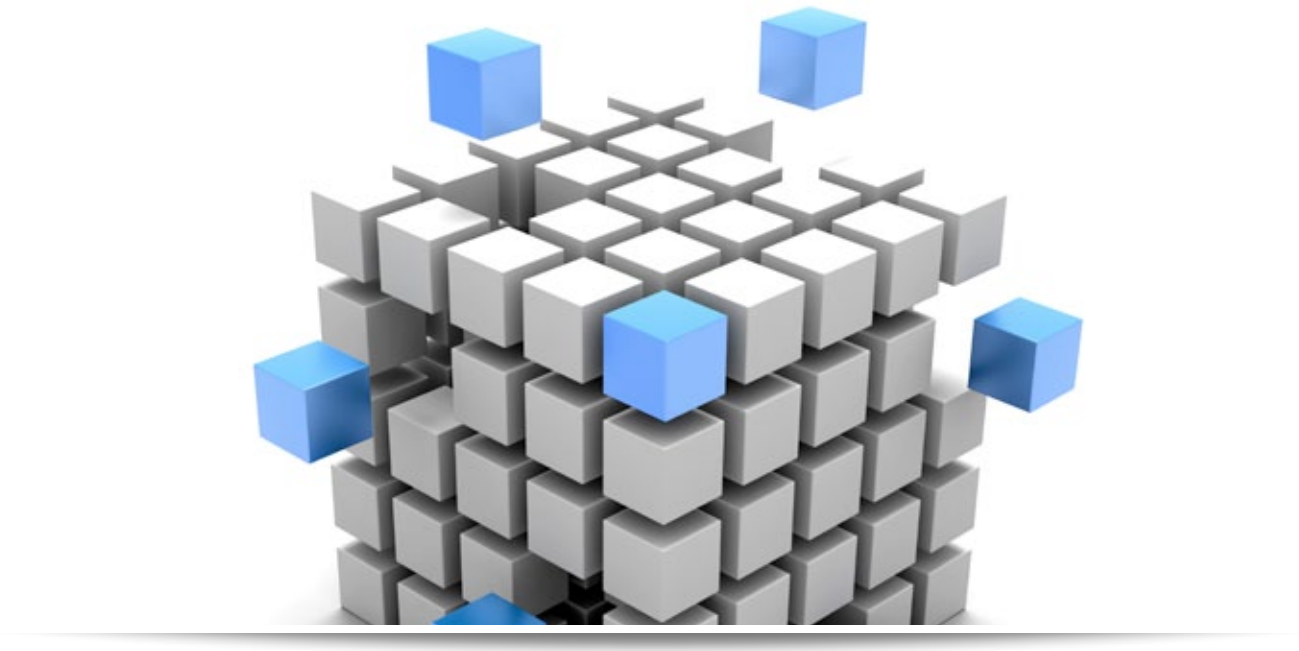

Oracle Multitenant: Einfachere Möglichkeiten für die Datenbank-Konsolidierung und mehr

Manuel Hoßfeld, ORACLE Deutschland B.V. & Co. KG

Die neue Option namens „Oracle Multitenant" basiert auf einer neuen Datenbank-Architektur, die es optional erlaubt, mehrere sogenannte „Pluggable Databases" in einem „Multitenant Container" – also einer „Container-Datenbank" – zu betreiben. Der Artikel zeigt, was das genau für Datenbank-User und DBAs bedeutet und welche Möglichkeiten sich bei der Nutzung dieser Option ergeben.

Ein Oracle-Datenbanksystem besteht bis zur Version 11 grundsätzlich aus zwei Komponenten: der Datenbank selbst in Form von Dateien und einer Instanz (oder mehrerer im Fall von RAC) mit den dazugehörigen Betriebssystem-Prozessen und Hauptspeicher. Werden mehrere Datenbank-Systeme auf einem Server betrieben, sind entsprechend viele Instanzen aktiv.

Nun ist diese klassische Architektur zwar schon lange in dem Sinne mandantenfähig (engl. „multitenant"), dass man einzelnen Anwendungen beziehungsweise Mandanten verschiedene Datenbank-Schemata zuweisen kann. Dieses Konzept stößt in der Praxis jedoch schnell an seine Grenzen, wenn man daran denkt, dass es jeden Schema-Namen pro Datenbank-Instanz nur einmal geben kann, ebenso wie sämtliche Parameter für deren Betrieb.

