Progress®



OpenEdge® Management and OpenEdge Explorer:
Managing Table Partitioning in Databases

OpenEdge®

Copyright

© **2019 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.**

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

Corticon, DataDirect (and design), DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Deliver More Than Expected, Icenium, Kendo UI, Kinvey, NativeScript, OpenEdge, Powered by Progress, Progress, Progress Software Developers Network, Rollbase, SequeLink, Sitefinity (and Design), Sitefinity, SpeedScript, Stylus Studio, TeamPulse, Telerik, Telerik (and Design), Test Studio, and WebSpeed are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries. Analytics360, AppServer, BusinessEdge, DataDirect Autonomous REST Connector, DataDirect Spy, SupportLink, DevCraft, Fiddler, JustAssembly, JustDecompile, JustMock, Kinvey Chat, NativeChat, NativeScript Sidekick, OpenAccess, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, and WebClient are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

February 2019

Last updated with new content: Release 12.0.0

Updated: 2019/03/01

Table of Contents

Preface....7
Purpose....7
Audience....7
Organization....8
Using ABL documentation....8
References to ABL compiler and run-time features....8
References to ABL data types....8
Typographical conventions....9
Examples of syntax descriptions....10
Long syntax descriptions split across lines....11
Complex syntax descriptions with both required and optional elements....12
OpenEdge messages....12
Obtaining more information about OpenEdge messages....13

Chapter 1: Managing Table Partitioning in a Database....15
Enabling a database for table partitioning....16
Creating a table partition....16
Table partition policy....16
Allocating table partitions....16
Creating a table partition policy....17
Viewing and editing a table partition policy....20
Managing table partition policy details....21
Creating table partition policy details....21
Editing table partition policy details....23

Index....25

Preface

For details, see the following topics:

- Purpose
- Audience
- Organization
- Using ABL documentation
- Typographical conventions
- Examples of syntax descriptions
- OpenEdge messages

Purpose

This manual describes how you can use the Database Administration Console in OpenEdge Management and OpenEdge Explorer to manage table partitioning in databases. You can enable databases for table partitioning, create a table partition, and manage table partition policy details.

Audience

This manual is intended for OpenEdge® Management and OpenEdge Explorer users as well as OpenEdge Management database and system administrators.

Organization

Managing Table Partitioning in a Database on page 15

Describes how to enable databases for table partitioning and to create and manage partition policies and partition policy details.

Using ABL documentation

OpenEdge provides a special purpose programming language for building business applications. In the documentation, the formal name for this language is *ABL (Advanced Business Language)*. With few exceptions, all keywords of the language appear in all `UPPERCASE`, using a font that is appropriate to the context. All other alphabetic language content appears in mixed case.

For the latest documentation updates go to https://docs.progress.com.

References to ABL compiler and run-time features

ABL is both a compiled and an interpreted language that executes in a run-time engine. The documentation refers to this run-time engine as the *ABL Virtual Machine (AVM)*. When the documentation refers to ABL source code compilation, it specifies *ABL* or *the compiler* as the actor that manages compile-time features of the language. When the documentation refers to run-time behavior in an executing ABL program, it specifies *the AVM* as the actor that manages the specified run-time behavior in the program.

For example, these sentences refer to the ABL compiler's allowance for parameter passing and the AVM's possible response to that parameter passing at run time: "ABL allows you to pass a dynamic temp-table handle as a static temp-table parameter of a method. However, if at run time the passed dynamic temp-table schema does not match the schema of the static temp-table parameter, the AVM raises an error." The following sentence refers to run-time actions that the AVM can perform using a particular ABL feature: "The ABL socket object handle allows the AVM to connect with other ABL and non-ABL sessions using TCP/IP sockets."

