEATED	1	CHAR	Repeated process? Y (Yes) N (No)
8	SUG_ MECHANISM	1	CHAR	Suggested mechanism: B (Batch) L (Online) M (Manual) O (Other)
9	ACTION_ BLOCK_ID	4	INTEGER	ID of the action block that details this process (zero if not detailed by an action block).

PROMPT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PROMPT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ATTRIBUTE_ID	4	INTEGER	ID of the attribute to which the prompt applies
6	TEXT	2000	LONGVAR	Text of the prompt

REC_ENTRY_PT_USE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	REC_ENTRY_PT_USE
3	ID	4	INTEGER	Unique identifier

Col	Column Name	Len	Coltype	Description
4	ORG_ID	4	INTEGER	Original object identifier
5	ROLE	1	CHAR	Role of use: S (Source) T (Target) B (Both) P (Pointer)
6	UNIQUE	1	CHAR	Is the entry point unique? Y (Yes) N (No)
7	RECORD_ID	4	INTEGER	ID of the record referenced by this entry point
8	ENTRY_POINT_ID	4	INTEGER	ID of the entry point used by this record

PERFORM_MEASURE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PERFORM_MEASURE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the performance measure

PERMIT_VALUE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PERMIT_VALUE

Col	Column Name	Len	Coltype	Description
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	LOW_VALUE	4	INTEGER	ID of permitted value (or value at low end of the range). Join with PERMIT_VALUE_LOW (this column will be deleted in a future release).
6	HIGH_VALUE	4	INTEGER	ID of the high end of the range. Join with PERMIT_VALUE_HI (this column will be deleted in a future release).
7	ATTRIBUTE_ID	4	INTEGER	Attribute for which this is a permitted value range

PERMIT_VALUE_HI

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of permitted value for which this is a high end of the range (zero for permitted value that is not a range).
	VALUE	2000	LONGVAR	Value of the upper limit

PERMIT_VALUE_LOW

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of permitted value for which this is a low end of the range.
2	VALUE	2000	LONGVAR	Value of permitted value (or the lower limit)

PROCESS_DEF

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PROCESS_DEF
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Process name
6	ELEMENTARY	1	CHAR	Elementary process? Y (Yes) N (No)
7	REPEATED	1	CHAR	Repeated process? Y (Yes) N (No)
8	SUG_MECHANISM	1	CHAR	Suggested mechanism: B (Batch) L (Online) M (Manual) O (Other)
9	ACTION_BLOCK_ ID	4	INTEGER	ID of the action block that details this process (zero if not detailed by an action block).

PROMPT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	PROMPT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier

Col	Column Name	Len	Coltype	Description
5	ATTRIBUTE_ID	4	INTEGER	ID of the attribute to which the prompt applies
6	TEXT	2000	LONGVAR	Text of the prompt

REC_ENTRY_PT_USE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	REC_ENTRY_PT_USE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ROLE	1	CHAR	Role of use: S (Source) T (Target) B (Both) P (Pointer)
6	UNIQUE	1	CHAR	Is the entry point unique? Y (Yes) N (No)
7	RECORD_ID	4	INTEGER	ID of the record referenced by this entry point
8	ENTRY_POINT_ID	4	INTEGER	ID of the entry point used by this record

RECORD

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	RECORD

Col	Column Name	Len	Coltype	Description
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Record name
6	MACRO_NAME	32	VARCHAR	Macro name
7	EDITPROC_NAME	8	CHAR	Edit proc routine name
8	VALIDPROC_NAME	8	CHAR	Validation proc routine name
9	OWNER_NAME	8	CHAR	DB2 owner identifier
10	DB2_SYNONYM	8	CHAR	DB2 synonym for table
11	ROLE	1	CHAR	Role: D (Data record) L (Link record)
12	DBMS	1	CHAR	Database management system: 2 (DB2) 1 (DL/1)
13	TYPE	1	CHAR	Record type: P (Physical) L (Logical) V (Virtual)
14	DATA_STORE_ID	4	INTEGER	ID of the containing data store. Join with DATA_STORE_TBLSP.

RECORD_REFERENCE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	RECORD_REFERENCE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	RECORD_ID	4	INTEGER	ID of the record that is referenced

Col	Column Name	Len	Coltype	Description
6	IMPL_LOGIC_ID	4	INTEGER	ID of the implementation logic that makes reference to the record. Join with IMPLEMENT_LOGIC.

REL_IDENT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	REL_IDENT
3	ID	4	INTEGER	Identifier (not unique)
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence within entity type or subtype. Use ORDER_BY_SEQ to return relationship identifiers in the same order as specified in the toolset.
6	NAME	32	VARCHAR	Name of identifier
7	PRIMARY_KEY	1	CHAR	Primary identifier for entity type? Y (Yes) N (No)
8	ENTITY_ID	4	INTEGER	Entity type or subtype for which this relationship is an identifier. Join with: ENTITY_TYPE ENTITY_SUBTYP
9	RELATIONSHIP_ ID	4	INTEGER	Relationship ID

REL_MUTL_EXCL

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	REL_MUTL_EXCL
3	ID	4	INTEGER	Identifier (not unique)
4	ORG_ID	4	INTEGER	Original object identifier
5	ENTITY_ID	4	INTEGER	Entity type or subtype that has the mutually exclusive relationship. Join with: ENTITY_TYPE ENTITY_SUBTYP
6	RELATIONSHIP_ID	4	INTEGER	A relationship that is part of a mutually exclusive set

REL_PART_IMPL

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	REL_PART_IMPL
3	ID	4	INTEGER	Unique identifier
5	TECHNIQUE	1	CHAR	Implementation technique: F (Foreign key) M (Many-to-many)
6	REL_PART_ID	4	INTEGER	ID of the relationship or partitioning implemented. Join with: RELATIONSHIP PARTITIONING

REL_VIEW

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	REL_VIEW
3	ID	4	CHAR	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence within entity view. Use ORDER_BY_SEQ to return relationship views in the same order as specified in the toolset.
6	REQUIRED_INPUT	1	CHAR	Is this a required import view? Y (Yes) N (No)
7	ENTITY_VIEW_ID	4	INTEGER	Parent entity view
8	RELATIONSHIP_ID	4	INTEGER	Relationship included in this entity view

RELATIONSHIP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	RELATIONSHIP
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Relationship name
6	NAME_INVERSE	32	VARCHAR	Inverse relationship name
7	TYPE	1	CHAR	Relationship type: S (Source) D (Destination)

Col	Column Name	Len	Coltype	Description
8	TRANSFERABLE	1	CHAR	Transferable? Y (Yes) N (No)
9	OPT	1	CHAR	Optionality: M (Always) O (Sometimes)
10	OPT_EXPECTED_ PCT	4	INTEGER	Expected optionality (percent)
11	CARD	1	CHAR	Cardinality: 1 (One) M (One or many)
12	MIN_CARD	4	INTEGER	Cardinality - at least
13	MAX_CARD	4	INTEGER	Cardinality - at most
14	AVG_CARD	4	INTEGER	Cardinality - on average
15	CARD_MIN_ABS_ EST	1	CHAR	Absolute or estimated minimum cardinality: E (Estimated) A (Absolute)
16	CARD_MAX_ABS_ EST	1	CHAR	Absolute or estimated maximum cardinality: E (Estimated) A (Absolute)
17	CASCADE_OPTION	1	CHAR	Cascade option (Deletion Rule) for second entity type: D (Cascade delete) R (Restrict (disallow)) N (Nullify (disassociate)) C (Deletion rule not set)
18	MOD_OR_REF	1	CHAR	Associate option: M (Modifying) R (Referencing)
19	DISPLAY_NAME	1	CHAR	Display relationship name? Y (Yes) N (No)