Im Rahmen von Oracle Multitenant wird dieses Konzept geändert: Für bessere Ressourcen-Nutzung und neue Möglichkeiten der Konsolidierung innerhalb einer Datenbank wurde nun eine neue Ebene eingeführt, die gewissermaßen zwischen Instanz und Schema angesiedelt ist: die sogenannte „Pluggable Database" – eine nach außen hin vollständige Datenbank mit einem eigenen Satz Einstellungen und Schemata, die aber nicht autark auf einem Datenbank-Server Ressourcen beansprucht, sondern sich diese mit anderen solcher „Pluggable Databases" innerhalb einer „Container-Datenbank" teilt.

Grundsätzlich ändert sich bei der Nutzung einer Pluggable Database aus Sicht eines Endanwenders nichts: Für ihn verhält sie sich genauso wie eine normale Datenbank auch. Wichtig zu wissen ist lediglich, dass der Connect immer über einen Service erfolgt, da nur so eine eindeutige Identifikation der gewünschten Pluggable Database möglich ist.

Neue Architektur und Nomenklatur

Mit der Multitenant-Architektur gibt es einige neue Begrifflichkeiten, die man im Hinterkopf behalten sollte (siehe Abbildung 1):

- *Pluggable Database (PDB)*
  Eine für Anwendungen genutzte Datenbank, die unter Verwendung der neuen Architektur betrieben wird. Sie enthält alle Anwendungsdaten samt Data Dictionary, Datenbank-Benutzern etc.
- *Container Database (CDB)*
  Der Container, in den die PDBs eingeklinkt werden. Alle PDBs innerhalb einer CDB benutzen die gleiche Datenbank-Version, die durch die CDB vorgegeben ist.

Ohne Umwege zu optimaler Performance.

areto hat die Lösung.

ORACLE Gold Partner

Strukturierte Datenmodellierung, konsequente Technologie-Nutzung und zielführende Projektstandards ziehen sich von Anfang an als roter Faden durch die Entwicklung unserer Business-Intelligence-Lösungen. Sodass am Ende selbst bei komplexen Anforderungen überraschend einfache und intelligente Lösungen stehen.

Bringen Sie auch in Ihrem Unternehmen den Knoten zum Platzen.

Rufen Sie uns an:
0221 66 95 75-75

areto consulting gmbh · Data Warehouse · Business Intelligence
Julius-Bau-Straße 2 · 51063 Köln · 0221 66 95 75-0 · www.areto-consulting.de

areto
CONSULTING. IT WORKS.

- *Non-CDB*
  Eine Oracle-Datenbank, die nach herkömmlicher Architektur unter Oracle 12*c* betrieben wird.
- *Seed-DB*
  Ein Datenbank-Template, mit dem sehr schnell eine neue PDB erzeugt werden kann. Jede CDB enthält genau eine Seed-DB mit dem Namen „PDB$SEED".

Eine PDB unterscheidet sich von einer Non-CDB dadurch, dass sie keine der folgenden eigene(n) Komponenten besitzt:

- Instanz (Memory und Hintergrundprozesse)
- Kontrolldatei
- Redo-Log-Dateien
- Undo Tablespaces

Diese Komponenten werden zentral von der CDB bereitgestellt, in der die PDB betrieben wird. Auch beim Inhalt der Datenbanken gibt es Unterschiede zum klassischen Konzept. Im Rahmen der Pluggable Databases wird das Data Dictionary in folgende Bereiche aufgeteilt:

- Einen statischen Bereich, der die für die eingesetzte Software feststehenden Informationen beinhaltet. Dieser Bereich enthält Informationen, die während der Lebenszeit der Datenbank mit der jeweils gegebenen Version nicht mehr verändert werden. Dieser wird auch als „root" bezeichnet.
- Einen anwendungsbezogenen Bereich, dessen Inhalte Veränderungen unterliegen können.

Der statische Bereich ist Bestandteil der CDB und wird über eine Verlinkung in den anwendungsbezogenen Bereich, der Bestandteil der PDB ist, einbezogen. Aus der Sicht einer PDB hat diese also ein komplettes Data Dictionary, auch wenn die nicht veränderlichen Teile aus der CDB stammen.

