ARIS Process Performance Manager

DATENBANKSYSTEME

Version 10.1

Oktober 2017

This document applies to PPM Version 10.1 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.

Inhalt

1 Textkonventionen

Im Text werden Menüelemente, Dateinamen usw. folgendermaßen kenntlich gemacht:

- Menüelemente, Tastenkombinationen, Dialoge, Dateinamen, Eingaben usw. werden **fett** dargestellt.
- Eingaben, über deren Inhalt Sie entscheiden, werden **<fett und in spitzen Klammern>** dargestellt.
- Einzeilige Beispieltexte werden am Zeilenende durch das Zeichen ↵ getrennt, z. B. ein langer Verzeichnispfad, der aus Platzgründen mehrere Zeilen umfasst.
- Dateiauszüge werden in folgendem Schriftformat dargestellt:

  ```
  Dieser Absatz enthält einen Dateiauszug.
  ```

2 Allgemeines

Dieses Handbuch beschreibt die datenbanktechnischen Zusammenhänge zwischen PPM-Server und dem verwendeten Datenbanksystem. Datenbanktechnische Grundlagen sowie die grundsätzliche Arbeitsweise von ARIS Process Performance Manager sollten Ihnen bekannt sein.

Als Repository verwendet PPM ein SQL-RDBMS, das alle Konfigurationen und Daten speichert. PPM ist als Client-Server-Applikation in Java entwickelt.

Das Handbuch ist keine Installations- oder Customizing-Anleitung eines der von ARIS Process Performance Manager unterstützten Datenbanksysteme. Dieses Handbuch will Ihnen Kenntnisse vermitteln, um performance-kritische Szenarien zu erkennen und zu optimieren.

3 Datenbankschema von PPM

Die Datenbank dient PPM zur persistenten Speicherung aller Konfigurationen, administrativen Einstellungen sowie der importierten Daten. Sie bildet das Gegenstück zum flüchtigen, hauptspeicherbasierten Analyseserver, der die Analysestrukturen beinhaltet. Im Falle eines Datenverlustes im Analyseserver, kann dieser aus der Datenbank vollständig rekonstruiert werden. Zur Datensicherung eines PPM-Systems ist es also ausreichend ausschließlich das PPM-Datenbankschema bzw. den Benutzer zu sichern.

3.1 Namen von Kennzahlen und Dimensionen

Die internen Namen von Kennzahlen und Dimensionen werden auch zur internen Zuordnung von Konfigurationselementen in der Datenbank verwendet. Daher unterliegen diese Bezeichner teilweise den Beschränkungen des unterliegenden Datenbanksystems.

Die auf der PPM-Benutzeroberfläche angezeigten Namen von Kennzahlen und Dimensionen werden oberflächensprachabhängig in der Kennzahlenkonfiguration angegeben (XML-Elemente **<description>**).

Bitte beachten Sie beim Vergeben von internen Kennzahlen- und Dimensionsnamen folgende Tipps:

- Namen müssen mit einem Buchstaben beginnen.
- Verwenden Sie ausschließlich Großbuchstaben.
- Als Sonderzeichen ist nur der Unterstrich erlaubt (keine Umlaute).
- Vergeben Sie möglichst kurze Namen, vermeiden Sie überlange Namen. Die maximal zulässige Länge der Namen ist datenbankspezifisch. Wir empfehlen eine maximale Länge von 25 Zeichen.

Beispiel

Die folgende Tabelle zeigt einige Kennzahlen- und Dimensionsnamen:

Typ	Name	Beschreibung
Kennzahl	PNUM	Prozessanzahl
Dimension	D_MATERIAL	Material, zweistufig
Dimension	TIME	Dimension Zeit

3.2 Datenbanktabellen

PPM unterscheidet zwischen Datenbanktabellen mit festem Namen und Tabellen, deren Namen durch die Kennzahlenkonfiguration bestimmt wird. Die beiden folgenden Tabellen zeigen einige Beispiele:

KONFIGURATIONSUNABHÄNGIGE TABELLENNAMEN

Tabellenname	Beschreibung
ATTR_INFO	Importierte Attribute
DBVERSIONNUMBER	Versionsnummer des Datenbankschemas
XML_CONFIGS	eingelesene Konfigurationen (**runinitdb** oder **runppmconfig**)
EPK_IMPORT_TBL	eingelesene Fragmente
EPK_TBL	zusammengeführte Prozessinstanzen

KONFIGURATIONSABHÄNGIGE TABELLENNAMEN

Tabellenname	Beschreibung
PC_UMG_DASHBOARD	Importierte Daten der prozessunabhängigen Kennzahlreihe des Performance Dashboard

3.3 Interaktion PPM und Datenbank

In diesem Kapitel lernen Sie die Art der Zugriffe von PPM auf die verwendete Datenbank bei Datenimport und Analyse kennen. Zum besseren Verständnis sollte Ihnen die grundsätzliche Arbeitsweise von PPM vertraut sein.

3.3.1 Zugriff des PPM-Server auf die Datenbank

Der PPM-Server verwendet zum Zugriff auf die Datenbank die standardisierte Datenbankschnittstelle JDBC (Java DataBase Connectivity). Dadurch ist der verwendete Datenbanktyp für den PPM-Server weitgehend transparent. Datenbankhersteller-spezifische Unterschiede können jedoch dazu führen, dass der PPM-Server für dieselbe Aufgabe verschiedene Datenbankanfragen generiert.

3.3.1.1 Verbindung zur Datenbank

Jede von PPM unterstützte Datenbank lässt sich über JDBC zugreifen. Der JDBC-Treiber besteht aus einem oder mehreren Java-Archiven. Die entsprechenden Java-Archivdateien (jar) müssen manuell in das Verzeichnis
<Installationsverzeichnis>\ppm\server\bin\work\data_ppm\drivers kopiert werden.