Col	Column Name	Len	Coltype	Description
20	SOURCE_ENTITY_ ID	4	INTEGER	Entity type or subtype from which this relationship comes. Join with: ENTITY_TYPE ENTITY_SUBTYP
21	DEST_ENTITY_ID	4	INTEGER	Entity type or subtype to which this relationship goes. Join with: ENTITY_TYPE ENTITY_SUBTYP
22	INVERSE_REL_ID	4	INTEGER	ID of inverse relationship. Join with RELATIONSHIP.
23	TYPE_PERS_TRAN	1	CHAR	Type of relationship: B (Persistent) D (Transient)
24	ENCAP_LEVEL	1	CHAR	Level of encapsulation: space (Public) P (Protected) E (Encapsulated)

SCREEN_DEF

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCREEN_DEF
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	DEV_TYPE	1	CHAR	Device type for the screen. Not used.
6	AUTO_SCROLL	1	CHAR	Does the screen perform automatic scrolling? Y (Yes) N (No)

Col	Column Name	Len	Coltype	Description
7	BELL_ACTIVE	1	CHAR	Is the bell active? Y (Yes) N (No)
8	DFLT_SCROLL_LOC	1	CHAR	Default start item for scroll? T (Top) L (Last)
9	UPDATE_DISPLAY	1	CHAR	Update past display? Y (Yes) N (No)
10	BUS_PROC_STEP_ID	4	INTEGER	ID of the procedure step to which the screen applies

SCREEN_TMPLT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCREEN_TMPLT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the template
6	DEV_TYPE	1	CHAR	Type of device that the template supports. Not used.
7	BUSINESS_SYS_ID	4	INTEGER	Business system that owns the template

SCRN_FLD_LIT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model

Col	Column Name	Len	Coltype	Description
2	TBNAME	16	CHAR	SCRN_FLD_LIT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ROW	4	INTEGER	Row position of the field
6	COL	4	INTEGER	Column position of the field
7	DISPLAY_LENGTH	4	INTEGER	Display length of the field
8	CSTM_DFLT	1	CHAR	Video properties: D (Default) C (Custom)
9	COLOR	1	CHAR	Color of the field
10	INTENSITY	1	CHAR	Intensity of the field
11	HIGHLIGHT	1	CHAR	Highlight level for the field
12	SCREEN_DEF_ID	4	INTEGER	ID of the screen that contains the field. Join with: SCREEN_DEF SCREEN_TMPLT
13	TEXT	2000	LONGVAR	Text of the literal

SCRN_FLD_PRMT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCRN_FLD_PRMT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ROW	4	INTEGER	Row position of the field
6	COL	4	INTEGER	Column position of the field
7	DISPLAY_LENGTH	4	INTEGER	Display length of the field

Col	Column Name	Len	Coltype	Description
8	CSTM_DFLT	1	CHAR	Video properties: D (Default) C (Custom)
9	COLOR	1	CHAR	Color of the field
10	INTENSITY	1	CHAR	Intensity of the field
11	HIGHLIGHT	1	CHAR	Highlight level for the field
12	PROMPT_ID	4	INTEGER	ID of the prompt that is used by the field
13	SCREEN_DEF_ID	4	INTEGER	ID of the screen that contains the field. Join with: SCREEN_DEF SCREEN_TMPLT
14	SCRN_FLD_VAR_ ID	4	INTEGER	ID of the variable screen field that has the field. Join with SCRN_FLD_VAR

SCRN_FLD_VAR

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCRN_FLD_VAR
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	ROW	4	INTEGER	Row position of the field
6	COL	4	INTEGER	Column position of the field
7	BLANK_WHEN_ ZERO	1	CHAR	Blank field when zero? Y (Yes) N (No)
8	DFLT_EDT_PTRN_ ID	4	INTEGER	ID of the default edit pattern used by this field, if present (may be zero)

Col	Column Name	Len	Coltype	Description
9	SCRN_RG_OCC_ ID	4	INTEGER	If this is part of a repeating group, this is the ID of the controlling RG OCC. Otherwise, zero.
10	SCRN_VAR_DEF_ ID	4	INTEGER	ID of the variable that is used in the field
11	SCREEN_DEF_ID	4	INTEGER	ID of the screen that contains the field. Join with: SCREEN_DEF SCREEN_TMPLT

SCRN_FLD_VARE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCRN_FLD_VARE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	CSTM_DFLT	1	CHAR	Video properties: D (Default) C (Custom)
6	COLOR	1	CHAR	Color of the field
7	INTENSITY	1	CHAR	Intensity of the field
8	HIGHLIGHT	1	CHAR	Highlight level for the field
9	VIDEO_CURSOR	1	CHAR	Insert cursor? Y (Yes) N (No)
10	PROTECTION	1	CHAR	Protected? Y (Yes) N (No)

Col	Column Name	Len	Coltype	Description
11	JUSTIFICATION	1	CHAR	Field justification: L (Left) R (Right)
12	FILL_CHAR	1	CHAR	Field fill character. Not used.

SCRN_FLD_VARP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCRN_FLD_VARP
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	CSTM_DFLT	1	CHAR	Video properties: D (Default) C (Custom)
6	COLOR	1	CHAR	Color of the field
7	INTENSITY	1	CHAR	Intensity of the field
8	HIGHLIGHT	1	CHAR	Highlight level for the field
9	VIDEO_CURSOR	1	CHAR	Insert cursor? Y (Yes) N (No)
10	PROTECTION	1	CHAR	Protected? Y (Yes) N (No)
11	JUSTIFICATION	1	CHAR	Field justification: L (Left) R (Right)
12	FILL_CHAR	1	CHAR	Field fill character

SCRN_HELP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of SCREEN_DEF, SCRN_VAR_DEF, or SCRN_SYS_DEF for which this is a screen help identifier.
2	TEXT	2000	LONGVAR	Text of the help identifier.