References to ABL data types

ABL provides built-in data types, built-in class data types, and user-defined class data types. References to built-in data types follow these rules:

- Like most other keywords, references to specific built-in data types appear in all `UPPERCASE`, using a font that is appropriate to the context. No uppercase reference ever includes or implies any data type other than itself.
- Wherever *integer* appears, this is a reference to the `INTEGER` or `INT64` data type.
- Wherever *character* appears, this is a reference to the `CHARACTER`, `LONGCHAR`, or `CLOB` data type.
- Wherever *decimal* appears, this is a reference to the `DECIMAL` data type.
- Wherever *numeric* appears, this is a reference to the `INTEGER`, `INT64`, or `DECIMAL` data type.

References to built-in class data types appear in mixed case with initial caps, for example, `Progress.Lang.Object`. References to user-defined class data types appear in mixed case, as specified for a given application example.

Typographical conventions

This documentation uses the following typographical and syntax conventions:

Convention	Description
Bold	Bold typeface indicates commands or characters the user types, provides emphasis, or the names of user interface elements.
Italic	Italic typeface indicates the title of a document, or signifies new terms.
SMALL, BOLD CAPITAL LETTERS	Small, bold capital letters indicate OpenEdge key functions and generic keyboard keys; for example, **GET** and **CTRL**.
KEY1+KEY2	A plus sign between key names indicates a **simultaneous** key sequence: you press and hold down the first key while pressing the second key. For example, **CTRL+X**.
KEY1 KEY2	A space between key names indicates a **sequential** key sequence: you press and release the first key, then press another key. For example, **ESCAPE H**.
Syntax:	
`Fixed width`	A fixed-width font is used in syntax, code examples, system output, and file names.
`Fixed-width italics`	Fixed-width italics indicate variables in syntax.
`Fixed-width bold`	Fixed-width bold italic indicates variables in syntax with special emphasis.
`UPPERCASE fixed width`	ABL keywords in syntax and code examples are almost always shown in upper case. Although shown in uppercase, you can type ABL keywords in either uppercase or lowercase in a procedure or class.
Period (.) or colon (:)	All statements except `DO`, `FOR`, `FUNCTION`, `PROCEDURE`, and `REPEAT` end with a period. `DO`, `FOR`, `FUNCTION`, `PROCEDURE`, and `REPEAT` statements can end with either a period or a colon.
[ ]	Large brackets indicate the items within them are optional.
[ ]	Small brackets are part of ABL.
{ }	Large braces indicate the items within them are required. They are used to simplify complex syntax diagrams.
{ }	Small braces are part of ABL. For example, a called external procedure must use braces when referencing arguments passed by a calling procedure.

Convention	Description
|	A vertical bar indicates a choice.
. . .	Ellipses indicate repetition: you can choose one or more of the preceding items.

Examples of syntax descriptions

In this example, ACCUM is a keyword, and *aggregate* and *expression* are variables:

Syntax

```
ACCUM aggregate expression
```

FOR is one of the statements that can end with either a period or a colon, as in this example:

```
FOR EACH Customer NO-LOCK:
  DISPLAY Customer.Name.
END.
```

In this example, STREAM *stream*, UNLESS-HIDDEN, and NO-ERROR are optional:

Syntax

```
DISPLAY [ STREAM stream ] [ UNLESS-HIDDEN ] [ NO-ERROR ]
```

In this example, the outer (small) brackets are part of the language, and the inner (large) brackets denote an optional item:

Syntax

```
INITIAL [ constant [ , constant ] ]
```

A called external procedure must use braces when referencing compile-time arguments passed by a calling procedure, as shown in this example:

Syntax

```
{ &argument-name }
```

In this example, EACH, FIRST, and LAST are optional, but you can choose only one of them:

Syntax

```
PRESELECT [ EACH | FIRST | LAST ] record-phrase
```

In this example, you must include two expressions, and optionally you can include more. Multiple expressions are separated by commas:

Syntax

```
MAXIMUM ( expression , expression [ , expression ] ... )
```

In this example, you must specify MESSAGE and at least one *expression* or SKIP [(*n*)], and any number of additional *expression* or SKIP [(*n*)] is allowed:

Syntax

```
MESSAGE { expression | SKIP [ ( n ) ] } ...
```

In this example, you must specify {*include-file*, then optionally any number of *argument* or &*argument-name* = "*argument-value*", and then terminate with }:

Syntax

```
{ include-file
    [ argument | &argument-name = "argument-value" ] ... }
```

Long syntax descriptions split across lines

Some syntax descriptions are too long to fit on one line. When syntax descriptions are split across multiple lines, groups of optional and groups of required items are kept together in the required order.

