webMethods Process Performance Manager

DATABASE SYSTEMS

Version 9.10

April 2016

This document applies to PPM Version 9.10 and to all subsequent releases. Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2000 - 2016 Software AG, Darmstadt, Germany and/or Software AG USA Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners. Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at http://softwareag.com/licenses.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at http://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at http://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Contents

1 Text conventions

Menu items, file names, etc. are indicated in texts as follows:

- Menu items, key combinations, dialogs, file names, entries, etc. are displayed in **bold**.
- User-defined entries are shown in **<bold and in angle brackets>**.
- Single-line example texts (e.g., a long directory path that covers several lines) are separated by ↵ at the end of the line.
- File extracts are shown in this font format:

```
This paragraph contains a file extract.
```

2 General

This guide describes the database-related context pertaining to PPM server and the database system used. You should have basic knowledge of database technology and webMethods Process Performance Manager.

PPM uses an SQL RDBMS as the repository in which all configurations and data are saved. PPM has been developed in Java as a client-server application.

This user guide does not provide installation or customizing instructions for any of the database systems supported by webMethods Process Performance Manager. It intends to help you identify and optimize performance-critical scenarios.

3 PPM database schema

PPM uses the database for the persistent storage of all configurations, administrative settings, and imported data. It is the counterpart of the volatile, main-memory based analysis server that contains the analysis structures. In case of data loss in the analysis server, you can completely restore the analysis server from the database. Therefore, for a PPM system backup it is sufficient to back up only the PPM database schema or user.

3.1 Measure and dimension names

The internal names of measures and dimensions are also used for internal allocation of configuration elements in the database. Therefore, these names are subject to the restrictions posed by the underlying database system.

The measure and dimension names displayed on the PPM user interface are specified in the measure configuration (**<description>** XML elements) and depend on the interface language.

Please take the following information into account when assigning internal measure and dimension names:

- Names must begin with a letter.
- Please use capital letters only.
- The only special character allowed is the underscore (no umlauts).
- Assign brief names, avoid long names. The maximum name length allowed depends on the database. We recommend a maximum name length of 25 characters.

Example

The following table shows a few measure and dimension names:

Type	Name	Description
Measure	PNUM	Number of processes
Dimension	D_MATERIAL	Material, two-level
Dimension	TIME	Time dimension

3.2 Database tables

PPM differentiates between database tables with a fixed name and tables whose names are specified by the measure configuration. The following two tables provide a few examples.

CONFIGURATION-BASED TABLE NAMES

Table name	Description
ATTR_INFO	Imported attributes
DBVERSIONNUMBER	Version number of the database schema
XML_CONFIGS	Imported configurations (**runinitdb** or **runppmconfig**)
EPK_IMPORT_TBL	Imported fragments
EPK_TBL	Consolidated process instances

CONFIGURATION-BASED TABLE NAMES

Table name	Description
PC_UMG_DASHBOARD	Imported data of the process instance-independent measure series of Performance Dashboard

3.3 PPM and database interaction

In this chapter, we describe the types of database access that the PPM software performs when importing and analyzing data. To grasp the content of this chapter, you should be familiar with the basic PPM functions and operation.

3.3.1 PPM server access to the database

To access the database, the PPM server uses the standardized JDBC (Java DataBase Connectivity) database interface. Therefore, the database type used is almost completely transparent for the PPM server. However, database manufacturer-specific differences can lead to the PPM server generating different database queries for the same task.

3.3.1.1 Connection to the database

Each database supported by PPM can be accessed via JDBC. The JDBC driver consists of one or multiple Java archives. The relevant Java archive files (jar) must be copied manually to **<installation directory>\ppmmashzone\server\bin\work\data_ppm\drivers**.

DIFFERENT DATABASES IN ONE PPM INSTALLATION

In a PPM installation, you can address database systems of different manufacturers for different clients. The relevant Java archive files (jar) must be copied manually to **<installation directory>\ppmmashzone\server\bin\work\data_ppm\drivers**.