SCRN_RG_OCC

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCRN_RG_OCC
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	AUTO_SCROLL	1	CHAR	Automatic scrolling? Y (Yes) N (No)
6	PARENT_ID	4	INTEGER	Parent construct. Join with: SCRN_RP_GRP SCRN_RG_OCC
7	SCREEN_DEF_ID	4	INTEGER	ID of the screen that contains the occurrence

SCRN_RP_GRP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCRN_RP_GRP
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	DFLT_SCROLL_LOC	1	CHAR	Default start item for scroll? Y (Yes) N (No)
6	UPDATE_DISPLAY	1	CHAR	Update past display? Y (Yes) N (No)
7	GROUP_VIEW_ID	4	INTEGER	ID of the group view that controls the repetition
8	SCREEN_DEF_ID	4	INTEGER	ID of the screen that contains the variable

SCRN_SYS_DEF

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCRN_SYS_DEF
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	EDIT_PTRN_ CLASS	1	CHAR	Edit pattern class of the field
6	DISPLAY_LENGTH	4	INTEGER	Display length of the field
7	DISPLAY_ DECIMALS	4	INTEGER	Number of decimal places displayed

Col	Column Name	Len	Coltype	Description
8	DYN_ATTR_MOD	1	CHAR	Unused. Dynamic attribute modification allowed: Y (Yes) N (No)
9	SYS_ATTRIBUTE_ ID	4	INTEGER	ID of the system attribute that appears in the field
10	SCREEN_DEF_ID	4	INTEGER	ID of the screen that contains the field. Join with: SCREEN_DEF SCREEN_TMPLT

SCRN_VAR_DEF

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCRN_VAR_DEF
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	EDIT_PTRN_ CLASS	1	CHAR	Edit pattern class of the variable
6	DISPLAY_ LENGTH	4	INTEGER	Display length of the variable
7	DISPLAY_ DECIMALS	4	INTEGER	Number of decimal places displayed
8	DYN_ATTR_MOD	1	CHAR	Dynamic modification of video attributes: Y (Yes) N (No)
9	DLG_MGMT_ HIDDEN	1	CHAR	Dialog management hidden field? Y (Yes) N (No)

Col	Column Name	Len	Coltype	Description
10	SCREEN_DEF_ID	4	INTEGER	ID of the screen that contains the variable

SCRN_VAR_IO

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCRN_VAR_IO
3	INOUT	1	CHAR	Is this input (I) from or output (O) to the screen?
4	ATTR_VIEW_ID	4	INTEGER	Attribute view of the displayed attribute
5	SCRN_VAR_DEF_ID	4	INTEGER	ID of screen variable definition

SCROLL_AMOUNT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SCROLL_AMOUNT
3	ID	4	CHAR	Unique Identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of scroll amount display
6	ACCEL_VALUE	1	CHAR	Accelerator value
7	SCRP_AMT_DISP	2000	LONGVAR	Scroll amount display

SESSION

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SESSION
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	CHANGE_DATE	4	INTEGER	Date of last change
6	CHANGE_TIME	4	INTEGER	Time of last change
7	CHANGE_ USERID	8	CHAR	Last change by user
8	IDENT_NUMBER	4	INTEGER	Identification number

SPEC_TYPE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SPEC_TYPE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	CHAR	Interface type name
6	MIN_OCCUR	4	INTEGER	Minimum number of occurrences.
7	MAX_OCCUR	4	INTEGER	Maximum number of occurrences.
8	AVG_OCCUR	4	INTEGER	Average number of occurrences.
9	GROWTH_RATE	4	INTEGER	Growth rate (percent).

Col	Column Name	Len	Coltype	Description
10	GROWTH_RATE_PER	1	CHAR	Growth rate period: Y (Year) M (Month) W (Week) D (Day)
11	TYPE	1	CHAR	Type of entity: B (Persistent) D (Transient)
12	NO_INSTANCE	1	CHAR	No instances? space (Instances supported) Y (No instances)
13	CATEGORY	1	CHAR	Category: space (generic) B (Business object type) T (Task object type)
14	VIEWABLE	1	CHAR	Viewable? space (No) Y (Yes)
15	ENCAP_LEVEL	1	CHAR	Level of encapsulation: space or O (Open) R (Restricted) E (Encapsulated)
16	SUBJECT_AREA_ID	4	INTEGER	ID of subject area that contains this entity type

STG_DVOL_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	STG_DVOL_USAGE
3	DASD_VOLUME_ID	4	INTEGER	ID of the DASD volume used by the storage group

Col	Column Name	Len	Coltype	Description
4	STORAGE_GRP_ ID	4	INTEGER	ID of the storage group using the DASD volume

STORAGE_GROUP

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	STORAGE_GROUP
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	8	CHAR	Storage group name
6	VSAM_CAT_ NAME	8	CHAR	VSAM catalog name
8	TECHDESN_ID	4	INTEGER	ID of the technical design this storage group is for. Join with TECHNICAL_DESIGN.

STRATEGY

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	STRATEGY
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the strategy
6	SEQ	2	SMALLINT	Sequence within strategy. Use ORDER_BY_SEQ to return strategies in the same order as the specific toolset.

Col	Column Name	Len	Coltype	Description
7	PRIORITY	4	INTEGER	Priority The range of possible values is 0, for low to 9 for high
8	PARENT_STRAT_ID	4	INTEGER	ID of the parent strategy. Join with STRATEGY. Zero for root strategy (will be used in a future release).

SUBJECT_AREA

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SUBJECT_AREA
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of subject area
6	PARENT_SUBJ_ID	4	INTEGER	ID of parent subject area. Join with SUBJECT_AREA (zero for root subject area).

SYS_ATTRIBUTE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SYS_ATTRIBUTE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier

Col	Column Name	Len	Coltype	Description
5	NAME	32	VARCHAR	Name of the system-defined attribute for the system-defined entity type $IEF.
6	TYPE	1	CHAR	B (Basic)
7	DOMAIN	1	CHAR	Domain of the system-defined attribute
8	LENGTH	4	INTEGER	Length of the system-defined attribute
9	DEC_PLACES	4	INTEGER	0 (zero)
10	INPUT	1	CHAR	Can this attribute be input? ■ Y (Yes) ■ N (No)
11	OUTPUT	1	CHAR	Can this attribute be output? ■ Y (Yes) ■ N (No)
12	ENTITY_TYPE_ ID	4	INTEGER	ID of the system-defined entity type. Join with SYS_ENT_TYPE.
13	DESCRIPTION	2000	LONGVAR	Description of the system-defined attribute

SYS_ENT_TYPE (Work Attribute Set)

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SYS_ENT_TYPE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Entity type name (work attribute set)

Col	Column Name	Len	Coltype	Description
6	MIN_OCCUR	4	INTEGER	Minimum number of occurrences. Not used.
7	MAX_OCCUR	4	INTEGER	Maximum number of occurrences. Not used.
8	AVG_OCCUR	4	INTEGER	Average number of occurrences. Not used.
9	GROWTH_RATE	4	INTEGER	Growth rate (percent). Not used.
10	GROWTH_RATE_PER	1	CHAR	Growth rate period (Not used).: ■ Y (Year) ■ M (Month) ■ W (Week) ■ D (Day)

SYSTEM_PF_KEY

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	SYSTEM_PF_KEY
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NUMBER	4	INTEGER	Number of the system PF key
6	STANDARD	1	CHAR	Standard? ■ Y (Yes) ■ N (No)
7	COMMAND_ID	4	INTEGER	Command set by this PF key
8	BUSINESS_SYS_ID	4	INTEGER	Business system that contains the system PF key

TACTIC

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	TACTIC
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the tactic
6	SEQ	2	SMALLINT	Sequence within tactic. Use ORDER_BY_SEQ to return tactics in the same order as the specific toolset.
7	PRIORITY	4	INTEGER	Priority The range of possible values is 0 for low to 9 for high.
8	PARENT_ TACTIC_ID	4	INTEGER	ID of the parent tactic. Join with TACTIC. Zero for root tactic (will be used in a future release).

TD_LIBRARY_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	TD_LIBRARY_USAGE
3	ID	4	INTEGER	Unique Identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence within technical design. Use ORDER_BY_SEQ to return library usages in same order as specified.

