Viele Upgrades, wenig Downtime

Versionswechsel in einer MAA-Umgebung – Bericht aus einem laufenden Projekt

Markus Flechtner

Trivadis – Our key figures.

- Founded in 1994.
- 16 Trivadis locations with more than 650 employees.
- Sales of CHF 115 million (EUR 106 million).
- Over 250 Service Level Agreements.
- More than 4000 training participants.
- Research and development budget: CHF 5.0 million.
- More than 1900 projects each year with over 800 customers.
- Financially independent and sustainably profitable.

Markus Flechtner

- Principal Consultant, Trivadis, Düsseldorf
- Oracle seit 1990: SW-Entwicklung, Support, DBA
- Schwerpunkte: RAC, HA, Upgrade & Migration
- Kursreferent: RAC, New Features, Multitenant
- Co-Autor des Buches "Der Oracle DBA" (Hanser, 2016)

@markusdba

www.markusdba.de

RACSIG Real Application Clusters Special Interest Group

DOAG

BASEL | BERN | BRUGG | BUKAREST | DÜSSELDORF | FRANKFURT A.M. | FREIBURG I.BR. | GENF
HAMBURG | KOPENHAGEN | LAUSANNE | MANNHEIM | MÜNCHEN | STUTTGART | WIEN | ZÜRICH

trivadis

Agenda

trivadis

- Die Aufgabenstellung
- Upgrade von OS und Grid Infrastructure
- Datenbank-Upgrade
- Zwischenstand