VERSCHIEDENE DATENBANKEN IN EINER PPM-INSTALLATION

Sie können in einer PPM-Installation für verschiedene Mandanten auch Datenbanksysteme verschiedener Hersteller ansprechen. Die entsprechenden Java-Archivdateien (jar) müssen manuell in das Verzeichnis
<Installationsverzeichnis>\ppm\server\bin\work\data_ppm\drivers kopiert werden.

Für einen bestimmten Datenbanktyp kann immer nur ein einziger JDBC-Treiber angegeben werden. Es ist nicht möglich, unterschiedliche Versionen eines Datenbanktyps, z. B. Oracle 11 und Oracle 12, mit verschiedenen JDBC-Treiberversionen anzusprechen.

3.3.1.2 Datenbankbenutzer

Für jeden PPM-Mandanten benötigen Sie genau ein Schema eines dedizierten Datenbankbenutzers. Der Datenbankbenutzer benötigt uneingeschränkten Zugriff auf die Objekte seines Schemas.

Im datenbankspezifischen Schlüssel **<RDBMS>_USE_CASE_SENSITIVE_USERNAME** der mandantenspezifischen Konfigurationsdatei **Database_settings.properties** können Sie angeben, ob das verwendetet Datenbanksystem bei der Angabe von Benutzernamen zwischen Groß- und Kleinschreibung unterscheidet (Wert **true**) oder nicht (Wert **false**). Default-Wert ist **false**.

Warnung

Wenn das verwendetet Datenbanksystem bei der Angabe von Benutzern Groß- und Kleinschreibung berücksichtigt, müssen Sie beim Anlegen eines PPM-Mandaten für den Datenbankbenutzer dieselbe Schreibweise angeben, wie dieser erstellt wurde. Anschließend müssen Sie in der mandantenspezifischen Konfigurationsdatei **Database_settings.properties** den Wert des Schlüssels auf **true** setzen.

3.3.2 Datenbankobjekte

Unabhängig vom verwendeten Datenbanksystem benötigt der PPM-Server folgende Datenbankobjekte:

- Tabellen und Constraints
- Sequences (falls vom DB-System unterstützt, anderenfalls simuliert der PPM-Server eine vergleichbare Funktionalität)
- Indizes
- Foreign-keys (Fremdschlüssel)

Um Ihr PPM-System performant zu halten, sollten Sie regelmäßig Objekt- und Systemstatistiken berechnen lassen. Am besten ist es, wenn der Administrator Ihrer Datenbank diese Aufgabe übernimmt.
Falls Sie Oracle als Datenbanksystem für PPM verwenden, können Sie die Objektstatistiken mittels Argument **-genstats** des Kommandozeilenprogramm **runppmimport** während des Datenimports berechnen lassen. Weitere Informationen erhalten Sie in der Hilfe des Kommandozeilenprogramms, die durch den Aufruf **runppmimport -h** angezeigt wird. Insbesondere bei neueren Oracle-Versionen empfehlen wir die administrative Statistikberechnung anstelle der von PPM gesteuerten Berechnung von Statistiken.

3.3.2.1 Behandlung von NULL-Werten

Technisch wird zwischen den Aussagen **Wert ist nicht vorhanden** (NULL-Wert) und **Wert ist nicht angegeben** (Zeichenkette der Länge Null) unterschieden. Die PPM-Datenbankextraktoren können zwischen NULL-Werten und leeren Zeichenketten unterscheiden.

Die von PPM unterstützen Datenbanksysteme verhalten sich diesbezüglich unterschiedlich. Beim Auslesen von Quellsystemdaten mit den PPM Extraktoren können diese auf das Auftreten von NULL und "" unterschiedlich reagieren.

ORACLE

Leerstrings werden beim Einfügen/Aktualisieren und Fehlende Spalten beim INSERT als NULL zurückgelesen.

SQLSERVER UND DB2

Leerstrings werden beim Einfügen/Aktualisieren auch so zurückgelesen.

Fehlende Spalten werden beim INSERT als NULL zurückgelesen.

3.3.2.2 Transaktionen

Der PPM-Server arbeitet auf der Datenbank transaktionsgesteuert. Datenbankanfragen werden innerhalb einer Kommandokette ausgeführt und abschließend im Datenbankschema angewendet (commit). Diese Vorgehensweise heißt Transaktion. Transaktionen werden zunächst temporär in der Laufzeitumgebung des Datenbanksystems ausgeführt. Noch nicht abgeschlossenen Transaktionen können widerrufen werden (rollback).

Der transaktionsgesteuerte Zugriff auf die Datenbank erfordert ein bestimmtes Minimum an System-Ressourcen der Datenbankinstanz.

Die von PPM unterstützten Datenbanksysteme unterscheiden sich im Transaktions-Handling. Der PPM-Applikationsserver nutzt dieses unterschiedliche Transaktions-Handling der unterstützten Datenbanksysteme optimal aus.

Beispiele

- Bei Verwendung eines Oracle basierten Datenbankschemas zeigen die PPM-Administrationskomponenten eine Schaltfläche **Speichern** an. Nach dem Speichern sind die Änderungen sofort wirksam.
- Bei Verwendung eines DB2- oder SQL-Server basierten Datenbankschemas sind die Analysemöglichkeiten durch die Transaktionslogik **READ_LOCK** eingeschränkt.

3.3.3 Datenimport

Dieses Kapitel zeigt die datenbanktechnischen Vorgänge beim Import von Prozessinstanzdaten. Der Datenimport setzt sich aus zwei, nacheinander ablaufenden Phasen:

1. Einlesen von Fragmenten aus Anwendungssystemen (Seite 7)
2. Prozessinstanzen erzeugen und berechnen (Seite 7)

3.3.3.1 Fragmente importieren

Unabhängig vom Format des Datenimports (System-Event- oder Graph-Format) werden für die zu importierenden Fragmentinstanzen zunächst Prozessschlüssel, Hierarchieschlüssel und Shared-Fragment-Schlüssel berechnet, gültige Konfiguration der entsprechenden Regelwerke vorausgesetzt. Konnte mindestens ein Prozessschlüssel berechnet werden, wird das gelesene Fragment in der Datenbanktabelle **EPK_IMPORT_TBL** gespeichert. Vollständige Prozessinstanzen werden direkt, ohne Berechnung eines Prozessschlüssels gespeichert.