Col	Column Name	Len	Coltype	Description
6	LIB_TYPE	32	VARCHAR	Library type usage
7	LIBRARY_ID	4	INTEGER	ID of the library that implements the library usage. Join with LIBRARY.
8	TECH_DESIGN_ ID	4	INTEGER	ID of the technical design that the library usage is for. Join with TECHNICAL_DESIGN.

TECHNICAL_DESIGN

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	TECHNICAL_DESIGN
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the technical design
6	OWNER_NAME	8	CHAR	DB2 owner identifier
7	VSAM_CAT_NAME	8	CHAR	Name of the VSAM catalog
8	DBMS	1	CHAR	Database management system: ■ 2 (DB2 z/OS) ■ A (Datacom) ■ B (ODBC/ADO.NET) ■ C (MS/SQL) ■ E (DB2 UDB) ■ H (NONE) ■ J (JDBC) ■ O (Oracle)
9	RESERVED_WORD	1	CHAR	Reserved word check? ■ Y (Yes) ■ N (No)

Col	Column Name	Len	Coltype	Description
10	REF_INTEGRITY	1	CHAR	Referential Integrity Default enforcement: ■ D (DB2) ■ I (System)
11	STG_VSAM_DEF	1	CHAR	Storage group/VSAM default: ■ S (Storage group) ■ V (VSAM)
12	DEF_STG_DB_ID	4	INTEGER	ID of the default storage group for databases. Join with STORAGE_GROUP (zero if ID does not exist).
13	DEF_STG_TBLSP_ ID	4	INTEGER	ID of the default storage group for tablespaces (zero if ID does not exist).
14	DEF_STG_INDEX_ ID	4	INTEGER	ID of the default storage group for indexspaces (zero if ID does not exist).

TECHNICAL_SYSTEM

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	1ID of the containing model
2	TBNAME	16	CHAR	TECHNICAL_SYSTEM
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	HOST_TELEPROCESS	1	CHAR	Host teleprocessing monitor: ■ I (IMS) ■ C (CICS) ■ T (TSO)
6	SOURCE_LANGUAGE	32	VARCHAR	Name of source language

Col	Column Name	Len	Coltype	Description
7	DBMS	1	CHAR	Database management system: ■ 2 (DB2 z/OS) ■ A (Datacom) ■ B (ODBC/ADO.NET) ■ C (MSSQL) ■ E (DB2 UDB) ■ H (NONE) ■ J (JDBC) ■ O (Oracle)
8	PROFILE_MANAGER	1	CHAR	Type of profile manage, D (DB2)
9	SCREEN_HANDLER	1	CHAR	Type of screen handler: ■ T (Bypass) ■ M (Mapped)
10	DB2_SYS_IDENT	8	CHAR	Name of DB2 system identification
11	APPLIC_IDENT	32	VARCHAR	Name of application ID
12	RESTART_INDIC	1	CHAR	Restart or reset indicator: ■ Y (Yes) ■ N (No)
13	EXTENDED_ATTR	1	CHAR	Allow extended attribute not output? ■ Y (Yes) ■ N (No)
14	HELP_EXIT	8	CHAR	Name of help routine exit csect
15	DEC_PNT_OR_ COMMA	1	CHAR	Decimal point or comma? ■ . (Decimal point) ■ , (Comma)
16	DB2_REFERENCE	1	CHAR	DB2 reference fully qualified? ■ Y (Yes) ■ N (No)
17	TRAP_ABENDS	1	CHAR	Trap abends from CICS? ■ Y (Yes) ■ N (No)

Col	Column Name	Len	Coltype	Description
18	TRAP_PA_KEYS	1	CHAR	Trap CLEAR, PA1, PA2, PA3 keys? ■ Y (Yes) ■ N (No)
19	START_XCTL	1	CHAR	Use START or XCTL in CICS? S (Start)
20	COMPILER_LANG	1	CHAR	Computer language dialect: ■ 2 (VSCOB) ■ 4 (CSC) ■ 5 (UNIXC) ■ G (GCC) ■ J (JAVAC) ■ V (MSVCNT)
21	OPERATING_SYSTEM	32	VARCHAR	Operating system
22	RESTARTABLE	1	CHAR	Restartable application? ■ Y (Yes) ■ N (No)
23	BUSINESS_SYS_ID	4	INTEGER	Business system that contains this technical system

TEXT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	ID	4	INTEGER	ID of object
2	TEXT	2000	LONGVAR	Text

TMPLT_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	TMPLT_USAGE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SCREEN_DEF_ID	4	INTEGER	ID of the screen that uses the template
6	SCREEN_TMPLT_ID	4	INTEGER	ID of the template that is used. Join with SCREEN_TMPLT

UNFORMAT_INPUT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	UNFORMAT_INPUT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	SEQ	2	SMALLINT	Sequence within procedure step. Use ORDER_BY_SEQ to return unformatted input in the same order as specified in the toolset.
6	POSITION_ NUMBER	4	INTEGER	Position number
7	ATTR_VIEW_ID	4	INTEGER	ID of the attribute view that is input to the unformatted input (zero for unformatted input that does not input to an attribute view).

Col	Column Name	Len	Coltype	Description
8	SYS_ATTRIBUTE_ ID	4	INTEGER	ID of the system attribute that is referenced by the unformatted input (zero for unformatted input that does not reference a system attribute).
9	BUS_PROC_STEP_ ID	4	INTEGER	ID of the procedure step that uses the unformatted input

UNFRMT_INP_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	UNFRMT_INP_USAGE
3	PROMPT_ID	4	INTEGER	ID of the prompt that is implemented
4	UNFORMAT_INP_ ID	4	INTEGER	ID of the unformatted input this implements. Join with UNFORMAT_INPUT.

USE_DATA_RTND

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	USE_DATA_RTND
3	ACTN_BLK_USE_ID	4	INTEGER	ID of action block usage

Col	Column Name	Len	Coltype	Description
4	SRC_DATA_VIEW_ ID	4	INTEGER	Entity view in USEd action block that contains data being returned. Join with: ■ GROUP_VIEW ■ ENTITY_VIEW
5	DST_DATA_VIEW_ ID	4	INTEGER	Entity view in USEing action block that receives returned data. Join with: ■ GROUP_VIEW ■ ENTITY_VIEW

USE_DATA_SENT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	USE_DATA_SENT
3	ACTN_BLK_USE_ID	4	INTEGER	ID of action block usage
4	SRC_DATA_VIEW_ ID	4	INTEGER	Entity view in USEing action block that contains data being sent. Join with: ■ GROUP_VIEW ■ ENTITY_VIEW
5	DST_DATA_VIEW_ ID	4	INTEGER	Entity view in USEd action block that receives sent data. Join with: ■ GROUP_VIEW ■ ENTITY_VIEW

USER_DEF_OBJECT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	USER_DEF_OBJECT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	Name of the user-defined object
6	SEQ	2	SMALLINT	Not used. This column will be deleted in a future release.
7	CLASS_TYPE	32	VARCHAR	NAME of the user-defined class that this user-defined object belongs to
8	OBJECT_CLASS_ID	4	INTEGER	ID of the user-defined object class to which this user-defined object belongs.
9	PARENT_OBJECT_ ID	4	INTEGER	ID of the parent user-defined object. Join with USER_DEF_OBJECT. Zero for root user-defined object (for future release).