In this example, WITH is followed by six optional items:

Syntax

```
WITH [ ACCUM max-length ] [ expression DOWN ]
  [ CENTERED ] [ n COLUMNS ] [ SIDE-LABELS ]
  [ STREAM-IO ]
```

Complex syntax descriptions with both required and optional elements

Some syntax descriptions are too complex to distinguish required and optional elements by bracketing only the optional elements. For such syntax, the descriptions include both braces (for required elements) and brackets (for optional elements).

In this example, `ASSIGN` requires either one or more *`field`* entries or one *`record`*. Options available with *`field`* or *`record`* are grouped with braces and brackets:

Syntax

```
ASSIGN   { [ FRAME frame ] { field [ = expression ] }
           [ WHEN expression ] } . . .
       | { record [ EXCEPT field . . . ] }
```

OpenEdge messages

OpenEdge displays several types of messages to inform you of routine and unusual occurrences:

- **Execution messages** inform you of errors encountered while OpenEdge is running a procedure; for example, if OpenEdge cannot find a record with a specified index field value.
- **Compile messages** inform you of errors found while OpenEdge is reading and analyzing a procedure before running it; for example, if a procedure references a table name that is not defined in the database.
- **Startup messages** inform you of unusual conditions detected while OpenEdge is getting ready to execute; for example, if you entered an invalid startup parameter.

After displaying a message, OpenEdge proceeds in one of several ways:

- Continues execution, subject to the error-processing actions that you specify or that are assumed as part of the procedure. This is the most common action taken after execution messages.
- Returns to the Procedure Editor, so you can correct an error in a procedure. This is the usual action taken after compiler messages.
- Halts processing of a procedure and returns immediately to the Procedure Editor. This does not happen often.
- Terminates the current session.

OpenEdge messages end with a message number in parentheses. In this example, the message number is `200`:

```
** Unknown table name table. (200)
```

If you encounter an error that terminates OpenEdge, note the message number before restarting.

Obtaining more information about OpenEdge messages

In Windows platforms, use OpenEdge online help to obtain more information about OpenEdge messages. Many OpenEdge tools include the following Help menu options to provide information about messages:

- Choose **Help** > **Recent Messages** to display detailed descriptions of the most recent OpenEdge message and all other messages returned in the current session.
- Choose **Help** > **Messages** and then type the message number to display a description of a specific OpenEdge message.
- In the Procedure Editor, press the **HELP** key or **F1**.

On UNIX platforms, use the OpenEdge `pro` command to start a single-user mode character OpenEdge client session and view a brief description of a message by providing its number.

To use the pro command to obtain a message description by message number:

1. Start the Procedure Editor:

```
OpenEdge-install-dir/bin/pro
```

2. Press **F3** to access the menu bar, then choose **Help** > **Messages**.
3. Type the message number and press **ENTER**. Details about that message number appear.
4. Press **F4** to close the message, press **F3** to access the Procedure Editor menu, and choose **File** > **Exit**.

1

Managing Table Partitioning in a Database

Table partitioning is a data organization scheme in which data is divided horizontally in a table by column value according to predefined data ranges or data values or a combination of both. Each data partition is stored separately in its database.

Table partitioning physically segregates data between partitions, improving performance, maintenance, and availability of the data by breaking up the partitioned table into smaller manageable segments. You can manage table partitioning and table-partitioning policies through the Database Administration Console.

You can perform table partitioning configuration from the Database Administration Console by:

- **Establishing connections with table-partitioned databases** — You can set up local and remote connections to managed databases with table partitioning. A list of these managed databases automatically appears in OpenEdge Management or OpenEdge Explorer.