Vorteile der neuen Multitenant-Architektur

Die neue Architektur bietet folgende interessante Vorteile und Anwendungsmöglichkeiten:

- *Konsolidierung und Einsparung von Ressourcen*
  Dadurch dass Prozesse, RAM und große Teile des Data Dictionary nur einmal vorhanden sind, können wesentlich mehr Datenbanken auf einem System betrieben werden als mit der klassischen Architektur
- *Mobilität*
  Schnelles und einfaches „Unplug" und „Plug" einer kompletten PDB
- *Cloud-Fähigkeit*
  Schnelles Bereitstellen einer PDB durch Erzeugen aus der Seed-DB oder als Klon einer bestehenden PDB sind die ideale Basis für DBaaS (Database-as-a-Service)
- *Patchen/Migration*
  Wenn eine CDB einmal gepatcht ist, sind dies automatisch auch alle darin enthaltenen PDBs
- *Effizientere Möglichkeit für Desaster Recovery sowie Backup & Recovery*
  Sowohl Data Guard als auch Backups werden zentral auf Ebene einer CDB eingerichtet. Ein Restore eines Backups ist dennoch auch für einzelne PDBs möglich

Der Weg zur Pluggable Database

Die einfachsten Möglichkeiten stellen natürlich die Wahl der entsprechenden Installationsoption im „Oracle Universal Installer" oder die Nutzung des „Database Creation Assistant" (DBCA) dar. Diese ergeben sich allerdings nur für neue Installationen. Da die Oracle-Landschaft vieler Anwender jedoch eher nicht der sprichwörtlichen „grünen Wiese" entspricht, seien an dieser Stelle kurz zwei Möglichkeiten erwähnt, wie man bereits in einer Version vor 12*c* bestehende Datenbanken „pluggable" machen kann. Voraussetzung ist jeweils das Vorhandensein einer installierten 12*c*-Datenbank, um überhaupt eine CDB zur Verfügung zu haben. Diese muss lediglich laufen und kann gegebenenfalls auch vollkommen leer sein. Die folgenden Schritte stellen nur einen groben Leitfaden dar – die Details stehen in der Dokumentation sowie in dem am Ende des Artikels beschriebenen Oracle-Dojo Nr. 7.

Direktes Einklinken einer Non-CDB als PDB

Hierzu muss zunächst ein Upgrade der Quell-Datenbank auf 12*c* erfolgen, da nur ab 12*c* das für eine Umwandlung von Non-CDB in PDB verwendete Package „DBMS_PDB" überhaupt vorhanden ist. Nach dem Upgrade erzeugt man mithilfe dieses Packages einen Satz Metadaten im XML-Format, der Informationen zu den Einstellungen und Datendateien der Quell-Datenbank beinhaltet. Für den gängigen Fall, dass die neue 12*c*-Datenbank nicht auf dem gleichen Server läuft wie die Quell-Datenbank, kopiert man die gerade erzeugte XML-Datei sowie die Daten-Dateien auf den neuen Server.

Im nächsten Schritt wendet man das Kommando „CREATE PLUGGABLE DATABASE <Name> USING <Pfad-zur-XML-Datei>..." an. Je nachdem, wo die Dateien liegen, kommen dabei verschiedene Optionen zum Einsatz, die zum Beispiel die Dateinamen an die neuen Lokationen anpassen. Mit dem Skript „$ORACLE_HOME/rdbms/