VIEW_SET

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	VIEW_SET
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	TYPE	32	VARCHAR	Import, export, local, or entity action

Col	Column Name	Len	Coltype	Description
6	ACTIVITY_ ID	4	INTEGER	ID of the function, process, procedure step, action block, or PAD function for which this is a viewset. Join with: ■ FUNCTION_DEF ■ PROCESS_DEF ■ BUS_PROC_STEP ■ ACTION_BLOCK ■ PAD_FUNCTION

XT_EVNT_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	XT_EVNT_USAGE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	TYPE	32	VARCHAR	Import or export - event to activity
6	PARENT_ACTIV_ID	4	INTEGER	Parent activity ID. Join with: ■ FUNCTION_DEF ■ PROCESS_DEF
7	XTERNL_EVENT_ID	4	INTEGER	External event ID

XT_OBJ_USAGE

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	XT_OBJ_USAGE
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	TYPE	32	VARCHAR	Import or export - object to activity
6	PARENT_ACTIV_ID	4	INTEGER	Parent activity ID. Join with: ■ FUNCTION_DEF ■ PROCESS_DEF
7	XTERNL_OBJECT_ ID	4	INTEGER	External object ID

XTERNL_EVENT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	XTERNL_EVENT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	External event name

XTERNL_OBJECT

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	XTERNL_OBJECT
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	NAME	32	VARCHAR	External object name

XT_IMPL_LOGIC

The following table details various column names along with their descriptions.

Col	Column Name	Len	Coltype	Description
1	MODEL_ID	4	INTEGER	ID of the containing model
2	TBNAME	16	CHAR	XT_IMPL_LOGIC
3	ID	4	INTEGER	Unique identifier
4	ORG_ID	4	INTEGER	Original object identifier
5	MEMBER_NAME	8	CHAR	Member name
6	SOURCE_ LANGUAGE	32	VARCHAR	Name of source language
7	DATE_GENERATED	4	CHAR	Generation date
8	TIME_GENERATED	4	CHAR	Time generated
9	GEN_BY_USER	8	CHAR	Generated by user ID
10	OK_SESSION_ID	4	INTEGER	Session ID at which checked OK
11	ACTION_BLOCK_ID	4	INTEGER	ID of the action block implemented by the external implementation logic. Join with ACTION_BLOCK.

Appendix D: Possible Joins in the Public Interface

The following table definitions list only the table name and those columns that represent some form of key. Each table definition lists the model_ID and unique table ID. Each also lists the columns that represent foreign keys (foreign key names end in the letters _ID). The last column (Tables with Which to Join) lists possible joins for each column that represents a foreign key.

In each instance, the possible joins column should be joined with the ID column of the target table. For example, in table ACTIV_USAGE, column PARENT_ ACTIV_ID can be joined with column ID of tables FUNCTION_DEF or PROCESS_DEF. Further, column CHILD_ACTIV_ID can be joined with the ID column of tables FUNCTION_DEF or PROCESS_DEF.

Most tables contain ORG_ID in column 4. This is the Original Object ID used to establish common ancestry among objects. ORG_ID can be used with objects of the same type in performing joins but not with dissimilar objects types. For example, you cannot join an entity type to an attribute using ORG_ID as there could never be a match.

ACTION_BLOCK

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	BUSINESS_SYS_ID	BUSINESS_SYS

ACTIV_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	

Col	Column Name	Table with Which to Join
3	ID	
13	PARENT_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_ DEF (zero for root function)
14	CHILD_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_DEF

ACTN_BLK_USE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	ACTN_BLK_ID	■ ACTION_BLOCK or ■ BAA_ACTN_BLK or ■ BSD_ACTN_BLK ■ DERIVATION_ALGOR
6	USED_ACTN_BLK_ID	■ ACTION_BLOCK or ■ BAA_ACTN_BLK or ■ BSD_ACTN_BLK ■ DERIVATION_ALGOR

ADD_ATOM_DEP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	

Col	Column Name	Table with Which to Join
5	ADD_DEPENDNCY_ID	ADD_DEPENDNCY
6	EXPORT_USAGE_ID	■ ACTIV_USAGE ■ XT_OBJ_USAGE ■ XT_EVNT_USAGE ■ ADD_PARALLEL ■ ADD_MUTL_EXCL ■ ADD_CLOSURE
7	IMPORT_USAGE_ID	■ ACTIV_USAGE ■ XT_OBJ_USAGE ■ ADD_PARALLEL ■ ADD_MUTL_EXCL ■ ADD_CLOSURE

ADD_CLOSURE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_DEF
6	ADD_MUTL_EXCL_ID	ADD_MUTL_EXCL

ADD_EXT_FLOW

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	

Col	Column Name	Table with Which to Join
3	ADD_DEPENDNCY_ID	ADD_DEPENDNCY
4	DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW

ADD_MUTL_EXCL

See also the PDD_MUTL_EXCL table.

Col	Column Name	Table with Which to Join
3	ID	
5	PARENT_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_DEF

ADD_PARALLEL

See also the PDD_PARALLEL table.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_DEF

ALIAS

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	

Col	Column Name	Table with Which to Join
9	DATA_ITEM_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP ■ ATTRIBUTE

ATTR_IDENT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP
7	ATTRIBUTE_ID	ATTRIBUTE

ATTR_VIEW

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	ENTITY_VIEW_ID	ENTITY_VIEW
8	ATTRIBUTE_ID	ATTRIBUTE

ATTRIBUTE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
14	PARENT_ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP ■ SYS_ENT_TYPE
15	ACTION_BLOCK_ID	■ ACTION_BLOCK ■ DERIVATION_ALGOR
16	DEF_PERM_VAL_ID	PERMIT_VALUE

BAA_ACTN_BLK

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	BUSINESS_SYS_ID	BUSINESS_SYS

BSD_ACTN_BLK

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	

Col	Column Name	Table with Which to Join
7	BUSINESS_SYS_ID	BUSINESS_SYS

BUS_GOAL

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
11	PARENT_GOAL_ID	BUS_GOAL (in future release)

BUS_OBJECTIVE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	PARENT_BUSOBJ_ID	BUS_OBJECTIVE (in future release)

BUS_PROC_SCOPE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	BUSINESS_PROC_ID	BUSINESS_PROC
4	ELEM_PROCESS_ID	PROCESS_DEF

BUS_PROC_STEP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	BUS_PROCEDURE_ID	BUSINESS_PROC
11	ACTION_BLOCK_ID	■ ACTION_BLOCK or ■ BAA_ACTN_BLK or ■ BSD_ACTN_BLK

BUS_SYS_SCOPE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	BUSINESS_SYS_ID	BUSINESS_SYS
4	ELEM_PROCESS_ID	PROCESS_DEF

BUSINESS_PROC

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	BUSINESS_SYS_ID	BUSINESS_SYS

CD_ACTN_BLK

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	ENTITY_ID	ENTITY_TYPE
7	LINKAGE_ID	LINKAGE

CELL_VALUE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	MATRIX_ID	MATRIX
7	X_OBJECT_ID	Table that corresponds to the class on the X axis of the matrix. ■ For a user-defined class, this will be table USER_DEF_OBJECT. ■ For a system-defined class, the table name is stored in PI_NAME of the OBJECT_CLASS table (depends on the class).
8	Y_OBJECT_ID	Table that corresponds to the class on the Y axis of the matrix. ■ For a user-defined class, this will be table USER_DEF_OBJECT. ■ For a system-defined class, the table name is stored in the PI_NAME column of the OBJECT_CLASS table.