  You can also set up unmanaged local or remote database connections. Depending on the type of connection you set up, you can edit, copy, and delete it.

- **Converting a database that is not enabled for partitioning into a database enabled for table partitioning** — You can enable a database to support table partitioning.

- **Creating a table partition policy** — You can create and view a policy that includes details, such as default storage areas and an object-allocation rule, so that you can reuse the policy to create another.

For details, see the following topics:

- Enabling a database for table partitioning
- Creating a table partition
- Managing table partition policy details

Enabling a database for table partitioning

You can enable a database for table partitioning from the Database Administration Console.

To enable a database for table partitioning:

1. Select **Database Administration** > **Database Connections**. The list of database connections appears.
2. Highlight the database connection of your choice from the list of database connections, and click **View**. The database home page appears.
3. Verify the **Table Partitioning** status in the **Database Features** panel.

   Note: You can click the **Sort** arrow to display the features in ascending or descending order. If the status displays a check mark in a green circle and the word **Enable** does not appear in the **Action** column then, the database is already enabled for table partitioning.

4. Click **Enable**. The **Enable Table Partitioning** page appears.
5. Click **Enable table partitioning**. A message confirms that the database is enabled for table partitioning.

   This action enables table partitioning and as a consequence, read-only partitioning also is enabled, by default.
6. Click **OK**.

   Note: You can also use the **Database Features** panel to know the status of the **Read-only Partitions**.

Creating a table partition

This section describes how to create a table partition using a table partition policy.

Table partition policy

A table partition policy defines how a table is partitioned. A table partition policy is a database meta-schema that consists of details such as policy name, the partition type, the columns involved and various attributes, such as the default areas for data, LOB, and index storage associated with the partition. You can define only one partition policy for each table being partitioned.

Allocating table partitions

A table partition in an existing database table is used to partition the table based on the values of specified columns. Each partition consists of a set of table rows that match the values specified while defining the partition policy. Based on the policy, all the data present in each field of each table row is placed into the partition. The row data cannot be split across partitions.

As part of creating a partition policy, you also assign a storage area for each table, index, and LOB field, or you can use the default storage areas as defined in the partition policy. You can also specify if or when table partitions must be allocated, as long as the following criteria exist:

- The table exists in the database and has been enabled for table partitioning.
- The table has not been enabled for multi-tenancy.
- Each column that defines the partition policy must be *indexable*. Partition column names must be indexes; only indexable data can be partitioned. For example, LOBs cannot be indexed so it cannot be part of a partition policy.

Creating a table partition policy

You can create a table partition policy and partition a table using the **Create the Table Partition Policy** wizard.

Before you begin, ensure that you have selected a database connection and the database server is running.

To create a new table partition policy and partition a table:

1. Do one of the following to open the **Create Table Partition Policy** wizard:
   - From the management console, select **Database Administration** > **Data Connections**. The **Connection** page appears. From the list of connections, select the connection in which you want to create the partition policy. Then, in the **Storage Management** section, click **Create partition policy**. The **Create Table Partition Policy** wizard appears.
   - From the management console, select **Database Administration** > **Data Connections**. The **Connection** page appears. From the list of connections, select the connection in which you want to create the partition policy. Then, in the **Storage Management** section, click **Partition policies**. The **Table Partition Policies** page appears.

     Click **New** from the page menu options. The **Create Table Partition Policy** wizard appears.
   - From the management console, select **Database Administration** > **New** > **Partition Policy**. The **Create Table Partition Policy** wizard appears.
2. Provide the following information:
   - **Policy name** — Specify a unique name for the partition policy. The name can contain only alphanumeric characters. It must not contain blank spaces or special characters, except underscore (“_”) and hyphen (“-”).
   - **Description** — Describe the partition policy. **Description** is an optional field.
   - **Database connection name** — Enter a database connection name to which you want to add the new policy.