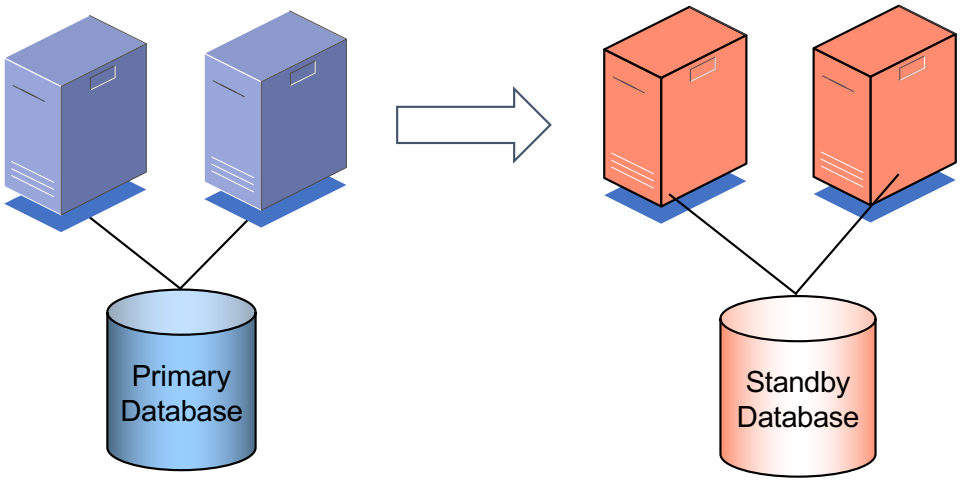
trivadis

Die Aufgabenstellung

Ausgangskonfiguration

trivadis

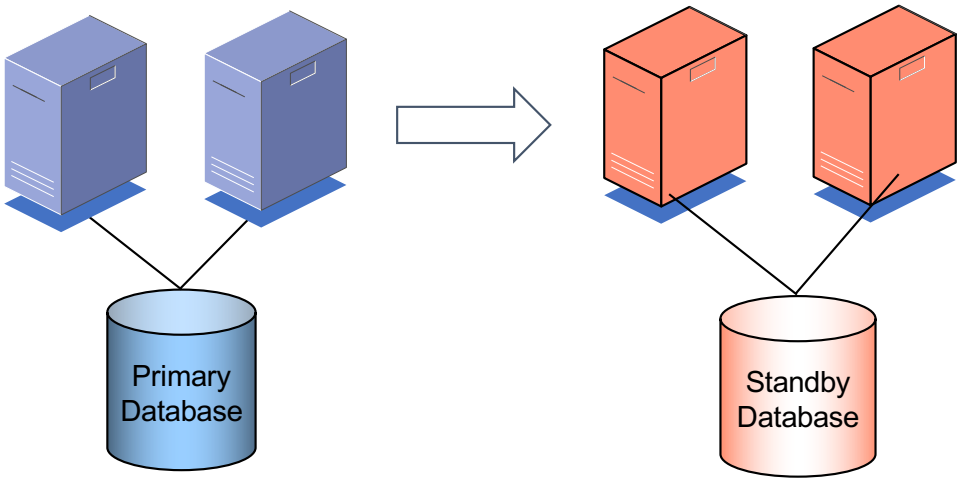
- 2 2-Knoten-Cluster
  - 1 x Primary (Server P1 + P2)
  - 1 x Standby (Server S1 + S2)
- SLES 11 SP 3
- Oracle Grid Infrastructure 12.1.0.2
- Oracle RDBMS 11.2.0.4 und 12.1.0.2



Geplante Zielkonfiguration

trivadis

- 2 2-Knoten-Cluster
  - 1 x Primary (Server P1 + P2)
  - 1 x Standby (Server S1 + S2)
- SLES 12 SP4
- Oracle Grid Infrastructure 19c
- Oracle RDBMS 18c oder 19c



Die Rahmenbedingungen

trivadis

Ausgangssituation

- 2 2-Knoten-Cluster
  - 1 x Primary (Server P1 + P2)
  - 1 x Standby (Server S1 + S2)
- SLES 11 SP 3
- Oracle Grid Infrastructure 12.1.0.2
- Oracle RDBMS 11.2.0.4 und 12.1.0.2

Geplante Zielkonfiguration

- 2 2-Knoten-Cluster
  - 1 x Primary (Server P1 + P2)
  - 1 x Standby (Server S1 + S2)
- SLES 12 SP4
- Oracle Grid Infrastructure 19c
- Oracle RDBMS 18c oder 19c

- **Inplace-Upgrade von SLES 11 SP3 nach SLES 12 SP4 ist nicht möglich**
- **Keine neue Server-Hardware**
- **Failoverfähigkeit muss gewährleistet bleiben**
- **Vorgabe lt. SLA: maximale Datenbank-Downtime 30 Minuten**