CHANGE_OCCUR

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
4	SESSION_ID	SESSION
5	OBJECT_ID	Table that corresponds to a object that can be associated to a session

CLASSIFIER

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARTITIONING_ID	PARTITIONING
6	ATTRIBUTE_ID	ATTRIBUTE

CMD_SYNONYM

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	COMMAND_ID	COMMAND

COMMAND

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	BUSINESS_SYS_ID	BUSINESS_SYS

COMPONENT_IMPLEM

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
4	ORG_ID	

COMPONENT_SPEC

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
4	ORG_ID	

CRITICAL_SUCCESS

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	PARENT_CSF_ID	CRITICAL_SUCCESS (in future release)

CSTM_EDT_PTRN

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	SCRN_FLD_VAR_ID	SCRN_FLD_VAR

CURRENT_DATA

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	PARENT_DATA_ID	CURRENT_DATA (future release)

CURRENT_EFFECT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	MATRIX_ID	MATRIX
10	CUR_DATA_ID	CURRENT_DATA
11	CUR_INFO_SYS_ID	CURRENT_INFO_SYS

CURRENT_INFO_SYS

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	PARENT_SYS_ID	CURRENT_INFO_SYS (in future release)

DASD_VOL_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	DATASET_ID	■ DATASET_TBLSP ■ DATASET_INDEX
6	DASD_VOLUME_ID	DASD_VOLUME

DATA_BASE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	DEF_STG_GRP_ID	STORAGE_GROUP

DATA_BASE_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	TECH_DESIGN_ID	TECHNICAL_DESIGN
4	DATABASE_ID	DATA_BASE

DATA_CLUSTER

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	

DATA_STORE_INDEX

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
17	DATABASE_ID	DATA_BASE
18	STORAGE_GRP_ID	STORAGE_GROUP

DATA_STORE_TBLSP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
18	DATABASE_ID	DATA_BASE
19	STORAGE_GRP_ID	STORAGE_GROUP

DATACOM_COLUMN

The following table details various column names along with the tables with which they join.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
11	FIELD_ID	FIELD

DATACOM _CONSTRNT

The following table details various column names along with the tables with which they join.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
9	LINKAGE_ID	LINKAGE

DATACOM _DATABASE

The following table details various column names along with the tables with which they join.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
10	DATABASE_ID	DATA_BASE

DATACOM _INDEX

The following table details various column names along with the tables with which they join.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	

Col	Column Name	Table with Which to Join
4	ORG_ID	
6	ENTRY_POINT_ID	ENTRY_POINT

DATACOM _TABLE

The following table details various column names along with the tables with which they join.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
12	RECORD_ID	RECORD

DATACOM _TD

The following table details various column names along with the tables with which they join.

Note: DATACOM is no longer supported. This view will exist for compatibility reasons.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
5	OWNER_ID	
30	TECH_DSN_ID	TECHNICAL_DESIGN

DATASET_INDEX

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
15	DATA_STORE_ID	DATA_STORE_INDEX
16	STORAGE_GRP_ID	STORAGE_GROUP

DATASET_TBLSP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
15	DATA_STORE_ID	DATA_STORE_TBLSP
16	STORAGE_GRP_ID	STORAGE_GROUP

DB2_DDL_DB

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	DATA_BASE_ID	DATA_BASE

DB2_DDL_INDEX

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	ENTRY_POINT_ID	ENTRY_POINT

DB2_DDL_TABLE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	RECORD_ID	RECORD

DB2_DDL_TBLSP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	DATA_STR_TBSP_ID	DATA_STORE_TBLSP

DB2_DEF_CLUSTER

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	DATA_SET_ID	■ DATASET_INDEX ■ DATASET_TBLSP

DB2_MVS_COLUMN

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
11	FIELD_ID	FIELD

DB2_MVS_CONSTRAINT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
9	LINKAGE_ID	LINKAGE

DB2_MVS_DATABASE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
10	DATABASE_ID	DATA_BASE

DB2_MVS_INDEX

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
6	ENTRY_POINT_ID	ENTRY_POINT

DB2_MVS_INDEXSPC

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
10	INDEXSPACE_ID	DATA_STORE_INDEX

DB2_MVS_TABLE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
12	RECORD_ID	RECORD

DB2_MVS_TABLESPC

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
11	TABLESPACE_ID	DATA_STORE_TBLSP

DB2_MVS_TD

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	
2	MODEL_ID	
4	ORG_ID	
5	OWNER_ID	
30	TECH_DSN_ID	TECHNICAL_DESIGN

DB2_RESRC_MODULE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	IMPL_LOGIC_ID	IMPLEMENT_LOGIC

DERIVATION_ALGOR

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	BUSINESS_SYS_ID	BUSINESS_SYS

DESC

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	Anything with a description

DFLT_EDT_PTRN

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	BUSINESS_SYS_ID	BUSINESS_SYS

DFLT_LIT_VDAT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	BUSINESS_SYS_ID	BUSINESS_SYS

DFLT_PRM_VDAT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	BUSINESS_SYS_ID	BUSINESS_SYS

DFLT_VAR_VDAT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
13	BUSINESS_SYS_ID	BUSINESS_SYS

DFLT_VARE_VDAT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
10	BUSINESS_SYS_ID	BUSINESS_SYS

DIALECT_TEXT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	DIALECT_ID	DIALECT

Col	Column Name	Table with Which to Join
8	OBJECT_ID	■ PROMPT ■ SCRN_FLD_LIT ■ COMMAND ■ CMD_SYNONYM ■ DFLT_EDT_PTRN ■ CSTM_EDT_PTRN ■ MESSAGE ■ SCROLL_AMOUNT

DIALOG_FLOW

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
10	INITBY_P_STEP_ID	BUS_PROC_STEP
11	INITS_P_STEP_ID	BUS_PROC_STEP

DLG_DATA_SENT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	DIALOG_FLOW_ID	DIALOG_FLOW
4	SRC_DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW
5	DST_DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW

DLG_FLWS_EXST

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
4	DIALOG_FLOW_ID	DIALOG_FLOW
5	EXIT_STATE_ID	EXIT_STATE

DLG_SETS_CMD

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	DIALOG_FLOW_ID	DIALOG_FLOW
4	COMMAND_ID	COMMAND

ENTITY_ST_TRANS

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	SOURCE_ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP
6	DEST_ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP

ENT_ST_TRANS_USE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	ENT_ST_TRANS_ID	ENTITY_ST_TRANS
6	ACTIVITY_ID	■ FUNCTION_DEF ■ PROCESS_DEF

ENTITY_REC_IMPL

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP
6	RECORD_ID	RECORD

ENTITY_SUBTYP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
12	PARTITIONING_ID	PARTITIONING

Col	Column Name	Table with Which to Join
13	PERMIT_VALUE_ID	PERMIT_VALUE

ENTITY_TYPE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
13	SUBJECT_AREA_ID	SUBJECT_AREA

ENTITY_VIEW

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	PARENT_ID	■ VIEW_SET ■ GROUP_VIEW
10	VIEW_SET_ID	VIEW_SET
11	ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP

ENTRY_POINT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
10	DATA_STR_NDX_ID	DATA_STORE_INDEX
11		■ IDENTIFIER or ■ ATTR_IDENT or ■ REL_IDENT
12	REL_IMPL_ID	REL_PART_IMPL

EXIT_STATE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	BUSINESS_SYS_ID	BUSINESS_SYS
8	MESSAGE_ID	MESSAGE

EXIT_STATE_US

The following table details various column names along with the tables with which they join.