Only one JDBC driver can be specified for a particular database type. It is impossible to address different versions of a database type, e.g., Oracle 9 and Oracle 10 with different JDBC driver versions.

3.3.1.2 Database user

For each PPM client, you need a single schema of a dedicated database user. The database user requires unrestricted access to the objects of their schema.

In the database-specific **<RDBMS>_USE_CASE_SENSITIVE_USERNAME** key of the client-specific **Database_settings.properties** configuration file, you can specify if the database system is to consider case-sensitive spelling of user names (value **true**) or not (value **false**). The default value is **false**.

Warning

If the database system used considers case-sensitive spelling of user names, you must use the same spelling for the database user when you create the PPM client. Subsequently, you must set the key value in the client-specific **Database_settings.properties** configuration file to **true**.

3.3.2 Database objects

Regardless of the database system used, the PPM server needs the following database objects:

- Tables and constraints
- Sequences (if supported by DB system, otherwise the PPM server simulates a similar functionality)
- Indices
- Foreign keys

To keep up the performance of your PPM system, it is recommended that you have database system statistics calculated on a regular basis. The administrator of your database is the best person to take care of this.
If you use Oracle as a database system for PPM, you can calculate the object statistics using the **-genstats** argument of the **runppmimport** command line program during data import. Further information is available in the command line program's help which you call up with **runppmimport -h**. Particularly for newer Oracle versions, we recommend administrative statistics calculation instead of the PPM-controlled calculation.

3.3.2.1 Handling of NULL values

There is a technical difference between the statements **value does not exist** (NULL value) and **value is not specified** (string with the length null). PPM database extractors can differentiate between NULL values and empty strings.

However, the database systems supported by PPM behave differently. When extracting source system data using PPM extractors, they can exhibit different reactions to NULL and "".

ORACLE

Data fields containing empty strings or without content are read as NULL.

SQL SERVER AND DB2

Empty strings are returned when inserting/updating.

Missing columns are returned as NULL for INSERT.

3.3.2.2 Transactions

The PPM server is transaction-controlled on the database. Database queries are performed within a chain of command and finally applied (committed) in the database schema. This procedure is called transaction. Initially, transactions are executed temporarily in the runtime environment of the database system. Transactions that are not yet completed can be canceled (rolled back).

Transaction-controlled access to the database requires a minimum amount of system resources of the database instance.

However, the database systems supported by PPM behave differently in terms of transaction handling. The PPM application server optimally exploits the different transaction handling of the database systems supported.

Examples

- When using an Oracle-based database schema, the PPM administration components display a **Save** button. After saving, any changes apply immediately.
- When using a DB2- or SQL server-based database schema, analysis options are limited by the **READ_LOCK** transaction logic.

3.3.3 Data import

This chapter describes database-related processes when importing process instance data. Data import consists of two consecutive phases:

1. Import of fragments from application systems (Page 7)
2. Creation and calculation of process instances (Page 7)

3.3.3.1 Import fragments

Regardless of the data import format (system event format or graph format), process keys, hierarchy keys and shared fragment keys are calculated for the fragment instances to be imported. It is assumed that the configuration of the relevant rules is valid. If at least one process key was calculated, the imported fragment is saved in the **EPK_IMPORT_TBL** database table. Complete process instances are saved directly, i.e., without calculating a process key.

Complete process instances are considered done. They can no longer be extended by merging them with imported fragments. A complete process instance is identified by the **AT_EPK_KEY** process instance attribute.

3.3.3.2 Create process instances and calculate measures

COPY EPC PHASE

In this step, the imported fragments are exported from the **EPK_IMPORT_TBL** table and written to the **EPK_TBL** table. The calculated values of all process-related keys (process, shared fragment, hierarchy keys) are saved with foreign key relation to the corresponding fragment in certain tables. Exactly to which tables they are saved depends on the state of the database, for example, if process instances already exist in the database. Subsequently, the fragment is deleted from the **EPK_IMPORT_TBL**.