Vollständige Prozessinstanzen gelten als abgeschlossen, sie können nicht mehr durch Zusammenführen mit importierten Fragmenten erweitert werden. Eine vollständige Prozessinstanz ist durch das Prozessinstanzattribut **AT_EPK_KEY** gekennzeichnet.

3.3.3.2 Prozessinstanzen erzeugen, Kennzahlen berechnen

PHASE EPK KOPIEREN (COPY EPC)

In diesem Schritt werden die importierten Fragmente aus der Tabelle **EPK_IMPORT_TBL** gelesen und in die Tabelle **EPK_TBL** geschrieben. Die berechneten Werte aller prozessbezogenen Schlüssel (Prozess-, Shared Fragment, Hierarchieschlüssel) werden mit Fremdschlüsselbeziehung auf das entsprechende Fragment in bestimmten Tabellen gespeichert. Welche Tabellen genau das ist, hängt vom Zustand der Datenbank ab, ob z. B. bereits Prozessinstanzen in der Datenbank existieren. Anschließend wird das Fragment in der **EPK_IMPORT_TBL** gelöscht.

PHASE MERGER

Fragmente mit identischem Prozessschlüssel werden in einer Prozessinstanz zusammengefasst. Dabei werden schrittweise immer 2 Fragmente bearbeitet, indem alle Objekte und Kanten des Fragments mit dem älteren Importzeitstempel in das mit jüngerem Importzeitstempel kopiert werden. Dieser Vorgang läuft solange, bis ausschließlich eindeutige Prozessschlüssel in der Datenbank vorhanden sind.

PHASE HIERARCHIEN

Die Reihenfolge der Berechnung hierarchisch abhängiger Prozessinstanzen wird bestimmt, der Zustand der Referenzen wird aktualisiert.

PHASE KENNZAHLENBERECHNUNG

In dieser Phase werden Prozessinstanzen typisiert und berechnet. Nicht mehr benötigte Fragmente und Schlüssel in der Datenbank werden gelöscht und die berechneten Prozessinstanzen in die **EPK_TBL** zurückgeschrieben.

Das Zusammenführen von Fragmenten zu Prozessinstanzen erfordert, dass die Prozessschlüssel aller importierten Fragmente im Hauptspeicher gehalten werden, damit sie gleichzeitig bearbeitet werden können. Wenn Sie in einem Importvorgang eine große Anzahl Fragmente importieren, Größenordnung über 100000, reicht die Größe des Hauptspeichers möglicherweise nicht mehr aus. Im Schlüssel **READ_RATE_EPC** der Konfigurationsdatei **EpkImport_settings.properties** können Sie einstellen, wie viele Fragmentinstanzen maximal in einem Arbeitsschritt eingelesen werden sollen, Vorgabewert ist 100000.

3.3.4 Datenanalyse

Nachdem im Rahmen der Kennzahlenberechnung die Prozesse berechnet und die resultierenden Kennzahlen und Dimensionen an den Analyseserver übertragen wurden, können die Prozessinstanzen analysiert werden.

3.4 Tablespaces

Alle von PPM unterstützten Datenbanksysteme benutzen zur persistenten Datenspeicherung bestimmte Bereiche auf einem Datenträger. Bei Oracle und DB2 Datenbanken werden diese Speicherbereiche **Tablespaces** genannt. Microsoft verwendet für seine SQL-Server Produkte den Begriff **Filegroups**. Im Folgenden wird einheitlich der Begriff Tablespace auch für Filegroups verwendet.

Das PPM-System verwendet verschiedene Klassen von Daten:

- Importierte Fragmentinstanzen und Prozessinstanzen (Binäre Datenobjekte)
- Verwaltungs- und Strukturinformationen
- Datenbankindizes

Sie können die Performance Ihres PPM-Systems in einem ersten Schritt optimieren, indem Sie den verschiedenen Datenklassen eigene Tablespaces zuweisen.

3.4.1 Tablespace-Typen

In der mandantenspezifischen Konfigurationsdatei **Database_settings.properties** geben Sie an, welche Tablespaces welche PPM-Daten speichern.

PPM unterscheidet folgende Typen von Tablespaces:

Tablespace-Name	Beschreibung
STDBLOB	Alle Datenbanktabellen mit Binärdaten werden in diesem Tablespace gespeichert, z. B. die Tabelle der berechneten Prozessinstanzen
STDTABLE	Standard-Tablespace. In diesem Tablespace werden alle Daten gespeichert, die Standarddatentypen des Datenbanksystems verwenden.
STDINDEX	Für MS SQL-Server und Oracle-Systeme wird dieser Tablespace zur Ablage der Primärschlüssel und Indexinformationen benutzt.

Der Speicherbedarf eines Tablespace ist stark abhängig von der Konfiguration und den importierten Daten des PPM-Mandanten.

Wenn Sie die Standardkonfiguration nach Anlegen eines Mandanten nicht verändern, werden alle Daten des PPM-Mandanten im Default-Tablespace gespeichert. Der Default-Tablespace wird beim Anlegen des Datenbankbenutzers bestimmt. Diese Konfiguration ist für ein produktives System äußerst ungeeignet.