trivadis

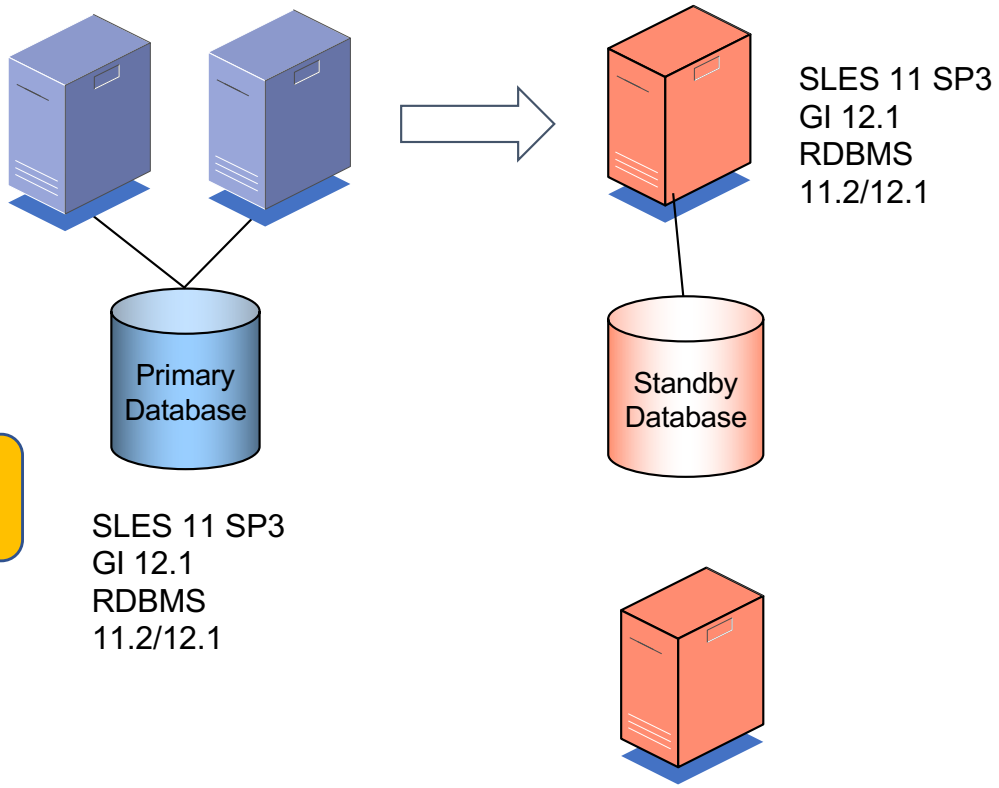
Upgrade von OS und Grid Infrastructure

Schritt 1: Standby-Cluster aufbrechen

trivadis

- Ein Knoten (S2) wird aus dem Standby-Cluster herausgenommen.
- Log-Apply läuft über den verbleibenden Knoten (S1).
- (Eingeschränkte Failover-Fähigkeit für den Disaster-Fall).

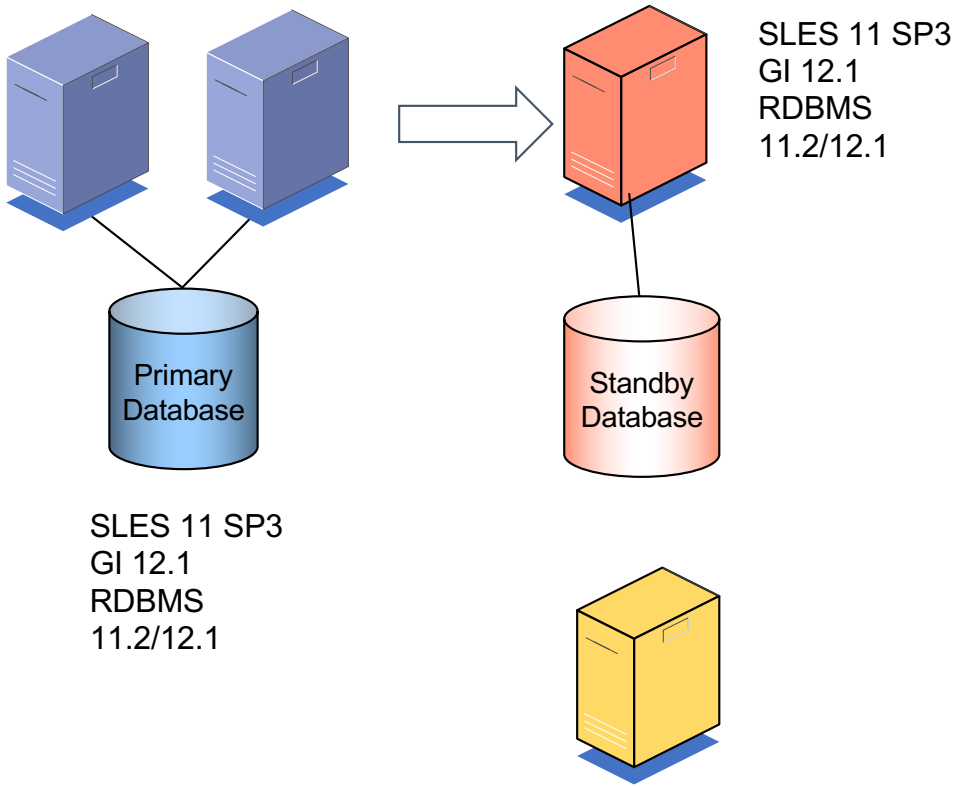
Ist im Vorfeld abzustimmen & anzukündigen!