MERGER PHASE

Fragments with identical process key are combined in one process instance. Two fragments are processed for this, i.e., all objects and connections of the fragment with the older import time stamp are copied to the fragment with the more recent import time stamp. This process is repeated until only unique process keys exist in the database.

HIERARCHIES PHASE

The sequence of calculation of hierarchically dependent process instances is specified and the state of the references is updated.

MEASURE CALCULATION PHASE

In this phase, process instances are typified and calculated. Fragments and keys no longer needed in the database are deleted and the calculated process instances are written back into the **EPK_TBL**.

Consolidating fragments into process instances requires the process keys of all imported fragments to be held in the main memory so that they can be processed simultaneously. If you import a large number of fragments in one import process, the main memory size might not be sufficient. Use the **READ_RATE_EPC** key of the **EpkImport_settings.properties** configuration file to set the maximum number of fragment instances to be imported in one process step. Default value = 100000.

3.3.4 Data analysis

Once the processes have been calculated during measure calculation and the resulting measures and dimensions have been transferred to the analysis server, the process instances are ready to be analyzed.

3.4 Tablespaces

All database systems supported by PPM use particular areas on a data carrier for permanent data storage. These areas are called **tablespaces** for Oracle and DB2 databases. Microsoft uses the term **filegroups** for its SQL server products. In the following, we will use tablespace as a uniform term for both.

The PPM system uses different classes of data:

- Imported fragment instances and process instances (binary data objects)
- Administration and structure information
- Database indices

The first step in optimizing the performance of your PPM is assigning individual tablespaces to the various data classes.

3.4.1 Tablespace types

In the client-specific **Database_settings.properties** configuration file, you specify which tablespaces are to save which PPM data.

PPM differentiates between the following types of tablespaces:

Tablespace name	Description
STDBLOB	All database tables with binary data are saved in this tablespace, e.g., the table of calculated process instances.
STDTABLE	Standard tablespace. In this tablespace, all data is saved that uses default data types of the database system.
STDINDEX	In MS SQL Server and Oracle systems, this tablespace is used for storing primary keys and index information.

Tablespace memory requirements strongly depend on the configuration and imported data of the PPM client.

If you do not change the default configuration after creating a client, all data of the PPM client is saved in the default tablespace. The default tablespace is specified upon creation of the database user. This configuration is highly unsuitable for a productive system.

3.4.2 Tablespace configuration

The assignment of database tables to tablespaces is specifically indicated in the **Database_settings.properties** client configuration file. General syntax:

<Database type>_TBLCONF_<Tablespace name> = Value

- Database type specifies one of the database types supported. Valid values: **ORACLE_11, ORACLE_12**, **DB2_9**, **DB2_10**, **SQL_SERVER_2008**, **SQL_SERVER_2012**, and **SQL_Server_2014**.

If you are working with an **SQL Server** database type for a multi-byte client, you need to use one of the additional Unicode variants **SQLSERVER_2008_UNICODE**, **SQLSERVER_2012_UNICODE**, or **SQL_Server_2014_UNICODE**.

The syntax of the tablespace specified by "value" is database-specific, too, and is described in the corresponding subchapters of the chapter on Supported database systems (Page 10).

4 Supported database systems

This chapter describes the administrative differences between the various database systems supported by PPM.

PPM supports the following database systems:

- Oracle 11g and 12c
- IBM DB2 9.5 and DB2 10.5
- Microsoft SQL Server 2008, 2012, and 2014

  All versions in UNICODE as well, if data is to be saved in a multi-byte character set, such as Japanese.

When creating a client, specify the database system to be used. The database system-specific settings are specified in the subsequent configuration dialogs: Host name, port number of the database service, name of the database, and for the PPM client the name of the database user to be used, and the database user password.