3.4.2 Tablespace-Konfiguration

Die Zuordnung von Datenbanktabellen zu Tablespaces wird in der Mandantenkonfigurationsdatei **Database_settings.properties** spezifisch angegeben. Die allgemeine Syntax ist:

<Database type>_TBLCONF_<Tablespace name> = Wert

- Datenbanktyp gibt einen der unterstützten Datenbanktypen an. Zulässige Werte: **ORACLE_11, ORACLE_12**, **DB2_10**, **SQL_SERVER_2012** und **SQL_Server_2014**.

Wenn Sie für einen MultiByte-Mandanten den Datenbanktyp **SQL Server** verwenden, müssen Sie eine der zusätzlich verfügbaren Unicode-Varianten **SQLSERVER_2012_UNICODE** oder **SQL_Server_2014_UNICODE** verwenden.

Die Syntax des mit Wert angegebenen Tablespace ist ebenfalls datenbankspezifisch und ist in den jeweiligen Unterkapitel von Kapitel Unterstützte Datenbanksysteme (Seite 11) beschrieben.

4 Unterstützte Datenbanksysteme

Dieses Kapitel informiert Sie über die administrativen Unterschiede der verschiedenen, von PPM unterstützten Datenbanksysteme.

PPM unterstützt folgende Datenbanksysteme:

- Oracle 11g und 12c
- DB2 10.5
- Microsoft SQL Server 2012 und 2014

  Alle Versionen auch in UNICODE, wenn Daten in einem Multibyte-Zeichensatz gespeichert werden sollen, z. B. Japanisch.

Beim Anlegen eines Mandanten geben Sie das zu verwendende Datenbanksystem an. In anschließenden Konfigurationsdialogen werden die datenbanksystemspezifischen Einstellungen angegeben: Hostname, Port-Nummer des Datenbankdienstes, Name der Datenbank sowie für den PPM-Mandanten den zu verwendenden Datenbankbenutzername und das Datenbankbenutzerkennwort.

Die von PPM unterstützten Datenbanksysteme verwenden den Begriff Datenbank mit unterschiedlicher Semantik:

ORACLE

Oracle verknüpft die Laufzeitprozesse und die Datenbank zu einer Datenbankinstanz. Im täglichen Sprachgebrauch wird fast ausschließlich der Begriff Datenbank verwendet.

IBM DB2 UND MICROSOFT SQL-SERVER

Bei diesen Datenbanksystemen können die Laufzeitprozesse der Datenbankinstanz beliebig viele Datenbanken verwalten. Wenigstens die Systemdatenbank muss vorhanden sein. Microsoft nennt die Systemdatenbank seiner SQL-Serverprodukte **master-Datenbank**.

Für die Datenbanksysteme **IBM DB** und **Microsoft SQL-Server** empfehlen wir, für jeden PPM-Mandanten eine eigene Datenbank zu verwenden. Im Besonderen trifft dies für die benötigten Datenbankbenutzer eines skalierten PPM-Systems zu.

4.1 Oracle

PPM unterstützt die Hauptversionen **Oracle 11g und 12c**.

PPM verwendet zum Zugriff auf die Oracle-Datenbank das JDBC-Thin-Interface (Typ4). In der mandantenspezifischen Konfigurationsdatei **Database_settings.properties** geben Sie im Schlüssel **URL** die Zugriffsparameter für die Oracle-Datenbank an.

Die Syntax ist folgende:

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

z. B. URL=jdbc:oracle:thin:@pcoracle:1521:orappm

4.1.1 JDBC-Treiber

Die JDBC-Treiber neuerer Oracle-Versionen sind in der Regel abwärtskompatibel. Bitte beachten Sie dazu die Angaben des Herstellers. Standardmäßig befindet sich der Treiber im Unterverzeichnis **jdbc\lib** Ihrer Oracle-Installation.

Warnung

Der PPM-Server benötigt einen JDBC-Treiber der verwendete JDBC-3-Features implementiert. Die in den Dateien classes*.zip enthaltenen JDBC-Treiber sind veraltet und für den Betrieb mit PPM ungeeignet. Bitte verwenden Sie das **ojdbc7.jar**-Archiv für Oracle Version **12**. Die JDBC-Treiber für Oracle 12 sind mit Oracle 11 kompatibel.

Kopieren Sie den Treiber nach
<Installationsverzeichnis>\ppm\server\bin\work\data_ppm\drivers.

Beim Starten gibt der PPM-Server den verwendeten Interface-Typ sowie die genauen Versionen der Datenbank und des verwendeten JDBC-Treiber in eine log-Datei oder auf der Kommandozeile aus.

4.1.2 Anlegen eines DB-Benutzers

Sie können einen Datenbankbenutzer für PPM bequem mit Hilfe geeigneter Administrationskomponenten wie **Database Control** oder **EM Database Express** anlegen.

Weisen Sie dem DB-Benutzer die Rollen **CONNECT** und **RESOURCE** zu und achten Sie auf genügend Berechtigungen beim Zugriff auf Tablespaces.

Alternativ können Sie den Benutzer auch mit einem SQL-Skript anlegen. Unter der Voraussetzung, dass Standard-Tablespace **PPMDATA** und temporärer Tablespace **TEMP** existieren, können Sie folgendes Skript zum Anlegen von DB-Benutzers für PPM verwenden:

```
prompt Dieses Skript erzeugt einen (neuen) Benutzer.
prompt unter Verwendung von TEMPORÄREN Speicherbereichen
prompt Bitte geben Sie die Zieldatenbank an und
prompt verbinden Sie sich als Benutzer "SYSTEM".
accept servername prompt 'Zieldatenbank : '
connect system@&servername


accept newusername prompt 'Name des Benutzers    :'
accept newuserpass prompt 'Kennwort des Benutzers :'

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 'Ausfuehrung beendet <return>'

exit
```

Warnung

Vermeiden Sie die Verwendung von System Tablespaces für einen PPM-DB-Benutzer.