     You can either enter the database connection name or click **Search** to select a database connection from the list of connections in the **Connection Selection** dialog box. To refine your search you can **Apply Filter** to the connection name based on its **AdminServer** and/or **Category** details.

     By default, the last used database connection name appears in the **Database connection** field.

Note: When you type the name of the database connection, follow the format that OpenEdge Management and OpenEdge Explorer use to define a connection: *`adminserver-name.database-name`*. A database server must be running to connect to the database. You cannot update the database connection name after you create a table partition policy.

- **Table** — Specify the table for which the partition policy must be created. Enter a name or click the **Search** icon to select a table to be partitioned from the **Table Selection** dialog box.

Note: You cannot update the table selection details after you create a table partition policy.

- **Default data area** — Specify the default storage location for the table data. Enter an area name or click the **Search** icon to select an area from the **Area Selection** dialog box.

  In the **Area Selection** dialog box, you can **Apply Filter** to the area name based on its **Records per block** and/or **Cluster size** details.
- **Default index area** — Specify the default storage location for the indexes. Enter an area name or click the **Search** icon to select an area from the **Area Selection** dialog box. To refine your search you can also **Apply Filter** to the area name based on its **Records per block** and/or **Cluster size** details.
- **Default LOB area** — Specify the default storage area for the partition's LOB fields. Enter an area name or click the **Search** icon to select an area from the **Area Selection** dialog box. To refine your search you can also **Apply Filter** to the area name based on its **Records per block** and/or **Cluster size** details.
- **Object allocation rule** — Specify **Set new partitions not to allocate space** to not allocate any space for the new partitions or **Set new partitions to allocate space** to allocate storage space for the new partitions.
- **Read-only composite** — Set this option to set a composite table partition policy as read-only.

3. Click **Next** to specify partition fields and partition aligned indexes for the table partition policy.

Note: The **Next** button is disabled until all the mandatory fields are specified.

4. Define the partition fields in the **Partition fields** area using the following:

Note: You cannot update the field selection details after you create a table partition policy.

- Select the **Has Range** check box if you want to partition the table based on a value range associated with a particular column.

  The column supports the following data types: numeric, character, date, datetime, and datetimetz. For example, a partition can be based on a range of values for a data type, such as a historical (date) range or a salary (numeric) range.
- Click **Add fields from table** to add fields from the table associated with the partitioning. Select fields from the list of fields from the **Field Selection** dialog box.
- Click **Add fields from index** to add fields from the index. Select an index from the list of indexes from the **Index Selection** dialog box.
- Select an existing field and click **Remove Field** to remove the field from the Table Partition Policy.
- Click the **Move field up** icon to move a field above an existing field in the list of fields.
- Click the **Move field down** icon to move a field down.

5. In the **Partition aligned indexes** area, select one or more indexes from the list of indexes as **Local** index.

Note: The indexes in this area are populated based on the partition fields added in the **Partition fields** area. The indexes that are common to the fields in the **Partition field** area are displayed in the **Partition aligned indexes** area. You must have at least one aligned index to create table partition policy. If no indexes are displayed in the **Partition aligned indexes** area, you must add or remove fields added in the Partition fields area or create an index that is aligned with the fields you want to add in the **Partition fields** area.

6. Click **Next** to load partition policy details.

   Note: The **Next** button is disabled until all the mandatory fields are specified.

7. Click **Properties** to set a template for automatically assigning unique names to discovered partition policy details. The **Template Properties** dialog appears. The default template is in the `&{PartitionPolicyName}-&{Number}` format.

   Note: This feature is useful if there is a large number of discovered partition policy details.

8. Select the **Name template** location and template property to change the default template.

   Note: Properties, such as **Values[1]** and **Values[2]**, are partition fields with which the indexes are aligned.

9. Click **Copy** to add the selected template property at the specified **Name template** location. You can also enter a string manually in the **Name template**.
10. Click **OK** to apply the name template changes or click **Cancel** to exit without changing the default name template.

