DB Optionen nachgebaut

Marco Patzwahl
MuniQSoft GmbH
Unterhaching

Schlüsselworte
Partitioning, Row Level Security, Standby DB, DBMS_CRYPTO, Verschlüsselung, VPD

Einleitung
Viele kostenpflichtige Optionen und Funktionen der Enterprise Edition wie Partitionierung, Advanced Security, Redaction und Data Masking, Parallelisierung und Standby DB lassen sich – in gewissen Grenzen – mit entsprechendem PL/SQL Wissen auch selber nachbauen. Der Vortrag zeigt Ihnen Ansätze, wie man diese Funktionen realisieren kann. Er richtet sich jedoch nicht nur an Benutzer der Standard Edition, sondern an alle, die sich – aus Kostengründen – diese Optionen nicht gekauft haben, aber trotzdem deren Funktionen nutzen wollen.

Standard Edition vs. Enterprise Edition
Was geht nicht in der STD?

FUNKTION	Prio	Alternativen in der STD
Active Data Guard (Opt)	**3**	**Evtl. Log Miner**
Advanced Compression (Opt)	**2**	**utl_compress Packages**
Advanced Replication	**2**	**Eigene Replication mit AQ**
Application Role	**3**	**???**
Advanced Security	**2**	**Eigene Verschlüsselung**
Automatic Storage Management	**2**	**SE bei RAC 2 Node dabei**
Backup Encryption (Opt)	**3**	**Eigene Verschlüsselung nach dem RMAN Backup**
Basic Compression	**2**	
Bit-mapped Indexes	**1**	**Non Unique Index als compressed**
Block Change Tracking	**3**	**Keine**
Block Media Recovery	**2**	**Keine**
Change Data Capture	**3**	**???**
Data Mining (Opt)	**3**	**???**
Database Resource Manager	**2**	**Keine, aber DEFAULT_MAINTENANCE_PLAN in STD verfügbar**

FUNKTION	Prio	Alternativen in der STD
Deferred Segment Creation	2	Keine
Duplexed Backups	3	Zwei Backups auf verschiedene Mountpoints
Enterprise User Security	3	???
Export Transportable Tablespaces	2	Keine
Fast-Start Fault Recovery	2	???
File Mapping	3	???
Fine-grained Access Control	2	Context mit Logon Trigger
Fine-grained Auditing	3	Normales Audit für komplette Tabelle
Flashback Data Archive (Opt)	3	Trigger und Hilfstabellen (ab 12 incl.)
Flashback Database	2	Zeitbasiertes Recovery
Flashback Table	2	Flashback Query
Heat Map	3	ROWDEPENDENCIES Option der Tabelle oder ora_rowscn
In Memory Column Store	2	Spalte indizieren
Join Index	3	Keine
Managed Standby	2	Eigene Standby Lösung
Materialized View Rewrite	1	Keine
OLAP (Opt)	3	Keine
Online Index Build	1	Zweiten Index erstellen
Online Redefinition	1	Materialized View
Oracle Data Guard (Redo Apply)	2	Keine / Fremdsoftware
Oracle Data Guard (SQL Apply)	2	Keine / Fremdsoftware, Log Miner

FUNKTION	Prio	Alternativen in der STD
Oracle Database Vault (Opt)	**3**	**DDL Trigger**
Oracle Label Security (Opt)	**3**	**Ab 12c: Invisible Columns**
Parallel Backup and Recovery	**3**	**Mehrere Backups parallel starten**
Parallel Execution	**2**	**Query: Keine / DML: dbms_parallel_execute**
Partitioning (Opt)	**2**	**View-Konzept mit UNION ALL auf Tab**
Pluggable Database (ab 12c)	**2**	**Mehrere DB´s in eine konsolidiren**
Point-In-Time Tablespace Recovery	**3**	**Recovery auf zweiten Rechner**
Real Application Clusters	**Möglich**	**Zwei Node Cluster möglich**
Real Application Testing (Opt)	**3**	**???**
Result Cache	**2**	**Keine**
Rolling Upgrade	**2**	**Keine**
SecureFiles Encryption (Opt)	**2**	**dbms_crypto Package verwenden**
Server Flash Cache (Opt)	**3**	**Ab 12c: Tablespaces auf Ram Disk**
Spatial (Opt)	**3**	**Fremdanbieter**
SQL Plan Management (Opt)	**3**	**Stored Outlines**
Streams Capture	**3**	**Log Miner**
Transparent Data Encryption (Opt)	**3**	**Betriebssystem Plattenverschlüsselung**
Trial Recovery	**3**	**Keine**
Unused Block Compression	**3**	**Keine**

Hinweis: Ab Version 12.1 ist es nicht erlaubt einige Container als Standard und andere als Enterprise Edition zu benutzen.