Col	Column Names	Table with Which to Join
1	MODEL_ID	
3	ID	

Col	Column Names	Table with Which to Join
6	EXIT_STATE_ID	EXIT_STATE
7	DIALOG_FLOW_ID	DIALOG_FLOW
8	COMMAND_ID	COMMAND

EXPECT_EFFECT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	IDs	
9	ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP
10	ACTIVITY_ID	■ FUNCTION_DEF ■ PROCESS_DEF

FIELD

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
15	RECORD_IS_IN_ID	RECORD
16	ATTRIBUTE_ID	ATTRIBUTE
17	REL_DENORM_ID	RELATIONSHIP

FLD_ENTPT_VALUE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	FLD_ENTPT_US_ID	FLD_ENTRY_PT_USE
9	DATASET_TBSP_ID	DATASET_TBSP_ID
10	DATASET_INDX_ID	DATASET_INDX_ID

FLD_ENTRY_PT_USE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	FIELD_ID	FIELD
8	ENTRY_POINT_ID	ENTRY_POINT

FLD_LINK_USE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	FIELD_FROM_ID	FIELD
6	FIELD_TO_ID	FIELD

Col	Column Name	Table with Which to Join
7	LINKAGE_ID	LINKAGE

GROUP_VIEW

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
15	PARENT_ID	■ VIEW_SET ■ GROUP_VIEW
16	VIEW_SET_ID	VIEW_SET

IDENTIFIER

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP
8	ATTR_OR_REL_ID	■ ATTRIBUTE ■ RELATIONSHIP

IMPL_LOGIC_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	IMPL_LOGIC_ID	IMPLEMENT_LOGIC
6	CALLED_IMPL_ID	IMPLEMENT_LOGIC

IMPLEMENT_LOGIC

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
14	ACTION_BLOCK_ID	■ ACTION_BLOCK ■ TD_ACTN_BLOCK

INTERFACE_TYPE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
4	ORG_ID	
16	SUBJECT_AREA_ID	
17	INTFCE_TYP_MDL_ID	

INTRFCE_TYPE_MDL

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
4	ORG_ID	

LIB_USAGE_SCOPE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	LIBRARY_USAGE_ID	LIBRARY_USAGE
4	LIBRARY_ID	LIBRARY

LIBRARY_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	TECHSYS_ID	TECHNICAL_SYSTEM

LINKAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	RECORD_FROM_ID	RECORD
9	RECORD_TO_ID	RECORD
10	IDENTIFIER_ID	IDENTIFIER
11	IMPLEMENTATON_ID	REL_PART_IMPL

LINK_DATA_RTND

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	DIALOG_FLOW_ID	DIALOG_FLOW
4	SRC_DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW
5	DST_DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW

LNK_RTNS_CMD

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	

Col	Column Name	Table with Which to Join
3	DIALOG_FLOW_ID	DIALOG_FLOW
4	COMMAND_ID	COMMAND

LNK_RTNS_EXST

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
4	DIALOG_FLOW_ID	DIALOG_FLOW
5	EXIT_STATE_ID	EXIT_STATE

LOCAL_PF_KEY

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	COMMAND_ID	COMMAND
7	BUS_PROC_STEP_ID	BUS_PROC_STEP
8	SYSTEM_PF_KEY_ID	SYSTEM_PF_KEY

MATRIX

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	

Col	Column Name	Table with Which to Join
3	ID	
10	OBJ_CLASS_X_ID	OBJECT_CLASS
11	OBJ_CLASS_Y_ID	OBJECT_CLASS

MATRIX_USAGE_X

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	MATRIX_ID	MATRIX
7	OBJECT_ID	Table that corresponds to the class on the X axis of the matrix ■ For a user-defined class, this will be table USER_DEF_OBJECT. ■ For a system-defined class, the table name is stored in PI_NAME of the OBJECT_CLASS table (depends on the class).

MATRIX_USAGE_Y

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	MATRIX_ID	MATRIX

Col	Column Name	Table with Which to Join
7	OBJECT_ID	Table that corresponds to the class on the Y axis of the matrix. ■ For a user-defined class, this will be table USER_DEF_OBJECT. ■ For a system-defined class, the table name is stored in PI_NAME of the OBJECT_CLASS table.

ORGANIZAT_UNIT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
11	PARENT_ORG_ID	ORGANIZAT_UNIT
12	PARENT_WILLBE_ID	ORGANIZAT_UNIT (in future release)

PAD_CREATE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ID	■ ACTION_BLOCK or ■ BAA_ACTN_BLK or ■ BSD_ACTN_BLK ■ DERIVATION ALGOR
6	ENTITY_VIEW_ID	ENTITY_VIEW

PAD_DELETE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ID	■ ACTION_BLOCK or ■ BAA_ACTN_BLK or ■ BSD_ACTN_BLK ■ DERIVATION ALGOR
6	ENTITY_VIEW_ID	ENTITY_VIEW

PAD_READ

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ID	■ ACTION_BLOCK or ■ BAA_ACTN_BLK or ■ BSD_ACTN_BLK ■ DERIVATION_ALGOR
6	ENTITY_VIEW_ID	ENTITY_VIEW

PAD_SET_ATTR

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ID	■ ACTION_BLOCK or ■ BAA_ACTN_BLK or ■ BSD_ACTN_BLK ■ DERIVATION ALGOR
6	ATTR_VIEW_ID	ATTR_VIEW

PAD_UPDATE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ID	■ ACTION_BLOCK or ■ BAA_ACTN_BLK or ■ BSD_ACTN_BLK ■ DERIVATION ALGOR
6	ENTITY_VIEW_ID	ENTITY_VIEW

PARM

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	FIELD

PARM_DELIMITER

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	BUS_PROC_STEP_ID	BUS_PROC_STEP
8	BUSINESS_SYS_ID	BUSINESS_SYS

PARM_STRING_DEL

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	BUS_PROC_STEP_ID	BUS_PROC_STEP
9	BUSINESS_SYS_ID	BUSINESS_SYS

PARTITIONING

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	PARENT_ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP

PDD_ATOM_DEP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PDD_DEPENDNCY_ID	PDD_DEPENDNCY
6	EXPORT_USAGE_ID	■ ACTIV_USAGE ■ OBJ_USAGE ■ XT_EVNT_USAGE ■ PDD_PARALLEL ■ PDD_MUTL_EXCL ■ PDD_CLOSURE
7	IMPORT_USAGE_ID	■ ACTIV_USAGE ■ XT_OBJ_USAGE ■ PDD_PARALLEL ■ PDD_MUTL_EXCL ■ PDD_CLOSURE

PDD_CLOSURE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_DEF
6	PDD_MUTL_EXCL_ID	PDD_MUTL_EXCL

PDD_EXT_FLOW

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	PDD_DEPENDNCY_ID	PDD_DEPENDNCY
4	DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW

PDD_MUTL_EXCL

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_DEF

PDD_PARALLEL

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	PARENT_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_DEF