    For example, consider that you create a partition policy, `Salary_Partition`, on the `salary` column that stores all the salary details in the `Employee` table. You can have a partition based on whether the salary is equal to 5000 and another partition based on whether the salary is equal to 6000. The `Values[1]` property is the `EmpID` column. To set a name template as `&{PartitionPolicyName}-&{Number}-&{Values[1]}EMP_ID`:

    1. Select the right-most location in **Name template** and click the **Values[1]** property.
    2. Click **Copy** to add `-&{Values[1]}` to the **Name template**.
    3. Select the right-most location in **Name template** and enter `EMP_ID`.

    The partition policy detail names can be `Salary_Partition-1-103988EMP_ID` and `Salary_Partition-2-102415EMP_ID`.

11. Click **Load Details** for OpenEdge Management or OpenEdge Explorer to discover and load partition policy details to the table partition policy. The names of the discovered policy details are defined based on the partition policy detail name template. After the details load, you can edit them, or commit the existing details.

    Note: If the table does not have any data, you must manually add the partition policy details in the next page of the wizard. Note that even though the partitions are discovered and created, the table data does not migrate to the respective partitions until you perform the **splittarget** operation. Also, the **splittarget** operation can be performed only on the table partition policy that has the **Has range** option selected. For information on **splittarget**, see the *Managing table partition policy details* section.

12. Click **Next** to manually load partition policy details or click **Finish** to accept the default partition policy details loaded in the previous step, or Click **Generate policy program** to generate OpenEdge ABL code (.p) that you can use in OpenEdge ABL programs to create a table partition policy. Note that generating the policy program **only** generates the policy program; you must click **Finish** to commit and add the partition policy to the database.
13. View any of the default partition policy details and perform any of the following:

- Click **Add** and then specify the partition policy detail values, name, allocation, and area details to manually add a partition policy detail. Click **Update** to add the partition policy detail or click **Cancel** to delete the partition policy detail. This adds the policy detail after all the existing policy details.

  Note: You can also select an existing partition policy detail and click **Insert Before** or **Insert After** to manually add a partition policy detail at a required position in the list of partition policy details.

- Double-click any of the existing table partition policy details to edit the **Name/Description**, **Allocation**, and **Areas** values, and then click **Update** to incorporate the recently made edits or click **Cancel** to ignore the recently made edits. For more information on setting table partition policy details, see Creating table partition policy details on page 21.

  Note: Allocation and **Areas** values are editable only if the partition policy details are not already in the allocated state.

- Click **Reset** to ignore all the recently made edits to the partition policy details.
- Select a partition policy detail and click **Delete** to delete that partition policy detail.

14. Click:

- **Finish** to commit and add the table partition policy in the database.
- **Previous** to navigate to a previous page of the wizard and edit any details of the table partition policy.
- **Cancel** to not add this table partition policy in the database.
- **Generate policy program** to generate an OpenEdge ABL code (.p) that you can use in OpenEdge ABL programs to create a table partition policy. Note that generating the policy program **only** generates the policy program; you must click **Finish** to add the partition policy to the database.

Loading a partition policy and partition policy details

You can create a table partition policy and its policy details by loading the partition policy table, `_partition-policy.d`, and partition policy details table, `_partition-policy-details.d`, in your database.

Use the OpenEdge Management or OpenEdge Explorer menu option, **Database Administration** > **Load data and definitions** > **Table Contents (.d files)**, to load tables in your database. For information about dumping and loading tables in a database, see the *OpenEdge Management and OpenEdge Explorer: Getting Started* guide.

Viewing and editing a table partition policy

After you set up a table partition policy, you can view its details in the **Table Partition Policies** page.

To view or edit a table partition policy:

1. Ensure that the database server is running and that you have set up the database connection.
2. Select **Database Administration > Database Connections**. The list of database connections appears in the list frame on the left.
3. Click the database connection whose contents you want to view. The database home page options appear in the details frame on the right.