4.1.3 Exportieren und Importieren einer PPM-Datenbank

Alle mandantenspezifischen Daten und Konfigurationen werden in dem Schema des konfigurierten Datenbankbenutzers gespeichert. Zur Datensicherung genügt es, das Schema des Benutzers mittels Oracle-Kommando **exp** zu exportieren. Geben Sie in einer Eingabeaufforderung eine Befehlszeile folgender Syntax an:

exp <DB-User>/<Kennwort>@<Net-Service-Name> file=<Dateiname>

Zum Importieren einer zuvor gesicherten PPM-Datenbank geben Sie an der Eingabeaufforderung eine Befehlszeile folgender Syntax an:

imp <DB-User>/<Kennwort>@<Net-Service-Name> file=<Dateiname>

Warnung

Bevor Sie das nicht leere Datenbankschema eines existierenden DB-Benutzers importieren, müssen Sie das Schema durch Löschen und Neuanlegen des DB-Benutzers leeren.

Beispiel DB exportieren

Um das Schema des DB-Benutzers **umg_de** der Demo-Datenbank bei gleich lautendem Kennwort aus der Datenbankinstanz **ppm_ppmdbsvr** in die Datei **umg_de.dmp** zu exportieren, geben Sie in einer Eingabeaufforderung folgende Befehlszeile ein:

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

Beispiel DB importieren

Um das Schema anschließend wieder zu importieren, führen Sie in einer Eingabeaufforderung folgende Schritte aus:

- Löschen des existierenden DB-Schemas durch Ausführen folgender Befehle:

  sqlplus system@ppm_ppmdbsvr

  drop user umg_de cascade

  exit

- Anlegen des DB-Benutzers wie in Kapitel Anlegen eines DB-Benutzers (Seite 12) beschrieben.
- Importieren der Schemasicherung:

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

4.1.4 Tablespace-Konfiguration

Die in Kapitel Tablespaces (Seite 9) beschrieben Tablespaces werden wie folgt konfiguriert:

4.1.4.1 Oracle 12c

Tablespace-Name	Schlüssel
Standard	ORACLE_12_TBLCONF_STDTABLE
Binärdaten	ORACLE_12_TBLCONF_STDBLOB
Indices	ORACLE_12_TBLCONF_STDINDEX

Der Tablespace wird durch folgende Syntax spezifiziert:

Schlüssel = TABLESPACE <Name>

Beispiel

Dateiauszug Database_settings.properties:

```
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	Schlüssel
Standard	ORACLE_11_TBLCONF_STDTABLE
Binärdaten	ORACLE_11_TBLCONF_STDBLOB
Indices	ORACLE_11_TBLCONF_STDINDEX

Der Tablespace wird durch folgende Syntax spezifiziert:

Schlüssel = TABLESPACE <Name>

Beispiel

Dateiauszug Database_settings.properties:

```
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-Treiber

Der PPM-Server kann auf eine DB2-Datenbank über den Interface-Typ 4 zugreifen. Je nach verwendeter Datenbankversion benötigen Sie den entsprechenden JDBC-Treiber. Die benötigten Dateien finden Sie im Unterverzeichnis **java** Ihrer DB2-Installation.

4.2.1.1 DB2 <Version>

Um auf einen DB2 10 Server zuzugreifen, benötigen Sie den JDBC-Treiber, der sich aus folgenden jar-Dateien zusammensetzt: **db2jcc.jar**, **db2jcc_license_cu.jar**, **db2jcc_javax.jar**. Die benötigten Dateien finden Sie im Unterverzeichnis **java** Ihrer DB2-Installation.

Beim Starten gibt der PPM-Server die genauen Versionen der Datenbank und des verwendeten JDBC-Treibers aus. In der log-Datei oder auf der Kommandozeile sehen Sie Meldungen der folgenden Art:

```
...Baue Verbindung von Benutzer UMG_DE mit jdbc:db2://ppmdbsrv:50001/PPM auf...
...Verwendete Datenbankversion: SQL09013.
...Verwendeter JDBC-Treiber: IBM DB2 JDBC Universal Driver Architecture (3.4.65).
```

4.2.2 Anlegen eines DB-Benutzers

DB2-Datenbanken verwenden das Benutzermanagement des Betriebssystems. Nach der Installation der DB2-Datenbank-Software existieren bereits der Betriebssystembenutzer **DB2ADMIN** und die Gruppe **DB2USERS**. Um einen Datenbankbenutzer für PPM zu erzeugen, legen Sie einen entsprechenden Betriebssystembenutzer an und nehmen diesen in die Mitgliedschaft der Gruppe **DB2USERS** auf. Der neue Betriebssystembenutzer erhält die für den Zugriff auf die DB2-Datenbank erforderlichen Berechtigungen durch die Gruppenzugehörigkeit. Die minimalen, datenbanktechnischen Systemberechtigungen sind folgende: **CONNECT**, **CREATETAB** und **IMPLICITSCHEMA**.

4.2.3 Exportieren und Importieren einer PPM-Datenbank

DB2 bietet nicht die Möglichkeit, das Datenbankschema eines einzelnen Datenbankbenutzers zu exportieren oder zu importieren. Schemabezogene Sicherungsstrategien wie das Exportieren und Importieren eines PPM-Datenbankbenutzers sind daher nicht möglich.

Sie können jedoch die datenbankbezogenen Sicherungsstrategien von DB2 nutzen, wenn Sie für jeden PPM-Mandanten eine eigene DB2-Datenbank vorsehen.

4.2.4 Tablespace-Konfiguration

Für DB2 werden die in Kapitel Tablespaces (Seite 9)beschrieben Tablespaces wie folgt konfiguriert:

4.2.4.1 DB2 <Version>

Tablespace-Name	Schlüssel
Standard	DB2_10_TBLCONF_STDTABLE
Binärdaten	DB2_10_TBLCONF_STDBLOB

Bei DB2 müssen die Speicherorte der Tabellenindizes bereits beim Erzeugen der Tabelle angegeben werden. Der Tablespace wird durch folgende Syntax spezifiziert:

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

Das Schlüsselwort **DB2_<vVersion>_TBLCONF_STDINDEX** der Konfigurationsdatei **Database_settings.properties** wird für DB2 nicht unterstützt.

