consinto

CONSULTiNG SYSTEMiNTEGRATiON OPERATiON





Hochverfügbarkeit

Rolling Upgrade einer 11g Datenbank

Rainer Klomps, Senior Consultant, DBA

Rolling Upgrade einer Oracle 11g Datenbank

DOAG Regionaltreffen NRW
am 23.02.2011 in Siegburg

Agenda

- Historie
- Benötigte Ressourcen / Infrastruktur
- Voraussetzungen
- Konventionen / Benennungen
- Bevor wir beginnen …
- Vorgehensweise
- Vorgehensweise / Details
- Q & A

Historie

- Raumangebot an die DOAG
- Themensuche
- Projekt zur Neukonfiguration einer 10g DataGuard Umgebung
- Vorschlag, über dieses Projekt zu berichten
- Diskussion mit Kollegen
  - ➔ DataGuard ist „alt“ und „langweilig“ ☺
  - ➔ Hochverfügbarkeit nimmt eine immer größere Bedeutung ein
  - ➔ Streams ist schwierig zu konfigurieren und vielen unbekannt
    - **➔ Online Upgrade einer 11g Datenbank**

Benötigte Ressourcen / Infrastruktur

- Freier Plattenplatz entsprechend der Speicherbelegung der Produktionsumgebung
- Zweiter Rechner, falls verfügbar
- Ggf. Netzwerkverbindung
- (Arbeits-)Zeit für die Konfiguration

Konventionen / Benennungen

- Die Datenbank die aktualisiert werden soll wird mit **A** bezeichnet, die zum Upgrade eingesetzte Standby Datenbank mit **B**. Die zugehörigen Server werden analog mit **A** bzw. **B** bezeichnet.
- **Tool A>** Steht für ein Statement, das auf Datenbank A ausgeführt werden muss.
- **Tool B>** Analog für Datenbank B.
- **Tool A/B>** Analog für Datenbanken A **UND** B.
- **Tool** kann hierbei z.B. SQL (=SQL*PLUS), RMAN (=Recovery Manager) oder SH (Unix-Shell) sein.

Bevor wir beginnen …

- Ist ein aktuelles Backup vorhanden?
- Ist das Backup lesbar?
- … und zwischendurch nach größeren Änderungen
- **SQL A/B>** create restore point <name> guarantee flashback database;

Voraussetzungen

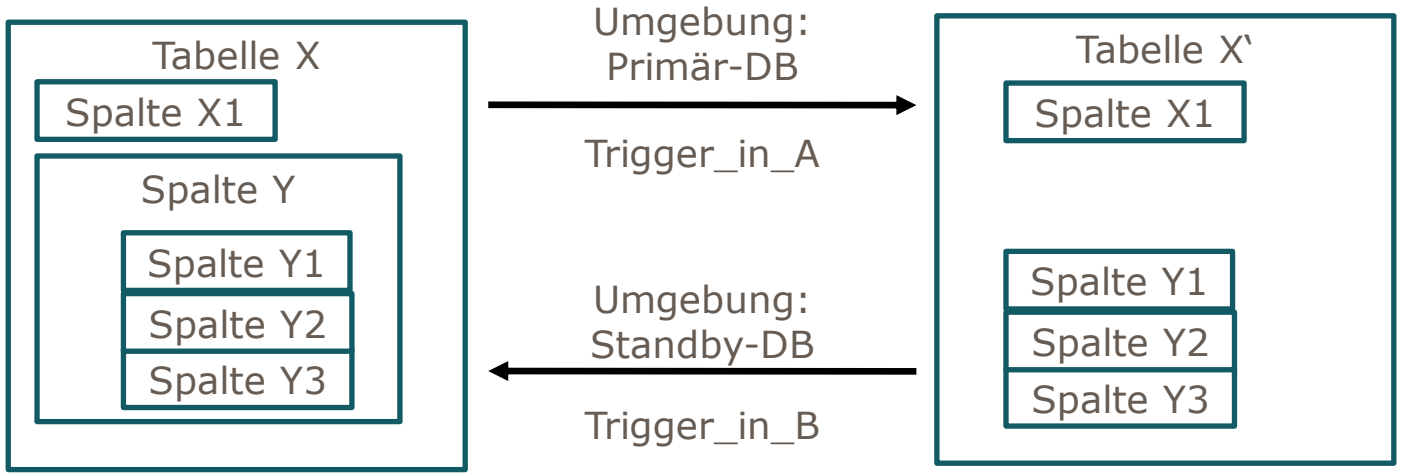
- Die Ausgangs-DB darf nicht Teil einer DataGuard **Broker** Konfiguration sein, die Nutzung von DataGuard Konzepten wird hier jedoch sehr wohl eingesetzt.
- Einrichtung einer logical Standby DB mit Protection Mode in {„maximum availability"; „maximum performance"}, also **NICHT** „maximum protection".
- LOG_ARCHIVE_DEST_<N> **für die Standby DB** darf nicht auf mandatory eingestellt sein.
- COMPATIBLE Parameter = Version der Ausgangs-DB
- Archivelog-Modus muss eingeschaltet sein.
- SUPPLEMENTAL LOGGING muss eingeschaltet werden, falls DBA_LOGSTDBY_NOT_UNIQUE Objekte enthält.
- Tabellen sollten einen Primärschlüssel oder zumindest einen Unique Key Constraint haben, evtl. mit der Option ‚RELY DISABLE' anlegen.