The frame displays the **Connections** option, which allows you to return to the list of connections if you want to select a different database.

4. In the **Database Connection Details** page on the right, scroll down to the **Storage Management** section and click **Partition policies** to view all the configured partition policies in the database. In the **Table Partition Policies** page that appears, you can view the list of partition policies that appear in the list frame.
5. Click on the required partition policies from the **Partition Policy Name** column. The **Edit Table Partition Policy** page appears.
6. View and edit the table partition policy as per your requirements.

Note: For information about the fields in the table partition policy area, refer to Creating a table partition policy on page 17. The **Has composite partition** setting shows whether the table partition policy has composite partitions.

---

7. Click:

   - **Edit Detail** to edit the partition policy details.

     Note: For information about the fields in the partition policy details area, refer to Editing table partition policy details on page 23.

   - **Commit** to update the partition policy with the new edits.
   - **Generate policy program** to generate OpenEdge ABL code (.p) that you can use in OpenEdge ABL programs to create a table partition policy with all the defined partition policy details. Note that generating the policy program **only** generates the policy program with the newly made edits; you must click **Commit** for the newly made partition policy updates to reflect in the database.

Note: To sort the information in the grid, click a column name to sort the column values in ascending or descending order. If you have a large number of policies, you can also move forward or backward a page, or specify a particular page, in the list of policies by using the **Page** feature at the bottom of the list; or, you can filter the list of partition policies by entering the policy name in the **Search** field.

Managing table partition policy details

When creating a partition policy, you define how records of a table are stored in the table partitions in a database. You can create conditions for the records using the **Partition Policy Details** page. For example, if you create a partition policy, `Salary_Partition`, on the column, `salary`, that stores all the salary details in the table, `Employee`, then, in the **Partition Policy Details** page, you can define that only when the salary is less than or equal to 5000, the record be stored in the partition, `partition1`, and when the salary is more than 5000, it be stored in `partition2`.

Creating table partition policy details

To create partition policy details:

1. Select **Database Administration** > **Data Connections** from the management console. The **Database Administration** page appears.
2. Select a database connection from the list of connections available in the **Connections** panel.
3. Click **Partition policies** in the Storage Management area. The **Table Partition Policies** page appears.
4. Click **Edit Details** in the **Action** column of a partition policy to edit its details. The **Edit Partition Policy Details** page appears.
5. Review the following policy detail options:
   - The **Policy name** setting is set to the name of the table partition policy.
   - The **Table** setting is set to the name of table.
   - The **Policy type** setting is set to one of the following:

     Table 1: Policy type options

Option	Description
range	This option indicates that the policy details specify a range of values.
list	This option indicates that the policy details specify a list of values.
list-range	This option indicates that the policy details allow both a range and a list of values.

   - The **Default allocation** setting is set to one of the following:

     Table 2: Default allocation options

Option	Description
None	This option indicates that the new partitions must not be allocated any space.
Immediate	This option indicates that the new partitions must be allocated storage space.

   - The **Has composite partition** setting shows whether the table partition policy has composite partitions.
   - The **Read-only composite partition** setting shows whether a composite table partition policy is read-only.
6. Click **Add** to add a partition policy detail at the end of the list of existing partition policy details. You can also select a partition policy detail and click **Insert Before** or **Insert After** to add a new partition policy detail before or after the selected partition policy, respectively. A new row for the partition policy detail is added to the list.
7. Enter a unique value in the **Values** column.
8. Enter the **Name** and **Description** of the partition detail in the **Name/Description** column. **Name** is a mandatory field.
9. In the **Allocation** column:
   - Select the **Allocated** option to allocate storage area in the database.

Note: This field is disabled in the following cases:

- If the **Object allocation rule** option is set to **Immediate** while creating the table partition policy.
- If partition policy details are added to a non-empty table before creating a table partition policy.

is set to partition object-allocation rules have been specified earlier while creating the partition policy.

- If the **Composite** option is selected, the table data is in the default partition.