Beispiel

Dateiauszug Database_settings.properties

```
DB2_10_TBLCONF_STDTABLE=IN PPMDATA INDEX IN PPMINDX
DB2_10_TBLCONF_STDBLOB=IN PPMBLOB INDEX IN PPMINDX LONG IN PPMLOB
```

Die obligatorische Erweiterung **LONG IN** <Tablespace-Name> bestimmt den Tablespace, in dem die binären Datenbankobjekte gespeichert werden, die anderen werden in dem mit Schlüsselwort **IN** angegebenen Tablespace gespeichert.

4.3 Microsoft SQL Server

PPM unterstützt verschiedene Versionen von **Microsoft SQL-Server**. Die Versionen unterscheiden sich in ihrem Verhalten und der Administration.

Detaillierte Informationen zu den unterstützten Versionen von **Microsoft SQL Server** erhalten Sie in den aktuellen Software AG-Systemanforderungen.

Folgende Schritte zeigen Ihnen beispielhaft, wie Sie Ihren Microsoft SQL-Server für PPM konfigurieren können.

4.3.1 Anlegen eines DB-Benutzers

Starten Sie das SQL-Kommandozeilenprogramm **sqlcmd** und verbinden sich als Datenbankadministrator (User sa) mit der gewünschten Datenbank:

```
sqlcmd -d <Datenbankname> -U sa
```

Nach Eingabe des Kennwortes wird die Verbindung zur angegebenen Datenbank hergestellt.

```
CREATE LOGIN <PPM Loginname> WITH PASSWORD = '<PPM Kennwort>', ↵
                                    DEFAULT_DATABASE = <Datenbankname>;
CREATE USER <PPM Username> WITH DEFAULT_SCHEMA = <PPM Username>;
GRANT CREATE SCHEMA, CREATE TABLE TO <PPM Username>;
GO
CREATE SCHEMA <PPM Username> AUTHORIZATION <PPM Username>;
GO
```

Wenn das anzulegende Benutzerkonto bereits existiert, können Sie es durch folgende Kommandofolge löschen:

```
DROP SCHEMA <Schemaname>
DROP USER <PPM Username>
DROP LOGIN <PPM Loginname>
```

Beispiel

Sie möchten in der existierenden MS SQL-Server-Datenbank **ppmdb** einen Login und User namens **ppmuser** anlegen. Starten Sie das Kommandozeilenprogramm:

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

Geben Sie folgende Kommandosequenz ein:

```
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
```

Im Kapitel **Datenbank erzeugen** (Seite 18) ist beschrieben, wie Sie mit Hilfe von **Microsoft SQL Server Management Studio** eine Datenbank und Datenbankbenutzer für PPM anlegen.

4.3.2 Exportieren und Importieren einer PPM-Datenbank

Am einfachsten erstellen Sie ein Backup Ihrer SQL-Server Datenbank, indem Sie zunächst ein SQL-Server-Skript erstellen mit folgendem Inhalt:

```
use <Datenbankname>;
DBCC SHRINKDATABASE (<Schemaname>, 10)
EXEC sp_addumpdevice 'disk', '<Alias>', '<Backup-Dateiname>';
backup database <Schemaname> to <Alias>;
go
```

Beispiel

Sie möchten das Schema **ppmuser** Ihrer SQL-Server-Datenbank **ppmdb** auf dem SQL-Server **sqlsvr** in die Datei **D:\sqlserver\backup\ppm.dat** sichern. Erstellen Sie folgendes SQL-Server-Skript:

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

Speichern Sie das Skript, z. B. unter **D:\sqlserver\backup_ppm.sql**. Führen Sie dieses Skript im Kommandozeilenprogramm **osql** aus:

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

Die erzeugte Sicherungsdatei **D:\sqlserver\bachup\ppm.dat** können Sie nun beliebig archivieren, z. B. durch Verschieben in ein Verzeichnis, welches regelmäßig auf Band gesichert wird.

4.3.3 Tablespace-Konfiguration

Für SQL-Server werden die in Kapitel Tablespaces (Seite 9)beschrieben Tablespace wie folgt konfiguriert:

Beispiel: SQL Server 2012

Tablespace-Name	Schlüssel
Standard	SQLSERVER_2012_TBLCONF_STDTABLE
Binärdaten	SQLSERVER_2012_TBLCONF_STDBLOB
Indizes	SQLSERVER_2012_TBLCONF_STDINDEX

Der Tablespace wird durch folgende Syntax spezifiziert:

Schlüssel = ON <Tablespace name>

Beispiel

Dateiauszug Database_settings.properties

```
SQLSERVER_2012_TBLCONF_STDTABLE=ON PPMDATA
SQLSERVER_2012_TBLCONF_STDINDEX=ON PPMINDX
SQLSERVER_2012_TBLCONF_STDBLOB=ON PPMBLOB
```

4.3.4 Datenbank anlegen

Dieses Kapitel zeigt ihnen exemplarisch, wie Sie mit Hilfe von **Microsoft SQL Server Management Studio** eine SQL-Server-Datenbank für ein PPM-Demo-System erzeugen und einen Datenbankbenutzer für einen PPM-Mandantenserver anlegen.

Wenn Sie eine SQL-Server-Datenbank für ein Produktivsystem einrichten möchten, wenden Sie sich bitte an Ihren SQL-Server-Systemadministrator. Detaillierte Informationen zu Microsoft SQL Server Management Studio erhalten Sie in der entsprechenden Produktdokumentation.

Das Beispiel bezieht sich auf SQL-Server Version 2012 und das Microsoft SQL Server Management Studio 11.0.