Einschränkungen

- Einige Datentypen werden von dem hier vorgestellten Verfahren nicht unterstützt, so dass entsprechende Tabellen nicht berücksichtigt werden.
  - BFILE
  - Collections (inkl VARRAYS und Nested Tables)
  - Multimedia Datentypen(inkl. Spatial, Image, Oracle Text)
  - ROWID, UROWID
  - Benutzerdefinierte Typen
  - XMLType stored as Object Relational
  - Binary XML
- Die durch folgende Statements gelieferten Objekte bzw. Objekte der betreffenden Benutzer werden nicht unterstützt:
- **SQL A>** select distinct owner, table_name from dba_logstdby_unsupported;
- **SQL A>** select owner from dba_logstdby_skip where statement_opt = ‚INTERNAL SCHEMA';

Vorgehensweise / 1

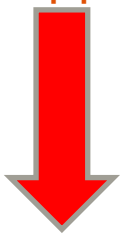
- Vorgehensweise bei nicht unterstützten Datentypen
  - Einrichtung von Datenbanktriggern die die betroffenen Objekte ‚normalisieren' und nur in der Primärdatenbank feuern.
  - Einrichtung von Datenbanktriggern die die normalisierten Objekte in das ursprüngliche Format zurückverwandeln und nur in der Standby-DB feuern.



Vorgehensweise / 2

- **SQL A>** create or replace trigger Trigger_in_A …
- **SQL B>** create or replace trigger Trigger_in_B …

- **SQL B>** execute dbms_ddl.set_trigger_firing_property(-
  trig_owner => '<SchemaName>', -
  trig_name => '<Trigger_in_B>', -
  property => dbms_ddl.apply_server_only, -
  setting => TRUE);



Trigger_in_B darf nur auf dem SQL apply Server, also in der Standby DB feuern.

Vorgehensweise / 3

- Einschalten der Protokollierung aller Transaktionen, die von logischen Standby Datenbanken nicht unterstützt werden
  - **SQL>** EXECUTE DBMS_LOGSTDBY.APPLY_SET('LOG_AUTO_DELETE', 'FALSE');
  - **SQL A>** EXECUTE DBMS_LOGSTDBY.APPLY_SET('MAX_EVENTS_RECORDED', DBMS_LOGSTDBY.MAX_EVENTS);
  - **SQL A>** EXECUTE DBMS_LOGSTDBY.APPLY_SET('RECORD_UNSUPPORTED_OPERATIONS‘, 'TRUE');
  - Sperren der betroffenen Objekte

  oder
  - Übertragen der betroffenen Objekte mittels Data Pump

Vorgehensweise / 4

- **Ggf.** Hinzufügen einer Connect-Failover und Transparent Application Failover Spezifikation für eine zweite (Standby-)DB zu derjenigen für die Quelldatenbank (TNSNAMES-Konfiguration). Dazu dynamischen Service „SWITCH" in der DB einrichten.