Schritt 2: Zweiten Standby-Cluster aufbauen (1)

trivadis

- Der Server S2 wird mit SLES 12 SP 4 installiert.
  - **Hostname + IP-Adresse bleiben unverändert**
- Mit diesem Knoten wird ein neuer Standby-Cluster aufgebaut.
  - Neuer SCAN-Name + SCAN-IPs erforderlich (temporär).
  - **(Neuer) Cluster-Name kann später nicht geändert werden**
- Alternativ hätte man S2 wieder in den vorhandenen Cluster aufnehmen können und anschl. den GI-Upgrade auf 19c machen können. Im Fehlerfall hätte man dann aber keine Failover-Möglichkeit mehr.



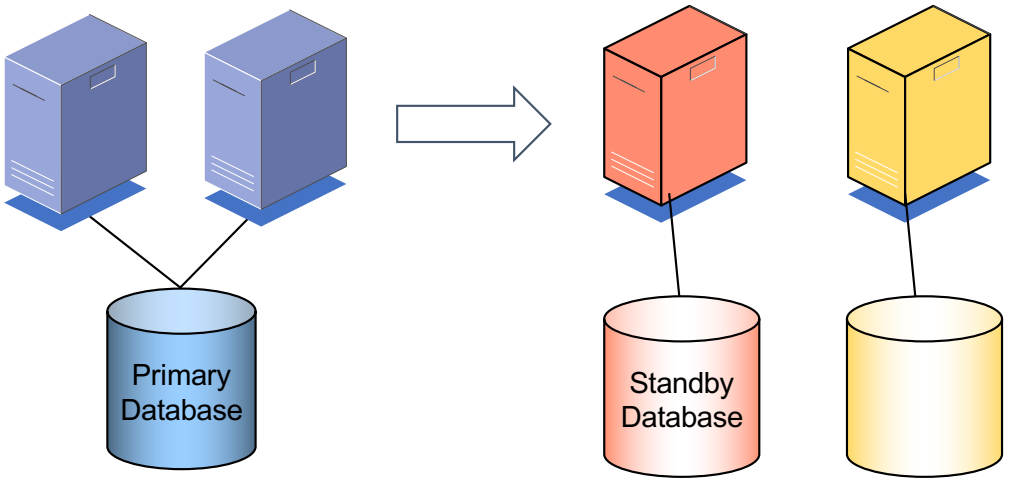
Schritt 2: Zweiten Standby-Cluster aufbauen (2)

trivadis

- Oracle-Software
  - Grid Infrastructure 19c
  - Oracle RDBMS 11.2.0.4 + 12.1.0.2

- Diskgruppen-Konfiguration kann geändert werden
➔ In diesem Fall muss mit STANDBY_FILE_NAME_CONVERT gearbeitet werden