Häufige Irrtümer, was angeblich nicht geht:
In Standard Edition sind folgende Funktionen kostenfrei nutzbar:

- Index (Standard-)Kompression (Advanced Compression ab 12c kostenpflichtig)
- Basic Tabellenkompression
- Segment und Undo Advisor
- Audit
- Query Flashback
- NOLOGGING auf Tabellen und Indizes

Partitioning
Die Standard- und Express Editions können keine partitionierten Tabellen anlegen.
Mit folgendem Trick geht es dann aber trotzdem:

```
ALTER SESSION SET EVENTS
'14524 trace name context forever, level 1';
```

Auch Row Level Security lässt sich mit einem ähnlichen Aufruf freischalten:

```
ALTER SESSION SET EVENTS
'28131 trace name context forever, level 1';
```

Partitioning auf SE:
In unserem Beispiel wird nur eine Range Partition simuliert. Hash, List, System und Subpartitionen werden nicht besprochen. Diese kommen in der Praxis jedoch auch seltener zum Einsatz.
Nachteile von Partitionierung in der Enterprise Edition:

- PL/SQL Code statt C++ im Kernel.
- Der Optimizer ist auf Partitioning optimiert.
- Eigenes Package zur Partitionsverwaltung, welches evtl. in jeder neuen Version individuelle Anpassungen benötigt.
- Es gibt keinen globalen Index.

Aufbau:
Partitionen sind wie Tabellen (Perlen), die an einer Kette zusammen aufgefädelt werden und damit als ein Objekt erscheinen. Mit einer View kann man sich dieses Verhalten nachbauen, siehe. Abb. 1.

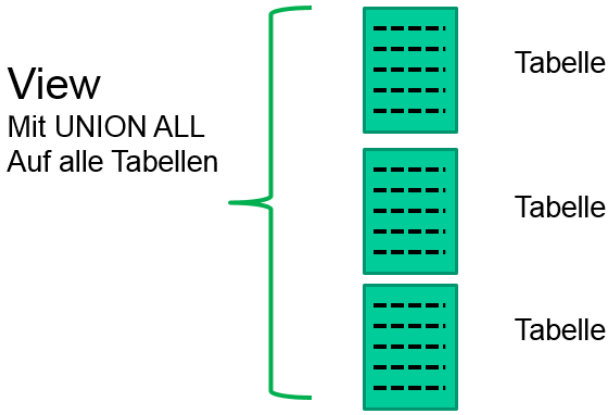

Abb. 1 Partition dargestellt als View mit UNION ALL auf Tabellen

Was man beachten sollte, ist dass der Optimizer wissen muss, welche Daten in welcher Tabelle zu finden sind. Nur dann wird er keinen FULL TABLE Scan über alle Tabellen durchführen. Ein Instead of Trigger muss die Daten, die in die View eingetragen werden auf die entsprechende Tabelle weiterleiten. Die View muss wissen, welche Tabellen (Partitionen) es insgesamt gibt.

Beispiel: Legen Sie mehrere Tabellen an, die Sie mit einer View verbinden

```
CREATE TABLE scott.emp_10
AS SELECT * FROM scott_emp WHERE deptno=10;

CREATE TABLE scott.emp_20
AS SELECT * FROM scott_emp WHERE deptno=20;

CREATE VIEW scott.empv AS
SELECT * FROM scott.emp_10
UNION ALL
SELECT * FROM scott.emp_20;
```

Alternative: Die Tabellen MÜSSEN einen Check-Constraint besitzen, damit der Optimizer unnötige Tabellen überspringt

```
ALTER TABLE scott.emp_10
ADD CONSTRAINT check_deptno_10
CHECK (deptno=10);

ALTER TABLE scott.emp_20
ADD CONSTRAINT check_deptno_20
CHECK (deptno=20);
```

Dann benötigt man noch einen Instead of Trigger, der die Daten in die richtige Tabelle verteilt.

Derzeit sind die folgenden Funktionen implementiert (Stand Juni 2015):

- CONVERT_TO_PARTITION (Tabelle in partitionierte Tabelle)
- ADD_PARTITION (Partition hinzufügen)
- DROP_PARTITION (Partition löschen)
- TRUNCATE_PARTITION (Partition truncaten)
- SPLIT_PARTITION (Partition in zwei Partitionen splitten)
- CREATE__INDEX (Index für jede Partition erstellen)
- GATHER_STATS (Statistiken für Partitionen erzeugen)

Standby DB
In der Standard Edition:
Beachten Sie die Lizensierung der beiden Rechner! Bei einer Prozessor-Lizenzierung zählen beide Rechner, wohingegen die Standby Datenbank bei einer Named User-Lizenzierung inklusive ist.