The database systems supported by PPM use the term "database" with different semantics:

ORACLE

Oracle links runtime processes and the database in one database instance. In their daily language use, they mostly use the term "database".

IBM DB2 AND MICROSOFT SQL SERVER

With these database systems, the runtime processes of the database instance can manage any number of databases. At least the system database must exist. Microsoft calls the system database of its SQL Server products **master database**.

For **IBM DB** and **Microsoft SQL Server** database systems, we recommend that you supply each PPM client with its own database. This applies particularly to the database users required in a scaled PPM system.

4.1 Oracle

PPM supports the main versions **Oracle 11g and 12c**.

To access the Oracle database, PPM uses the JDBC Thin Interface (type 4). Enter the access parameters for the Oracle database in the **URL** key of the client-specific **Database_settings.properties** configuration file.

The syntax is as follows:

jdbc:oracle:thin:@<host>:<port>:<dbname>

e.g., URL=jdbc:oracle:thin:@pcoracle:1521:orappm

4.1.1 JDBC driver

The JDBC drivers of newer Oracle versions are usually downward compatible. Please refer to the notes provided by the manufacturer. By default, the driver is located in the **jdbc\lib** subdirectory of your Oracle installation.

Warning

The PPM server needs a JDBC driver that implements the JDBC-3 features used. The JDBC drivers contained in the classes*.zip files are obsolete and unsuitable for operation with PPM. Please use the **ojdbc6.jar** archive for Oracle version **12**. The JDBC drivers for Oracle 12 are compatible with Oracle 11.

Copy the driver to <installation directory>\ppmmashzone\server\bin\work\data_ppm\drivers.

When starting, the PPM server outputs the interface type used as well as the exact database version and JDBC driver version to a log file or in the command line.

4.1.2 Create a database user

You can easily create a database user for PPM using suitable administration components, such as **Database control** or **EM Database Express**.

Assign the database user the roles **CONNECT** and **RESOURCE** and ensure that the user has sufficient privileges for accessing tablespaces.

Alternatively, you can create the user with an SQL script. Provided that the **PPMDATA** default tablespace and the **TEMP** tablespace exist, you can use the following script to create database users for PPM:

```
prompt This script generates a (new) user.
prompt when using TEMPORARY memories
prompt Please specify the target database and
prompt connect as "SYSTEM" user.
accept servername prompt 'Target database : '
connect system@&servername


accept newusername prompt 'Name of user     :'
accept newuserpass prompt 'Password of user :'

create user &newusername
  identified by &newuserpass
  default tablespace PPMDATA
  temporary tablespace TEMP;
grant CONNECT to &newusername;
grant RESOURCE to &newusername;
connect &newusername/&newuserpass@&servername

accept ende prompt 'Execution finished <return>'

exit
```

Warning

Please avoid using system tablespaces for a PPM database user.

4.1.3 Export and import a PPM database

All client-specific data and configurations are saved in the schema of the database user you configured. To back up the data, you only need to export the user's schema via the **exp** Oracle command. In the command prompt, enter a command line with the following syntax:

exp <DB user>/<password>@<net service name> file=<file name>

To import a backed up PPM database, enter a command line with the following syntax at the command prompt:

imp <DB user>/<password>@<net service name> file=<file name>

Warning

Before importing the non-empty database schema of an existing database user you need to clear the schema by deleting and recreating the database user.