- Zusätzliches Storage ist erforderlich



SLES 11 SP3
GI 12.1
RDBMS
11.2/12.1

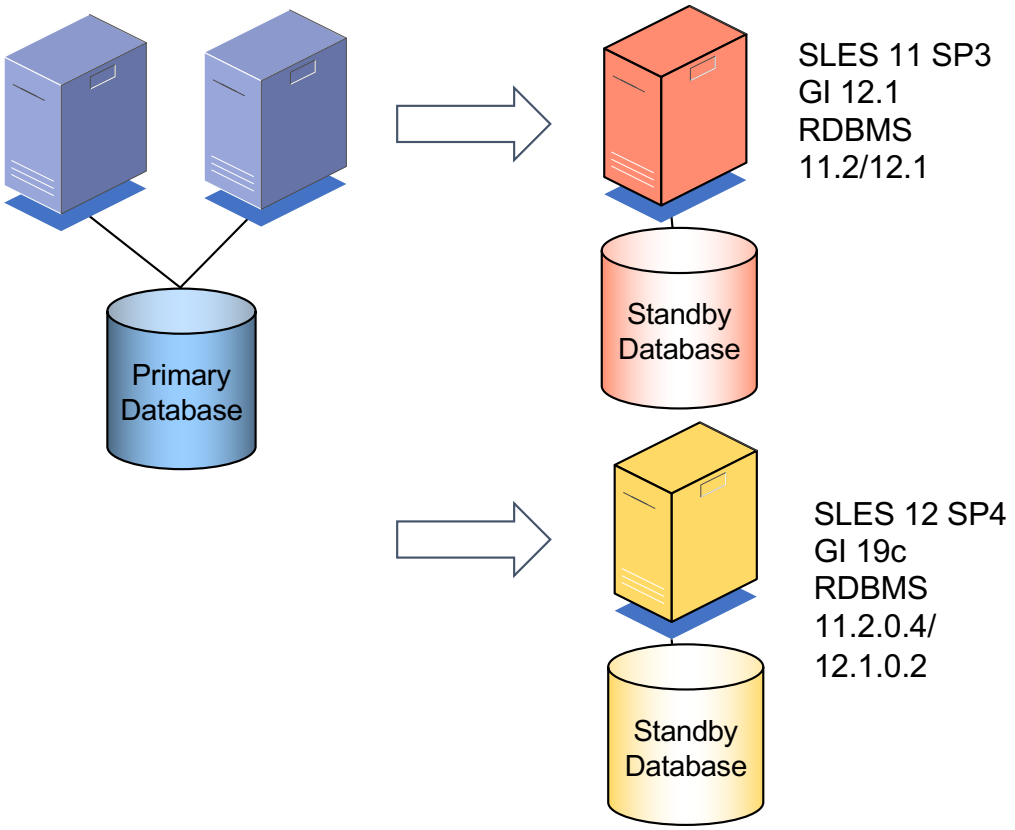
SLES 11 SP3
GI 12.1
RDBMS
11.2/12.1

SLES 12 SP4
GI 19c
RDBMS
11.2/12.1

Schritt 3: Zweite Standby-Datenbank aufbauen

trivadis

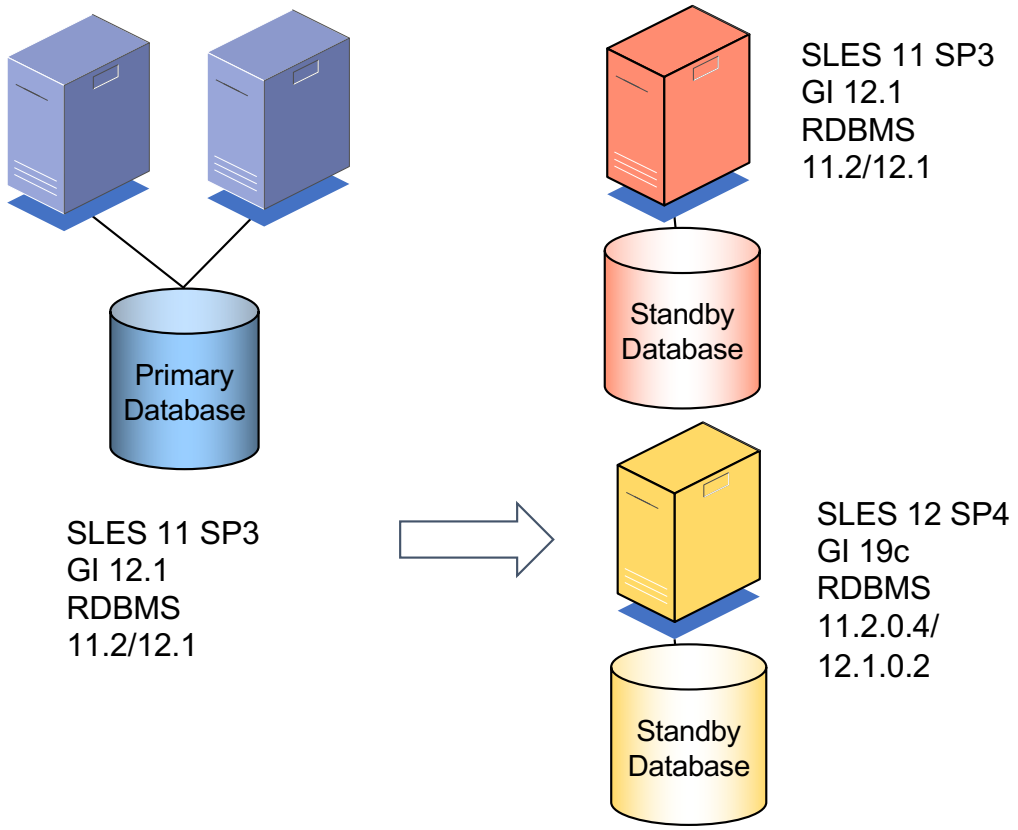
- Auf dem zweiten Standby-Cluster wird eine zusätzliche Standby-Datenbank aufgebaut
- Betrieb in der gleichen Version wie die Primär-Datenbank (11.2.0.4 oder 12.1.0.2)