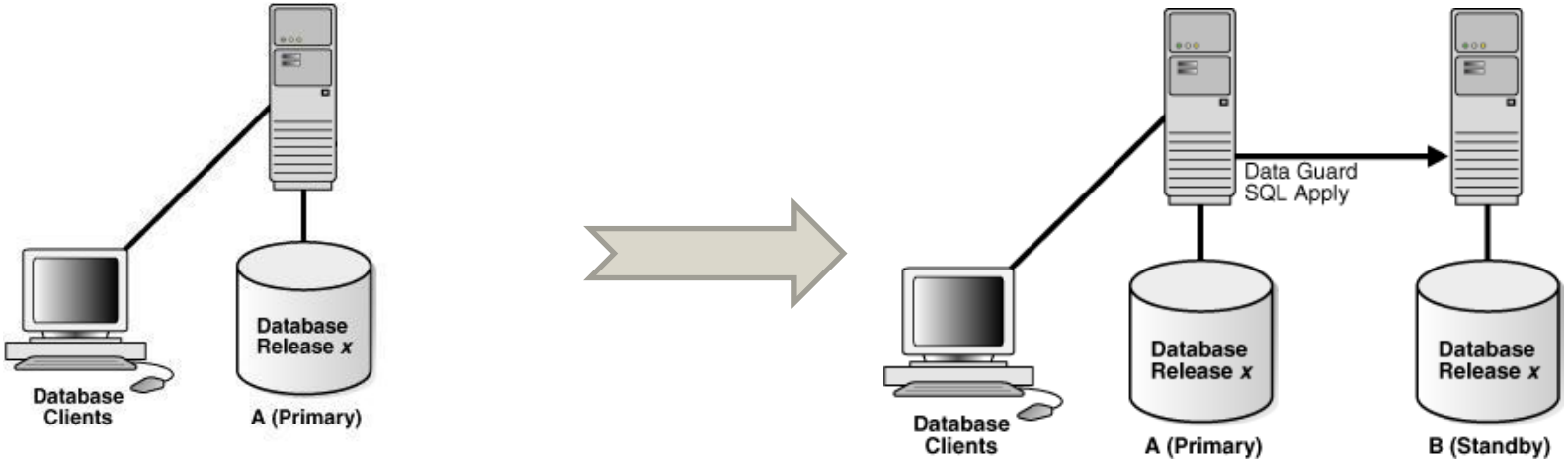
- **SQL A>** alter system set service_names=,<Originalwert>, **SWITCH**';
- **SQL B>** alter system set service_names=,<Originalwert>, **SWITCH**';

- Einrichten eines TNSNAMES Eintrags

```
SWITCH=
     (DESCRIPTION=
       (FAILOVER=ON)
       (ADDRESS_LIST=
         (ADDRESS=(PROTOCOL=tcp)(HOST=server_A)(PORT=1521))
         (ADDRESS=(PROTOCOL=tcp)(HOST=server_B)(PORT=1521)))
      (CONNECT_DATA=
       (SERVICE_NAME=SWITCH.consinto)
       (FAILOVER_MODE=
        (TYPE=SELECT)
        (METHOD=PRECONNECT))))
```