Example: Export a database

To export the schema of the database user **umg_en** (with the same password) from the **ppm_ppmdbsvr** database instance to the **umg_en.dmp** file, enter the following command line in the prompt:

exp umg_en/umg_en@ppm_ppmdbsvr file=D:\dmp\umg_en.dmp

Example: Import a database

To import the schema again, perform the following steps via a command prompt:

- Delete the existing database schema by executing these commands:

  sqlplus system@ppm_ppmdbsvr

  drop user umg_en cascade

  exit
- Create the database user as described in chapter Create a database user (Page 11).
- Import the schema backup:

  imp umg_en/umg_en@ppm_ppmdbsvr file=D:\dmp\umg_en.dmp

4.1.4 Tablespace configuration

The tablespaces described in the chapter on Tablespaces (Page 8) are configured as follows:

4.1.4.1 Oracle 12c

Tablespace name	Key
Default	ORACLE_12_TBLCONF_STDTABLE
Binary data	ORACLE_12_TBLCONF_STDBLOB
Indices	ORACLE_12_TBLCONF_STDINDEX

The tablespace is specified by the following syntax:

Key = TABLESPACE <name>

Example

Database_settings.properties file extract:

```
ORACLE_12_TBLCONF_STDTABLE=TABLESPACE PPMDATA
ORACLE_12_TBLCONF_STDINDEX=TABLESPACE PPMDATAIDX
ORACLE_12_TBLCONF_STDBLOB=TABLESPACE PPMBLOB
```

4.1.4.2 Oracle 11g

Tablespace name	Key
Default	ORACLE_11_TBLCONF_STDTABLE
Binary data	ORACLE_11_TBLCONF_STDBLOB
Indices	ORACLE_11_TBLCONF_STDINDEX

The tablespace is specified by the following syntax:

Key = TABLESPACE <name>

Example

Database_settings.properties file extract:

```
ORACLE_11_TBLCONF_STDTABLE=TABLESPACE PPMDATA
ORACLE_11_TBLCONF_STDINDEX=TABLESPACE PPMDATAIDX
ORACLE_11_TBLCONF_STDBLOB=TABLESPACE PPMBLOB
```

4.2 IBM DB2

4.2.1 JDBC driver

The PPM server can access a DB2 database via interface type 4. Depending on the database version used, you need corresponding JDBC drivers. The required files are located in the **java** subdirectory of your DB2 installation.

4.2.1.1 DB2 <version>

To access a DB2 9.5 or DB2 10 server, you need the JDBC driver consisting of the following jar files: **db2jcc.jar**, **db2jcc_license_cu.jar**, **db2jcc_javax.jar**. The required files are located in the **java** subdirectory of your DB2 installation.

When starting, the PPM server outputs the exact versions of the database and the JDBC driver. In the log file or on the command line, messages like this are displayed:

```
...Establishing connection between user UMG_EN and jdbc:db2://ppmdbsrv:50001/PPM...
...Database version used: SQL09013.
...JDBC driver used: IBM DB2 JDBC Universal Driver Architecture (3.4.65).
```

4.2.2 Create a database user

DB2 databases use the user management of the operating system. After the installation of the DB2 database software, the **DB2ADMIN** operating system user and the **DB2USERS** group already exist. To generate a database user for PPM, you create a corresponding operating system user and add this user to the **DB2USERS** group. Through the group membership, the new operating system user receives the privileges required for accessing the DB2 database. The minimum database-related system privileges are: **CONNECT**, **CREATETAB**, and **IMPLICITSCHEMA**.

4.2.3 Export and import a PPM database

DB2 does not offer an export or import option for the database schema of individual database users. Therefore, schema-related back-up strategies, such as exporting and importing a PPM database cannot be implemented.

However, you can use the database-related DB2 back-up strategies if you create a separate DB2 database for each PPM client.

4.2.4 Tablespace configuration

For DB2, the tablespaces described in the chapter on Tablespaces (Page 8) are configured as follows:

4.2.4.1 DB2 <version>

Tablespace name	Key
Default	DB2_9_TBLCONF_STDTABLE / DB2_10_TBLCONF_STDTABLE
Binary data	DB2_9_TBLCONF_STDBLOB / DB2_10_TBLCONF_STDBLOB

For DB2, the locations where the table indices are saved must be specified when creating the table. The tablespace is specified by the following syntax:

Key = IN <Tablespace name> INDEX IN <Index tablespace name>

The keyword **DB2_<vVersion>_TBLCONF_STDINDEX** of the configuration file **Database_settings.properties** is not supported for DB2.