Schritt 4: Standby-Cluster 1 runterfahren

trivadis

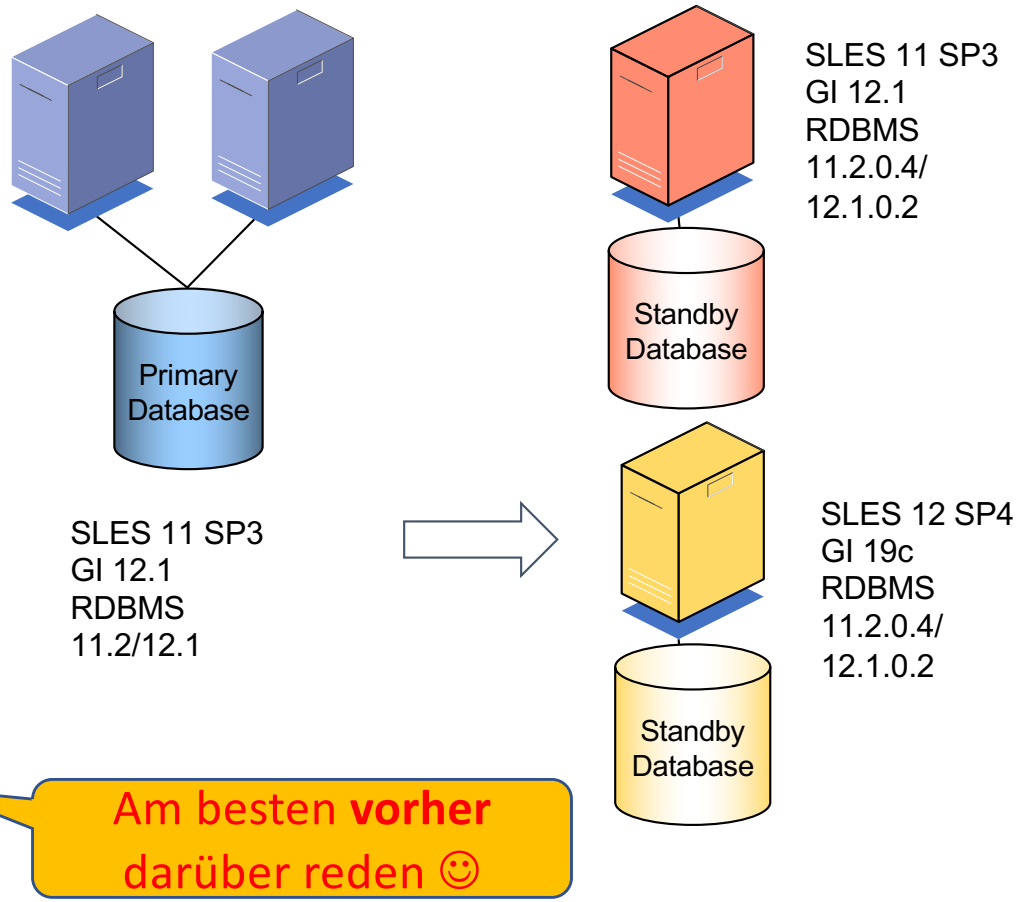
- Der erste Standby-Cluster wird heruntergefahren.
- Damit verbleibt die neue Standby-Datenbank als einzige Standby-Datenbank im Failover-Fall