PERMIT_VALUE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	■ PERMIT_VALUE_HI ■ PERMIT_VALUE_LOW
5	LOW_VALUE	PERMIT_VALUE_LOW (will be deleted in future release)
6	HIGH_VALUE	PERMIT_VALUE_HI (will be deleted in future release)
7	ATTRIBUTE_ID	ATTRIBUTE

PERMIT_VALUE_HI

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	PERMIT_VALUE

PERMIT_VALUE_LOW

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	PERMIT_VALUE

PROCESS_DEF

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	ACTION_BLOCK_ID	■ ACTION_BLOCK or ■ BAA_ACTN_BLK

PROMPT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	ATTRIBUTE_ID	ATTRIBUTE

REC_ENTRY_PT_USE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	RECORD_ID	RECORD
8	ENTRY_POINT_ID	ENTRY_POINT

RECORD

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
14	DATA_STORE_ID	DATA_STORE_TBLSP

RECORD_REFERENCE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	RECORD_ID	RECORD
6	IMPL_LOGIC_ID	IMPLEMENT_LOGIC

REL_IDENT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP
7	RELATIONSHIP_ID	RELATIONSHIP

REL_MUTL_EXCL

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP
6	RELATIONSHIP_ID	RELATIONSHIP

REL_PART_IMPL

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	REL_PART_ID	RELATIONSHIP

REL_VIEW

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	ENTITY_VIEW_ID	ENTITY_VIEW
8	RELATIONSHIP_ID	RELATIONSHIP

RELATIONSHIP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
18	SOURCE_ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP
19	DEST_ENTITY_ID	■ ENTITY_TYPE ■ ENTITY_SUBTYP
20	INVERSE_REL_ID	RELATIONSHIP

SCREEN_DEF

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	

Col	Column Name	Table with Which to Join
10	BUS_PROC_STEP_ID	BUS_PROC_STEP

SCREEN_TMPLT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	BUSINESS_SYS_ID	BUSINESS_SYS

SCRN_FLD_LIT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
12	SCREEN_DEF_ID	■ SCREEN_DEF ■ SCREEN_TMPLT

SCRN_FLD_PRMT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
12	PROMPT_ID	PROMPT

Col	Column Name	Table with Which to Join
13	SCREEN_DEF_ID	■ SCREEN_DEF ■ SCREEN_TMPLT
14	SCRN_FLD_VAR_ID	SCRN_FLD_VAR

SCRN_FLD_VAR

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	DFLT_EDT_PTRN_ID	DFLT_EDT_PTRN
9	SCRN_RG_OCC_ID	SCRN_RG_OCC
10	SCRN_VAR_DEF_ID	SCRN_VAR_DEF
11	SCREEN_DEF_ID	■ SCREEN_DEF ■ SCRN_TMPLT

SCRN_FLD_VARE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	SCRN_FLD_VAR

SCRN_FLD_VARP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	SCRN_FLD_VAR

SCRN_HELP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	■ SCREEN_DEF ■ SCRN_VAR_DEF ■ SCRN_SYS_DEF

SCRN_RG_OCC

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	PARENT_ID	■ SCRN_RP_GRP ■ SCRN_RG_OCC
7	SCREEN_DEF_ID	SCREEN_DEF

SCRN_RP_GRP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	GROUP_VIEW_ID	GROUP_VIEW
8	SCREEN_DEF_ID	SCREEN_DEF

SCRN_SYS_DEF

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
9	SYS_ATTRIBUTE_ID	SYS_ATTRIBUTE
10	SCREEN_DEF_ID	■ SCREEN_DEF ■ SCREEN_TMPLT

SCRN_VAR_DEF

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
10	SCREEN_DEF_ID	SCREEN_DEF

SCRN_VAR_IO

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
4	ATTR_VIEW_ID	ATTR_VIEW
5	SCRN_VAR_DEF_ID	SCRN_VAR_DEF

SPEC_TYPE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
4	ORG_ID	
16	SCREEN_AREA_ID	

STG_DVOL_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	DASD_VOLUME_ID	DASD_VOLUME
4	STORAGE_GRP_ID	STORAGE_GROUP

STORAGE_GROUP

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	TECHDESN_ID	TECHNICAL_DESIGN

STRATEGY

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	PARENT_STRAT_ID	STRATEGY (in future release)

SUBJECT_AREA

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	PARENT_SUBJ_ID	SUBJECT_AREA (zero for root subject area)

SYS_ATTRIBUTE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
12	ENTITY_TYPE_ID	SYS_ENT_TYPE

SYSTEM_PF_KEY

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	COMMAND_ID	COMMAND
8	BUSINESS_SYS_ID	BUSINESS_SYS

TACTIC

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	PARENT_TACTIC_ID	TACTIC (in future release)

TD_LIBRARY_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	LIBRARY_ID	LIBRARY
8	TECH_DESIGN_ID	TECHNICAL_DESIGN

TECHNICAL_DESIGN

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
12	DEF_STG_DB_ID	STORAGE_GROUP
13	DEF_STG_TBLSP_ID	STORAGE_GROUP
14	DEF_STG_INDEX_ID	STORAGE_GROUP

TECHNICAL_SYSTEM

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
23	BUSINESS_SYS_ID	BUSINESS_SYS

TEXT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	ID	■ EXIT_STATE ■ ORGANIZAT_UNIT

TMPLT_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	SCREEN_DEF_ID	SCREEN_DEF
6	SCREEN_TMPLT_ID	SCREEN_TMPLT

UNFORMAT_INPUT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
7	ATTR_VIEW_ID	ATTR_VIEW
8	SYS_ATTRIBUTE_ID	SYS_ATTRIBUTE
9	BUS_PROC_STEP_ID	BUS_PROC_STEP

UNFRMT_INP_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	PROMPT_ID	PROMPT
4	UNFORMAT_INP_ID	UNFORMAT_INPUT

USE_DATA_RTND

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ACTN_BLK_USE_ID	ACTN_BLK_USE
4	SRC_DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW
5	DST_DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW

USE_DATA_SENT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ACTN_BLK_USE_ID	ACTN_BLK_USE
4	SRC_DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW

Col	Column Name	Table with Which to Join
5	DST_DATA_VIEW_ID	■ GROUP_VIEW ■ ENTITY_VIEW

USER_DEF_OBJECT

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
8	OBJECT_CLASS_ID	OBJECT_CLASS
9	PARENT_OBJECT_ID	USER_DEF_OBJECT (in future release)

VIEW_SET

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
5	ACTIVITY_ID	■ FUNCTION_DEF ■ PROCESS_DEF ■ BUS_PROC_STEP ■ ACTION_BLOCK or ■ BSD_ACTN_BLK ■ DERIVATION_ALGOR ■ PAD_FUNCTION

XT_EVNT_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	PARENT_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_DEF
7	XTERNL_EVENT_ID	XTERNL_EVENT

XT_IMPL_LOGIC

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
10	OK_SESSION_ID	SESSION
11	ACTION_BLOCK	ACTION_BLOCK

XT_OBJ_USAGE

The following table details various column names along with the tables with which they join.

Col	Column Name	Table with Which to Join
1	MODEL_ID	
3	ID	
6	PARENT_ACTIV_ID	■ FUNCTION_DEF ■ PROCESS_DEF

Col	Column Name	Table with Which to Join
7	XTERNL_OBJECT_ID	XTERNL_OBJECT

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

O

P

Q

R

S

T

U

V

W

X