Example

File extract Database_settings.properties

```
DB2_9_TBLCONF_STDTABLE=IN PPMDATA INDEX IN PPMINDX
DB2_9_TBLCONF_STDBLOB=IN PPMBLOB INDEX IN PPMINDX LONG IN PPMLOB
```

The mandatory extension **LONG IN** <tablespace name> determines the tablespace in which the binary database objects will be saved, all others are saved in the tablespace identified with the keyword **IN**.

4.3 Microsoft SQL Server

PPM supports various versions of **Microsoft SQL Server**. The versions differ in terms of behavior and administration.

Detailed information on the **Microsoft SQL Server** versions supported is available in the current webMethods system requirements (wM_X_SystemRequirements; X is the current version, e.g., 9.6).

The following steps show an example of how to configure your Microsoft SQL Server for PPM.

4.3.1 Create a database user

Run the **sqlcmd** SQL command line program and connect to the relevant database as database administrator (sa):

```
sqlcmd -d <database name> -U sa
```

After you entered the password, the connection to the database is established.

```
CREATE LOGIN <PPM login name> WITH PASSWORD = '<PPM password>', ↵
                                    DEFAULT_DATABASE = <database name>;
CREATE USER <PPM user name> WITH DEFAULT_SCHEMA = <PPM user name>;
GRANT CREATE SCHEMA, CREATE TABLE TO <PPM user name>;
GO
CREATE SCHEMA <PPM user name> AUTHORIZATION <PPM user name>;
GO
```

If the user account already exists, you can delete it using the following sequence of commands:

```
DROP SCHEMA <schema name>
DROP USER <PPM user name>
DROP LOGIN <PPM login name>
```

Example

You want to create a login and user called **ppmuser** in the existing MS SQL Server database **ppmdb**. Start the command line program:

```
sqlcmd -d ppmdb -U sa
```

Enter the following command sequence:

```
DROP SCHEMA <ppmschema>;
DROP USER ppmuser;
DROP LOGIN ppmuser;
GO
CREATE LOGIN PPMUSER WITH PASSWORD = 'ppmuser', DEFAULT_DATABASE = ppmdb;
CREATE USER PPMUSER WITH DEFAULT_SCHEMA = PPMUSER;
GRANT CREATE SCHEMA, CREATE TABLE TO PPMUSER;
GO
CREATE SCHEMA PPMUSER AUTHORIZATION PPMUSER;
GO
```

The chapter **Create database** (Page 18) describes how to use **Microsoft SQL Server Management Studio** to create a database and database user for PPM.

4.3.2 Export and import a PPM database

The best way to create a backup of your SQL server database is to create an SQL server script first, containing the following:

```
use <database name>;
DBCC SHRINKDATABASE (<schema name>, 10)
EXEC sp_addumpdevice 'disk', '<Alias>', '<Backup file name>';
backup database <Schema name> to <Alias>;
go
```

Example

You want to save the **ppmuser** schema of your **ppmdb** SQL server database to the **D:\sqlserver\backup\ppm.dat** file on the **sqlsvr** SQL server. Create the following SQL server script:

```
use ppmdb;
DBCC SHRINKDATABASE (ppmdb, 10)
EXEC sp_addumpdevice 'disk', 'mydisk', ' D:\sqlserver\backup\ppm.dat ';
backup database ppmdb to mydisk;
go
```

Save the script, for example under **D:\sqlserver\backup_ppm.sql**. Execute this script in the **osql** command line program:

```
osql -S sqlsvr -U sa -P <password> -i D:\sqlserver\backup_ppm.sql
```

You can now archive the backup file you created (**D:\sqlserver\bachup\ppm.dat**) as you prefer, for example by moving it to a directory that is regularly backed up to tape.