Schritt 5: Standby-Cluster 2: SCAN ändern

trivadis

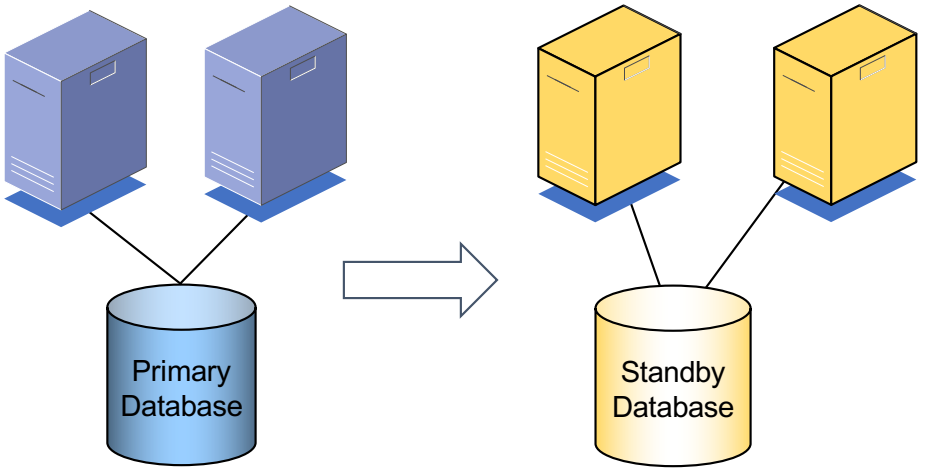
- Der SCAN-Name (und die SCAN-IPs) des 2.Standby-Clusters werden auf den SCAN-Namen bzw. die SCAN-IPs des ersten Clusters geändert
➔ dadurch muss die Client-Konfiguration nicht geändert werden
➔ aber (dauerhaft) Unterschied SCAN-Name <> Cluster-Name
- Nach der SCAN-Änderung muss die tnsnames.ora auf den Clustern angepasst werden, damit der Log-Transfer weiterhin funktioniert.
- **!! Während der SCAN-Änderung ist das System nicht failoverfähig!**



Schritt 6: Knoten S1 aktualisieren

trivadis

- Auf dem Server S1 wird SLES 12 SP 4 installiert
- Anschließend wird S1 wieder in den neuen Standby-Cluster aufgenommen
- Installation GI 19c ("addnode.sh")
- Installation RDBMS 11.2.0.4+12.1.0.2 ("addnode.sh")
- Anlegen der zusätzlichen DB-Instanz(en) für die Standby-Datenbank(en)

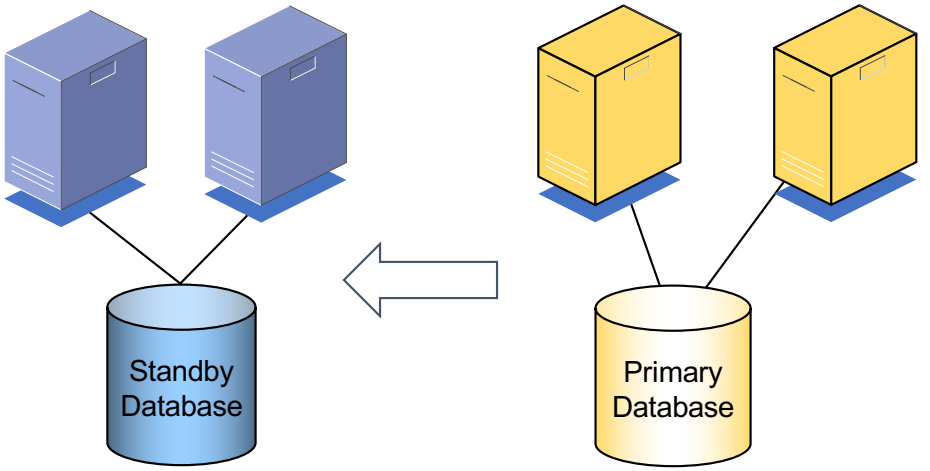


Schritt 7: Switchover

trivadis

- Es erfolgt ein Switchover auf den neuen Standby-Cluster
- **(kurze) Downtime der Applikation**

- Anschließend werden die Schritte 1 – 6 auf dem (ehemaligen) Primär-Cluster ausgeführt