DATENBANK ANLEGEN

Starten Sie **Microsoft SQL Server Management Studio** und melden Sie sich mit einem administrativen Benutzer an, z. B. durch Verwendung eines Windows-Systemadministratorzugangs.

Wählen Sie den Eintrag **Neue Datenbank** im Kontextmenü des Knotens **Datenbanken** des im Objekt Explorer angezeigten Baums.

SEITE **ALLGEMEIN**

Geben Sie einen Namen (z. B. PPM) für die neue Datenbank an.

Setzen Sie die Einstellungen für die Datenbankdateien. Zum Betrieb der PPM-Demodatenbanken genügt ein Container der Größe 1GB.

SEITE **OPTIONEN**

Überprüfen Sie die Einstellung **Sortierung**. Wir empfehlen den Wert **Latin1_General_BIN**. UTF-8-Zeichensätze werden für MS SQL-Server-Datenbanken nicht unterstützt.

Überprüfen Sie die Einstellung **Wiederherstellungsmodell**. Für eine Demodatenbank genügt die Wahl der Option **Einfach**.

Verlassen Sie das Fenster durch Klicken auf **OK**, damit die neue Datenbank erzeugt werden kann.

ANMELDUNG ANLEGEN

Nach dem Anlegen der Datenbank müssen Sie eine Anmeldung für diese Datenbank anlegen. Im Objekt-Explorer wählen Sie dazu im Kontextmenü des Knotens **Sicherheit / Anmeldungen** den Eintrag **Neue Anmeldung**.

Im folgenden Fenster wählen Sie zunächst die Seite **Allgemein**, wählen Sie oben die Option **SQL Server-Authentifizierung** und geben Sie den Namen des Datenbankbenutzers, z. B. PPMDEMO (in Großbuchstaben), und das Kennwort ein. Weisen Sie dem Benutzer die gewünschte Standarddatenbank zu, z. B. die zuvor erstellte Datenbank **PPM**.

Das vergebene Kennwort benötigen Sie später beim Anlegen eines PPM-Mandanten. Deaktivieren Sie die Option **Kennwortrichtlinie erzwingen**, wenn Sie ein beliebiges Kennwort eingeben möchten.

Wählen Sie jetzt die Seite **Benutzerzuordnung**, um das Standardschema automatisch beim ersten Anmelden des neu angelegten Benutzers zu erzeugen. In der Spalte **Zuordnen** wählen Sie die Datenbank, der Sie für diesen Anmeldenamen ein Schema zuordnen möchten. In der Spalte **Benutzer** wird dann der eben eingerichtete Anmeldename angezeigt und das Feld in der Spalte **Standardschema** wird veränderbar. Geben Sie in dieser Spalte den gewünschten Namen an. Es empfiehlt sich, als Standard-Schemanamen den Anmeldenamen zu verwenden, z. B. PPMDEMO.

Der Standard-Schemanamen muss in Großbuchstaben angegeben werden, z. B. PPMDEMO.

Mit der Eingabe des Namens des Standardschemas und durch Klicken auf **OK** wird dieses Schema angelegt.

DB-BENUTZER ANLEGEN

Öffnen Sie die gewünschte Datenbank, z. B. die zuvor erstellte Datenbank **PPM**, und wählen Sie im Konfigurationsbaum des Objekt-Explorer auf dem Knoten **Sicherheit / Benutzer** den Eintrag **Neuer Benutzer**.

Geben Sie in der Seite **Allgemein** den DB-Benutzernamen und den eben angelegten Anmeldenamen ein. Der DB-Benutzername sollte identisch mit dem Anmeldenamen sein. Geben Sie in das Feld **Standardschema** den Namen des Schemas ein, das Sie beim Anlegen des Anmeldenamens dem Benutzer zugeordnet haben.

DB-EIGENSCHAFTEN ANPASSEN

Um das erzeugte Datenbankschema mit dem angelegten Benutzer für PPM verwenden zu können, müssen Sie abschließend noch bestimmte Berechtigungen des neuen Benutzers und Schemas setzen. Wählen Sie im Objekt-Explorer des **Microsoft SQL Server Management Studio** den Schemaknoten der erzeugten Datenbank, z. B. den Knoten **/Datenbanken/PPM/Sicherheit/Schemas/PPMDEMO** der zuvor erstellten Datenbank **PPM**. Wählen Sie im Kontextmenü den Eintrag **Eigenschaften**.

Im folgenden Fenster **Schemaeigenschaften** wählen Sie den Knoten **Berechtigungen** und öffnen durch Klicken auf **Datenbankberechtigungen anzeigen** das Fenster **Datenbankeigenschaften**. Der Knoten **Berechtigungen** ist dort bereits gewählt. Wählen Sie den neu angelegten Benutzer, z. B. PPMDEMO, und weisen diesem die Berechtigungen **Schema erstellen** und **Tabelle erstellen** zu durch Aktivieren der Kontrollkästchen in der Spalte **Erteilen**.

ÜBERPRÜFEN DER DATENBANKANBINDUNG

Starten Sie PPM Customizing Toolkit und legen einen neuen Mandanten an. Machen Sie im Dialog **Datenbankeinstellungen** die Angaben zur neu erzeugten SQL-Server-Datenbank und klicken Sie anschließend auf **Datenbankanbindung testen**.

Das Ergebnis der Überprüfung der Datenbankanbindung wird in einem eigenen Fenster angezeigt.

Wenn der Verbindungsaufbau misslingt, können Sie sich durch Klicken auf die Info-Schaltfläche ausführliche Informationen anzeigen lassen.

Achten Sie vor dem Start von PPM Customizing Toolkit darauf, dass sich der aktuelle Treiber für die SQL-Server-Datenbank im dafür vorgesehenen Verzeichnis **<Installationsverzeichnis>\ppm\server\bin\work\data_ppm\drivers** befindet.