Note: For a non-empty table, the **Composite** option is selected after the partition policy creation.

- Select the **Split-target** option to specify that the table data is ready to migrate from the default partition to the respective partitions based on the partition policy details.

Note: Selecting this option enables the partition policy detail for splitting. The table data must be migrated to the respective partitions using the `PartitionSplitUtility` API. Once the `PartitionSplitUtility` API is executed, the **Split-target** option is disabled to specify that the table values have already been migrated to the respective partition as per the partition policy detail.

- Select the **Read-only** option to set the table partition policy detail as read-only.

Note: Since a composite partition is set at the policy level, if the **Composite** option is selected, the **Read-only** option is also selected and is not editable. You can still set the **Read-only** option for a non-composite table partition policy details.

10. Enter the **Data**, **Index**, and **LOB** values in the **Areas** column. These values are inherited from the parent policy.
11. Click **Partitions** in the **Partitions** column to set partitions for the policy detail. The **Policy Detail Partitions** dialog appears. You can set the following options:
    - Select a partition to view its properties in the **Partition Properties** area.
    - Set the **Area** by clicking the search icon. The **Area Selection** dialog enables you to search for and select the available partition areas.
    - Select the buffer pool in the **Buffer pool** list. The available options are **Primary** (default) and **Alternate**.
12. Click **OK** to apply the policy detail partition changes or **Cancel** to undo the changes.
13. In the **Edit Partition Policy Details** page, click **Update** and then click **Commit** at the bottom of the page to save your created table partition policy detail.

Editing table partition policy details

To edit partition policy details:

1. Select **Database Administration** > **Data Connections** from the management console. The **Database Administration** page appears.
2. Select a database connection from the list of connections available in the **Connections** panel.
3. Click **Partition policies** in the Storage Management area. The **Table Partition Policies** page appears.

4. Click **Edit Details** in the **Action** column of a partition policy to edit its details. The **Edit Partition Policy Details** page appears.
5. Double-click an existing partition policy detail to update the following options:
   - In the **Name/Description** column, you can click the policy detail description to update it.
   - In the **Allocation** column, you can set the **Split-target** and **Read-only** options.

     Note:

     - The **Split-target** option is editable for new table partition policy details. You can set this option for all policy details other than the final policy detail in the list.
     - The **Read-only** option is editable only if the **Allocated** and **Split-target** options are selected. However, you cannot edit the **Read-only** option for a composite partition policy detail since a composite partition is set at the policy level.

   - In the **Default Areas** column, you can set the **Index** and **LOB** values.
   - In the **Partitions** column, you can click **Partitions** to set partitions for the policy detail. The **Policy Detail Partitions** dialog appears.

     Note: The **Partitions** option is available only if the **Split-target** option is set.

     You can set the following options:

     - Select a partition to view its properties in the **Partition Properties** area.
     - Set the **Area** by clicking the search icon. The **Area Selection** dialog enables you to search for and select the available partition areas.

       Note: This field is disabled if the partition policy has been allocated area during partition policy creation.

     - Select the buffer pool in the **Buffer pool** list. The available options are **Primary** (default) and **Alternate**.

6. Click **OK** to apply the policy detail partition changes or **Cancel** to undo the changes.
7. Click **Update** to apply the changes to the partition policy or click **Cancel** to undo the changes.
8. Optionally, you can select a partition policy detail from the list and click **Delete** to delete the partition policy. Click **Yes** in the confirmation dialog.

Note: You can delete a partition policy only if the object-allocation rules are not allocated to the policy.

Index

A

Administration
- creating a partition *16*
- enabling databases *16*
- managing partition policy details *21*
- viewing and editing partition policies *20*

D

Databases
- enabling for table partitioning *16*

P

Partition Policy
- creating *17*

Partition Policy *(continued)*
- editing *20*
- loading *20*
- viewing *20*
- what is *16*

Partition Policy Detail
- creating *21*
- editing *23*
- what is *21*

T

Table Partitions
- allocating *16*
- what is *15*