Die Parameter *log_archive_dest_1-31* lassen sich nicht auf einen Wert SERVICE, sondern nur auf LOCATION setzen. Übertragung der Archivelogs muss also manuell erfolgen, zum Beispiel:

- Windows: Robocopy, Powershell, …
- Unix: scp, sftp, rsync, …
- Bsp.: rsync –e ssh –Pazv /ora/oracle/arch/ oracle@remote:/export/home/oracle/arch

Control-Datei für STBY DB:

```
ALTER DATABASE CREATE STANDBY CONTROLFILE
AS 'c:\temp\standby.ctl';
```

Ziel-Datenbank läuft in der Mount-Phase:

```
STARTUP MOUNT
```

Skript alle 5 Minuten starten mit den Zeilen:

```
WHENEVER SQLERROR EXIT

RECOVER AUTOMATIC STANDBY DATABASE
```

Beispiel mit Robocopy (Win)

Optional kann auch ein Netzlaufwerk eingerichtet werden:

- Net use o: \\oracle_backup_server\archivelog
- Robocopy c:\oracle\archive \\oracle_backup_server\archivelog /MIR /FFT /Z /W:5

Hinweis: Die Fast Recovery Area könnte sich auch gleich komplett auf dem Remote Rechner befinden. Dazu muss jedoch unter Windows der Dienst als Administrator laufen.

Eine Alternative bietet DBVISIT an, um auch in der Standard Edition eine Standby Datenbank betreiben zu können. Die Installation und Konfiguration gestaltet sich dort recht einfach und bietet zudem eine grafische Administration an.

Security
XOR Verschlüsselung:
Eine bessere Form, im Vergleich zu einer RAW-Konvertierung, ist die XOR-Verschlüsselung. Hierbei muss auch zum Entschlüsseln der Key bekannt sein:

```
SELECT utl_raw.bit_xor(utl_raw.cast_to_raw('ABCDEF'),
utl_raw.cast_to_raw('MARCO_KEY'))
FROM dual;
```

➔ 0C0311070A194B4559

```
SELECT utl_raw.cast_to_varchar2(
utl_raw.bit_xor('0C0311070A194B4559', utl_raw.cast_to_raw('MARCO_KEY')))
FROM dual;
```

DBMS_CRYPTO Package:
Dieses Package ersetzt bzw. ergänzt das bisherige Package dbms_obfuscation. Dabei müssen Strings nicht mehr ein Vielfaches von 8 Byte sein, jedoch müssen sie zuerst in RAW konvertiert werden. Das neue Package verfügt über eine größere Anzahl an Verschlüsselungsroutinen.

- dbms_crypto.des_cbc_pkcs5 (Schlüssellänge 8 Byte)
- dbms_crypto.des3_cbc_pkcs (24 Byte)
- dbms_crypto.encrypt_rc4
- dbms_crypto.encrypt_aes128

Beispiel: Verschlüsselung

```
p_encrypted_raw := dbms_crypto.encrypt(
src => p_text_raw,
typ => p_crypto_typ,
key => p_key);
```

Beispiel: Entschlüsselung

```
p_decrypted_raw := dbms_crypto.decrypt(
src => p_encrypted_raw,
typ => p_crypto_typ,
key => p_key);
```

Verbesserte Hashfunktion:
Selbst die Enterprise Edition hat derzeit (noch) nicht die pbkdf2-Hashfunktion implementiert. Hier wird ein Hashwert 1000 Fach über eine Hashfunktion laufen gelassen, um eine Brute Force Attacke zu erschweren. Diese Funktion ist zudem leicht implementierbar.

VPD nachgebaut:
VPD steht für Virtual Private Database und ermöglicht die Mandantenfähigkeit einzelner Tabellen, damit verschiedene Benutzer nur bestimmte Daten einer Tabelle sehen. Die Idee:
- Kontextparameter, die über sys_context wieder ausgelesen werden können, werden gesetzt
- View auf die Tabelle setzen, die einen Zusatz-Filter verwendet; in der Form: spalte=sys_context(…)
- Zusätzlich können einzelne Spalten verschlüsselt werden und müssen durch einen Session Key wieder freigeschalten werden.

Kontaktadresse:

Marco Patzwahl

MuniQSoft GmbH

Grünwalder Weg, 13a

D-82008 Unterhaching

Telefon: +49 (0) 89-679090-40

Fax: +49 (0) 89-679090-50

E-Mail info@muniqsoft.de

Internet: http://www.muniqsoft.de