4.3.5 JDBC-Treiber

Um auf einen Microsoft SQL-Server zuzugreifen, benötigen Sie den erforderlichen JDBC-Treiber. Sie können diesen Treiber von der Microsoft Homepage laden.

Beim Starten gibt der PPM-Server den verwendeten Interface-Typ sowie die genauen Versionen der Datenbank und des verwendeten JDBC-Treiber aus.

Detaillierte Informationen zu den erforderlichen JDBC-Treibern und unterstützten Java-Versionen erhalten Sie in den aktuellen Software AG-Systemanforderungen.

4.3.6 Unicode-Unterstützung

PPM unterstützt Multibyte-Zeichensätzen bei der Verwendung von Microsoft SQL-Server als Datenbanksystem.

Wenn Sie PPM mit Microsoft SQL-Server Unicode-Datenbanken verwenden möchten, müssen Sie folgendes beachten.

- PPM unterstützt SQL-Server-Datenbanken, für die Collation **Latin1_General_BIN** eingestellt ist. SQL-Server-Datenbanken mit anderen Collations werden nicht unterstützt.
- Vergeben Sie die Collation auf Datenbankebene.
- Wenn Sie die Einstellung der Collation nachträglich ändern, werden vorhandene Inhalte der Datenbank nicht umgesetzt. Damit die geänderten Datenbankeinstellung wirksam wird, müssen Sie das Datenbankschema des entsprechenden PPM-Mandanten durch Ausführen des Kommandozeilenprogramm **runinitdb** neu initialisieren.

Warnung

Beim Initialisieren der Datenbank werden alle importierten Daten gelöscht.

4.3.7 Migration

Wenn Sie Ihre vorhandenen SQL-Serverdatenbanken nach Microsoft SQL-Server migrieren möchten, überprüfen Sie, dass für diese Datenbanken die Collation **Latin1_General_BIN** eingestellt ist.

Warnung

Wenn Sie Datenbanken nach Microsoft SQL-Server migrieren, für die eine andere Collation als **Latin1_General_BIN** eingestellt ist, ist nicht gewährleistet, dass die migrierten Datenbanken weiterhin fehlerfrei von PPM verwendet werden können.

5 Performance Tuning

Dieses Kapitel zeigt Ihnen Performance beeinflussende Zusammenhänge zwischen der PPM-Server-Software und dem Datenbanksystem.

5.1 Gesamt-Performance

Die Laufzeiteffizienz des PPM-Serversystems insbesondere des Imports wird zu großen Teilen von dem Datenbanksystem bestimmt.

Beachten Sie folgende Performance beeinflussende Hinweise für Ihr PPM-System:

- Beobachten Sie die Laufzeitkonfiguration der Datenbankinstanz in regelmäßigen Abständen und passen Sie diese bei erkennbarem Verbesserungspotential entsprechend an.
- Halten Sie die Datenbankstatistiken auf aktuellem Stand.
- Reorganisieren Sie nach dem Verdichten oder Löschen einer großen Menge von Prozessinstanzen die Tabellen und lassen Sie alle Indizes des Datenbankschemas neu berechnen.
- Das Kommandozeilentool **runppmimport** unterstützt Sie beim Neuberechnen der Indizes, indem Sie beim Aufruf das Argument **-index new** angeben, z. B.:
- runppmimport -client <Mandantenname> -user system -password <Kennwort> -index new

Brechen Sie diesen Vorgang nach Möglichkeit nicht ab, um zeitaufwändige Konsolidierungsprozesse im Datenbankschema zu vermeiden.

5.1.1 Hardware spezifisch

Die eingesetzte Hardware beeinflusst direkt die Gesamtperformance des PPM-Serversystems.

- PPM-Server und -Datenbanksystem sollten nach Möglichkeit auf demselben Rechner installiert sein oder über ein ausreichend schnelles Netzwerk verbunden sein.
- Wählen Sie als Speicherort der Tablespace-Dateien nach Möglichkeit ein Dateisystem, das auf einem RAID 5 oder RAID 10 Array basiert.
- Eine weitere Steigerung der Performance erreichen Sie, indem Sie die Tablespace der PPM-Tablespace-Typen (siehe Kapitel Unterstützte Datenbanksysteme (Seite 11)) auf physikalisch unabhängige Dateisysteme verteilen.
- Konfigurieren Sie eine MS SQL-Server-Datenbank so, dass Transaktions-Logs und Datenbankdateien in physikalisch unabhängigen Dateisystemen gespeichert werden.

5.1.2 Konfigurationsspezifisch

- Melden Sie selbst definierte Dimensionen und Kennzahlen an einem bestimmten Prozesstyp oder Prozesstypgruppe an, damit diese Dimensionen und Kennzahlen nur für Prozessinstanzen dieses Prozesstyp oder dieser Prozesstypgruppe berechnet werden.

Warnung

Die Standardkennzahlen Prozess- und Funktionsanzahl, die Standarddimensionen Prozesstyp, Zeit und Funktion sowie alle Datenzugriffsdimensionen müssen an der Prozessbaumwurzel angemeldet sein, da sie für interne Berechnungen verwendet werden.

- Verwenden Sie wenn möglich Prozesskennzahlen und -dimensionen. Anfragen von Funktionskennzahlen und -dimensionen sind aufgrund der mehrfach größeren Datenmenge berechnungsintensiver. In der Regel werden Funktionskennzahlen nur benötigt, wenn eine Funktion in einer Prozessinstanz mehrfach vorkommt und Sie jede Funktion unabhängig voneinander auswerten möchten.

5.2 Import-Performance

Beim Einlesen der Fragmente und Berechnen der Prozessinstanzen werden große Datenmengen in die Tablespaces des Datenbanksystems geschrieben. Die Performance des Dateisystems, auf dem die Tablespace-Dateien gespeichert sind, bestimmt hauptsächlich die Performance des Datenimports.