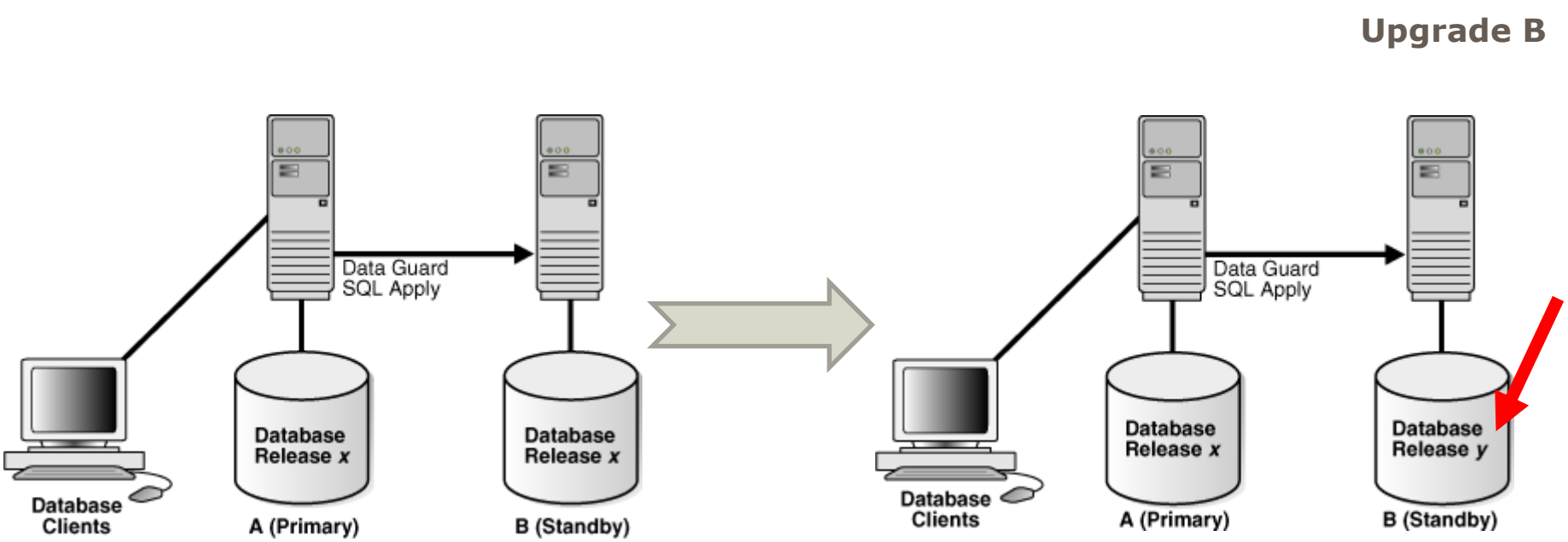
Vorgehensweise / 5

- Einrichtung einer physischen Standby Datenbank (B im mount-Status)
  SQL> startup nomount;
- **RMAN B>** connect target sys/<Password>@A auxiliary sys/<Password>@B
  RMAN B> duplicate target database for standby from active database dorecover nofilenamecheck;



- Umwandlung in eine logische Standby Datenbank
  - **SQL B>** RECOVER MANAGED STANDBY DATABASE CANCEL;
  - **SQL A>** EXEC DBMS_LOGSTDBY.BUILD;
  - **SQL B>** ALTER DATABASE RECOVER TO LOGICAL STANDBY KEEP IDENTITY;
  - **SQL A>** EXEC DBMS_LOGSTDBY.BUILD;
  - **SQL B>** ALTER DATABASE OPEN;
  - **SQL B>** ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE;

- Upgrade der logischen Standby Datenbank



Vorgehensweise Details / 1

- Es werden Standby Redolog Dateien benötigt
- Ausschalten der Anwendung von SQL auf der logischen Standby DB
  SQL B> alter database stop logical standby apply;
- Upgrade der logischen Standby DB
- Einschalten der Anwendung von SQL auf der logischen Standby DB
  SQL B> alter database start logical standby apply immediate;
- Überprüfen der Anwendung von SQL auf der logischen Standby DB (Synchronisationsprozess)
  SQL B> alter session set NLS_DATE_FORMAT = ‚DD.MM.YYYY HH24:Mi:ss‘;
  SQL B> select sysdate, applied_time from v$logstdby_progress;

Vorgehensweise Details / 2

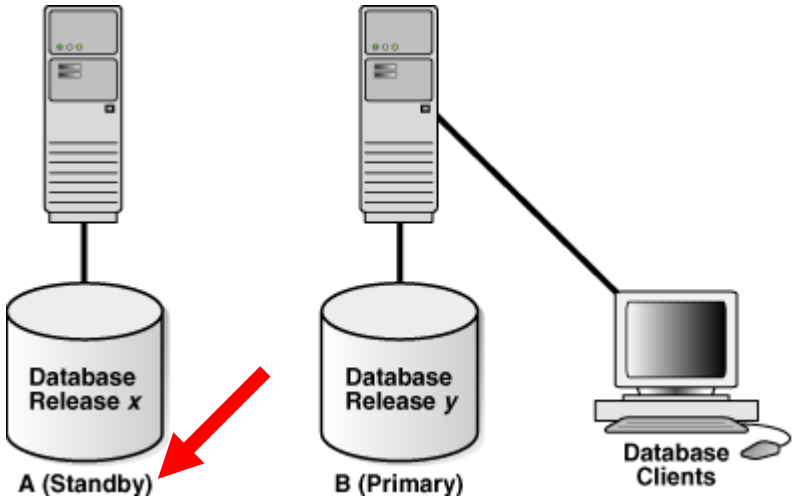
- **SQL B>** SELECT EVENT_TIMESTAMP, EVENT, STATUS FROM DBA_LOGSTDBY_EVENTS ORDER BY EVENT_TIMESTAMP;
- Import von Tabellen mit Datentypen die von logischen Standby-Datenbanken nicht unterstützt werden in die Standby-Datenbank (sofern die Trigger-Lösung nicht verwendet wurde).