4.3.3 Tablespace configuration

For SQL servers, the tablespaces described in the chapter on Tablespaces (Page 8) are configured as follows:

Example: SQL Server 2008

Tablespace name	Key
Default	SQLSERVER_2008_TBLCONF_STDTABLE
Binary data	SQLSERVER_2008_TBLCONF_STDBLOB
Indices	SQLSERVER_2008_TBLCONF_STDINDEX

The tablespace is specified by the following syntax:

Key = ON <Tablespace name>

Example

File extract Database_settings.properties

```
SQLSERVER_2008_TBLCONF_STDTABLE=ON PPMDATA
SQLSERVER_2008_TBLCONF_STDINDEX=ON PPMINDX
SQLSERVER_2008_TBLCONF_STDBLOB=ON PPMBLOB
```

4.3.4 Create a database

This chapter illustrates by example how you use **Microsoft SQL Server Management Studio** to set up an SQL server database for a PPM demo system and create a database user for a PPM client server.

If you want to set up an SQL server database for a productive system, please contact your SQL server system administrator. Detailed information on Microsoft SQL Server Management Studio is available in the relevant product documentation.

The example refers to SQL server version 2012 and Microsoft SQL Server Management Studio 11.0.

CREATE A DATABASE

Start **Microsoft SQL Server Management Studio** and log in with an administrative user, for example by using a Windows system administrator access.

In the pop-up menu of the node **Databases** in the tree displayed in Object Explorer, select **New Database**.

GENERAL PAGE

Specify a name (e.g., PPM) for the new database.

Specify the settings for the database files. To operate the PPM demo databases, a 1 GB container is sufficient.

OPTIONS PAGE

Check the **Collation** setting. We recommend that you use the value **Latin1_General_BIN**. UTF-8 character sets are not supported for MS SQL Server databases.

Check the **Recovery Model** setting. For a demo database, the **simple** option is sufficient.

Click **OK** to exit the window and create a new database.

CREATE A LOGIN

After having created the database, you need to create a login for it. To do so, select the entry **New Login** in the pop-up menu of the **Security/Logins** node in Object Explorer.

In the subsequent window, select **General** under **Select a page**, then enable the **SQL Server authentication** option, and enter the name of the database user, e.g., PPMDEMO (in uppercase letters) as well as the password. Assign the user the relevant default database, e.g., the database you just created, **PPM**.

You will need the assigned password later when creating a PPM client. Disable the **Enforce password policy** option if you want to specify any password.

Now, select the **User Mapping** page in order to generate the default schema automatically upon the first login of the newly created user. In the **Map** column, select the database you want to map a schema to for this login name. The **User** column then shows the login name you just created and the field in the **Default schema** column becomes editable. Specify the required name in this

column. We recommend that you use the login name as the default schema name, e.g., PPMDEMO.

The default schema name must be in upper-case letters, e.g., PPMDEMO.

Entering the name of the default schema and clicking **OK** creates this schema.

CREATE DB USER

Open the required database,e.g., the database **PPM** you just created, and select the entry **New User** under the node **Security/Login** node in the Object Explorer configuration tree.

On the **General** page, enter the DB user name and the login name you just created. The DB user name is recommended to be the same as the login name. In the **Default schema** field, enter the name of the schema that you had assigned to the user when creating the login name.

ADJUST DB PROPERTIES

To be able to use the database schema with the user for PPM, you need to specify certain permissions for the new user and schema. In the **Microsoft SQL Server Management Studio** Object Explorer, select the schema node of the created database, e.g., the node **/Databases/PPM/Security/Schemas/PPMDEMO** of the previously created **PPM** database. Select **Properties** from the pop-up menu.