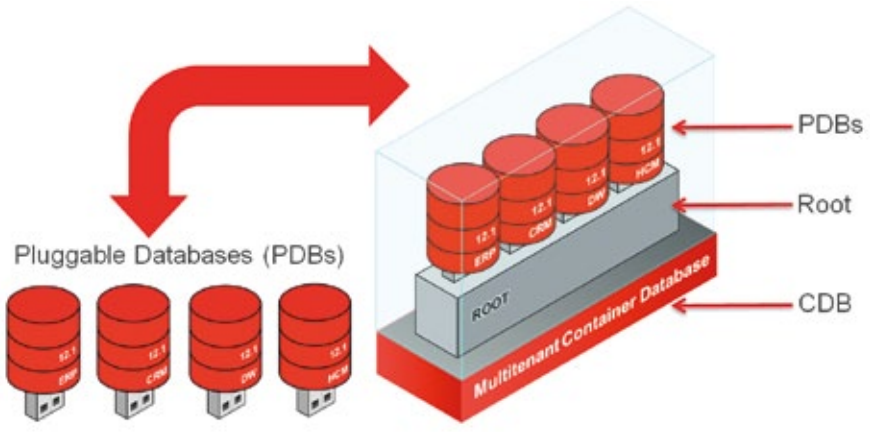

Abbildung 1: Komponenten und Begriffe von Oracle Multitenant

Pro CDB	Pro PDB
Eine Oracle-Software-Version	RMAN Point-in-time Recovery
Geplante RMAN-Backups	„Ad hoc"-RMAN-Backups
Data Guard	Möglichkeit für „Flush Shared_Pool"
Einige Parameter wie Character Set	Parameter, für die gilt „IsPDB_Modifiable = "TRUE""
Redo und Undo	

Tabelle 1

admin/noncdb_to_pdb.sql" wird nun der statische Anteil der (ehemaligen) Non-CDB endgültig entfernt, damit diese als PDB laufen kann.

Importieren eines Full Database Exports in eine leere PDB

Hierbei wird zunächst eine neue leere PDB in der 12*c*-CDB erzeugt – einschließlich des dazugehörigen Service, um diese überhaupt ansprechen zu können. Anschließend wird ein mittels „expdp … FULL=Y" erzeugter „Full Database Export" in die soeben erzeugte PDB importiert. Hierbei kann einem ein weiteres neues Feature von 12*c* behilflich sein: Data Pump unterstützt jetzt nämlich auch die Verwendung von Transportable Tablespaces, sodass das eigentliche Dump File sehr klein bleibt und schnell erzeugt ist. Hinweis: Ein Backport dieses Features ist auch in 11.2.0.3 eingeflossen, sodass es entsprechend auch für eine Migration einer 11*g*-Non-CDB in eine 12*c*-PDB verwendet werden kann.

CDB vs. PDB

Wie bereits eingangs erwähnt, verhält sich eine PDB nach außen hin grundsätzlich wie eine Non-CDB. Wegen der zugrunde liegenden Architektur mit geteilten Ressourcen aller PDBs innerhalb einer CDB existieren gleichzeitig aber auch Merkmale und Funktionen, die sich demzufolge nicht auf der Ebene einzelner PDBs abspielen. Tabelle 1 listet die wichtigsten auf.

Lizenzrechtlicher Hinweis

Wie schon aufgrund des Namens „Oracle Multitenant Option" zu erwarten ist, handelt es sich bei den in diesem Artikel vorgestellten Möglichkeiten um eine zusätzlich zur Datenbank zu lizenzierende Option. Weniger offensichtlich ist die Tatsache, dass dies für den Sonderfall von nur einer einzelnen PDB in einer CDB nicht gilt. Im Umkehrschluss bedeutet dies, dass auch ohne die entsprechende Option sämtliche 12*c*-Datenbanken mit einer einzelnen PDB betrieben werden können – auch in der Standard Edition.

Weitere Informationen

1. Umfangreiches (englischsprachiges) Whitepaper zu Oracle Multitenant: http://www.oracle.com/technetwork/database/multitenant-wp-12c-1949736.pdf
2. Aus der Reihe der deutschsprachigen „Oracle Dojos" die Ausgabe Nr. 7 zum Thema „Multitenant" – sowohl in gedruckter Form als auch online: http://www.oracle.com/webfolder/technetwork/de/community/dojo/index.html
3. Oracle Database Days: http://tinyurl.com/odd12c

Manuel Hoßfeld
manuel.hossfeld@oracle.com

Automatisierung des Information Lifecycle Managements

Ulrike Schwinn, ORACLE Deutschland B.V. & Co. KG

Was ist neu in Oracle Database 12*c* im Bereich „Information Lifecycle Management"? Was sind die sogenannten „Heat Maps"? Wie sieht die Automatisierung aus? Wie aufwändig ist das Setup? Dieser Artikel beantwortet diese und weitere Fragen und illustriert sie an einigen Beispielen.

Information Lifecycle Management (ILM) ist kein neues Schlagwort, sondern ein gängiger Begriff, um den Lebenszyklus von Daten zu beschreiben. Ziel ist es, Informationen entsprechend ihrem Wert und ihrer Nutzung optimal auf dem jeweils kostengünstigsten Speichermedium im besten Fall automatisch zu platzieren – so die Definitionen im Web.