- Beispiel:
  SH B> impdp <DB-User>
  network_link=<DB_LINK_ZU_A>
  tables=<Liste von Tabellen>
  table_exist_action = TRUNCATE

Vorgehensweise Details / 3

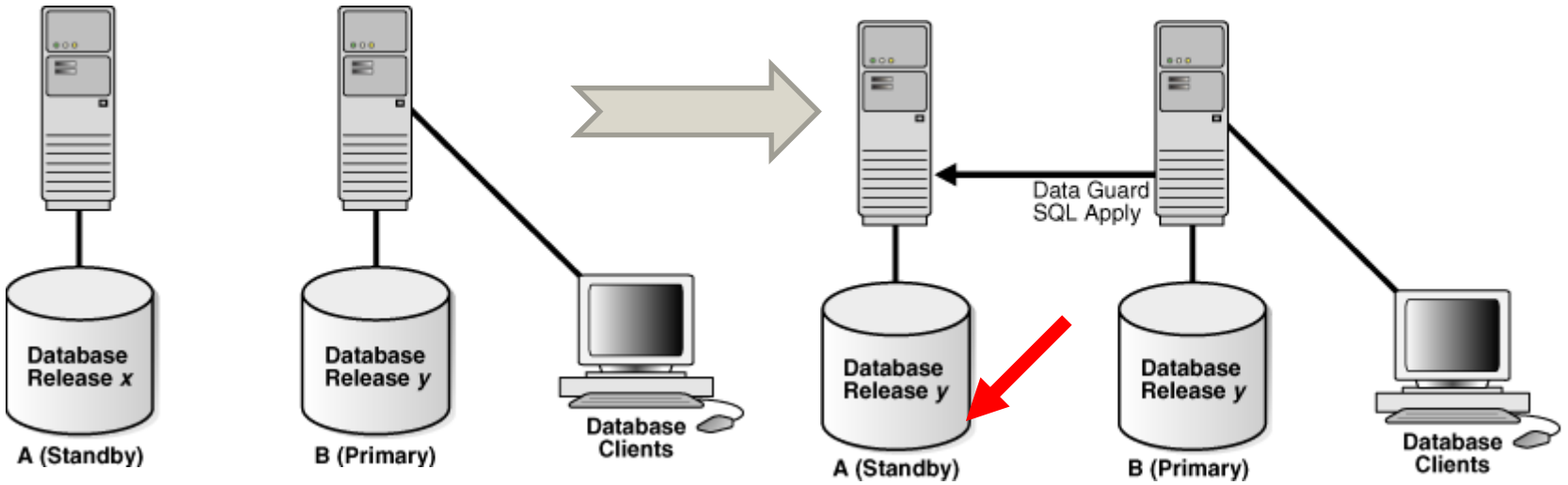
- Durchführen eines “unprepared switchover”
  Das übliche Zwei-Phasen Switchover Verfahren, in dem beide Datenbanken über ein Prepare-Statement auf den Wechsel vorbereitet werden, kann hier nicht angewandt werden, weil die DB Versionen zu diesem Zeitpunkt verschieden sind.

- **SQL A>** alter database commit to switchover to logical standby;



Vorgehensweise Details / 4

- Upgrade der Ausgangsdatenbank (Datenbank A)
- Einrichten eines DB-Link von A zu B „**von_A_zu_B**“
- Einschalten der Anwendung von SQL auf Datenbank A
  SQL A> ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE NEW PRIMARY **von_A_zu_B**;



Vorgehensweise Details / 5

- Ggf. Ändern des COMPATIBILITY Parameters
  SQL A/B> alter system set compatible=,<Versionsnr.>‘;
- Überprüfen der Datenübertragung mittels der View DBA_LOGSTDBY_EVENTS (wie bereits oben beim Umschalten A ➔ B gezeigt).
- Zurückschalten der Standby-DB A in den Primärmodus. Dies ist jetzt mit dem üblichen prepare möglich, da beide DBs den gleichen Versionsstand haben.
  SQL B> alter database prepare to switchover to logical standby;
  SQL A> alter database prepare to switchover to primary;
  SQL B> alter database commit to switchover to logical standby;
  SQL A> alter database commit to switchover to primary;
- Ggf. Löschen der Datenbank und Softwareinstallation auf Server B

Fragen?

Vielen Dank
für Ihre
Aufmerksamkeit