In the **Properties** window, select the **Permissions** node and click **View Database Permissions** to open the window **Database Properties**. The **Permissions** node is already selected there. Select the newly created user, e.g., PPMDEMO and assign it the permissions **Create Schema** and **Create table** by enabling the check boxes in the **Assign**.

CHECK DATABASE CONNECTION

Start PPM Customizing Toolkit and create a new client. In the **Database settings** dialog, enter the data for the new SQL server database and click Test database connection.

The test result is displayed in a separate window.

If the connection fails, you can click the Info button to display detailed information.

Before starting PPM Customizing Toolkit, make sure that the current driver for the SQL server database is located in the correct directory **<installation directory>\ppmmashzone\server\bin\work\data_ppm\drivers**.

4.3.5 JDBC driver

To access a Microsoft SQL Server you need the required JDBC driver. This driver is available for download on the Microsoft homepage.

When starting, the PPM server outputs the interface type used as well as the exact database version and JDBC driver version.

Detailed information on the JDBC drivers required and Java versions supported is available in the current webMethods system requirements (wM_X_SystemRequirements; X is the current version, e.g., 9.6).

4.3.6 Unicode support

PPM supports multi-byte character sets when using Microsoft SQL Server as a database system.

If you want to use PPM with Microsoft SQL Server Unicode databases, please note the following:

- PPM supports SQL Server databases set up for collation **Latin1_General_BIN**. SQL Server databases with other collations are not supported.
- Assign the collation at the database level.
- If you change collation settings at a later time, existing database content is not changed. For the changed database settings to become effective, you need to reinitialize the database schema of the corresponding PPM client by running the command line program **runinitdb**.

Warning

All imported data is deleted when you initialize the database.

4.3.7 Migration

If you want to migrate your existing SQL server databases to Microsoft SQL Server, please ensure that the collation **Latin1_General_BIN** is set for these databases.

Warning

If you migrate databases to Microsoft SQL Server with collations other than **Latin1_General_BIN** it is not sure that PPM can use the migrated databases without errors.

5 Performance tuning

This chapter provides performance-relevant information pertaining to PPM server software and the database system.

5.1 Overall performance

Runtime efficiency of the PPM server system, and especially the import largely depends on the database system.

Please observe the following performance-related instructions for your PPM system:

- Monitor the runtime configuration of the database instance in frequent intervals and, if required, adjust the configuration if you see room for improvement.
- Frequently update the database statistics.
- After aggregating or deleting a large number of process instances, reorganize the tables and recalculate all database schema indices.
- The **runppmimport** command line tool supports you in recalculating the indices. Indicate the argument **-index new**, e.g.:
- runppmimport -client <client name> -user system -password <password> -index new

If at all possible, do not interrupt this procedure to avoid time-consuming consolidation processes in the database schema.

5.1.1 Hardware-related

The hardware you use directly influences the overall performance of the PPM server system.

- PPM server and database system should be installed on the same computer or connected via a network with sufficient capacity.
- If possible, select a RAID 5 or RAID 10 array-based file system for tablespace file storage.
- You can further increase performance by distributing the tablespace of the PPM tablespace types (see chapter Supported database systems (Page 10)) to physically independent file systems.
- Configure an MS SQL Server database so that transaction logs and database files are saved in physically independent file systems.

5.1.2 Configuration-related

- Register user-defined dimensions and measures at a particular process type or process type group so that these dimensions and measures will be calculated for process instances of this process type/process type group only.

Warning

The standard measures (number of processes and number of functions, time, function) and all data access dimensions must be registered at the process tree root because they are required for internal calculations.

- If possible, use process measures and process dimensions. Function measure queries and function dimension queries require more calculation capacity due to the significantly larger data volume. Normally, function measures are required only if a function exists multiple times in a process instance and if you want to analyze the functions individually.

5.2 Import performance

When importing fragments and calculating process instances, large volumes of data are written into the database system's tablespaces. Therefore, the performance of the file system on which the tablespace files are saved determines the performance of the data import.