SLES 11 SP3
GI 12.1
RDBMS 11.2/12.1

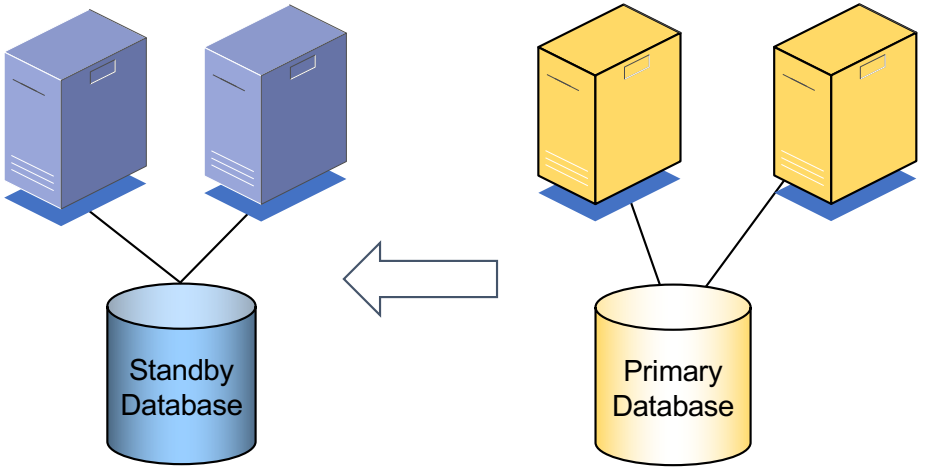
SLES 12 SP4
GI 19c
RDBMS 11.2.0.4/12.1.0.2

Zwischenergebnis

trivadis

- Die Infrastruktur auf beiden Clustern hat den Stand
  - **SLES 12 SP4**
  - **Grid Infrastructure 19c**
- Die Datenbank-Versionen sind noch unverändert.
- Die Cluster sind unter ihrem "alten" SCAN-Namen erreichbar.
- **Aber:** SCAN-Name <> Cluster-Name

- Der Upgrade der Datenbanken erfolgt zu einem späteren Zeitpunkt.



SLES 12 SP4
GI 19c
RDBMS 11.2.0.4/12.1.0.2

SLES 12 SP4
GI 19c
RDBMS 11.2.0.4/12.1.0.2

trivadis

Datenbank-Upgrade

Upgrade mit minimaler App-/DB-Downtime (1)

trivadis

- Grundsätzlich gilt

Je kürzer die Downtime beim Upgrade sein soll,
desto aufwändiger sind die Vorbereitungen!

- Es ist also abzuwägen, in welcher Phase man den "Preis" zahlen möchte
  - Vorbereitungsaufwand
  - ("längere") angekündigte Downtime

Upgrade mit minimaler App-/DB-Downtime (2)

trivadis

- Datenbank-Upgrade (oder Server-Wechsel) **mit bidirektionaler Replikation** zwischen zwei Datenbanken (alt/neu) inkl. Konfliktauflösung
  - ➔ theoretisch keine Downtime der Applikation erforderlich
  - Verbindungsdaten ändern, DISCONNECT_SESSION (Transactional)

- Voraussetzung: Applikation ist "HA-fähig" (=> automatischer Reconnect)

- **Möglichkeiten/Tools**
  - Oracle Streams (früher, desupported mit Oracle 19c)
  - Golden Gate (kostet extra)
  - Shareplex (kostet extra)

Steht uns nicht zur Verfügung

Upgrade mit minimaler App-/DB-Downtime (3)

trivadis

- Datenbank-Upgrade **ohne bidirektionale Replikation** erfordert zumindest

    1. Beenden der Applikation
    2. Anpassen der Verbindungsdaten (LDAP, tnsnames, JDBC-URLs …)
    3. Starten der Applikation

- .. kann dauern
- .. manchmal länger als die maximal erlaubte Downtime lt. SLA
- .. und mit der Datenbank ist noch nichts passiert

Manche SLA-Vorgaben sind graue Theorie und in der Praxis nicht (immer) umsetzbar!

Möglichkeiten

trivadis

- Transient Logical Standby Database

- Transportable Tablespaces

Transient Logical Standby Database (1)

trivadis

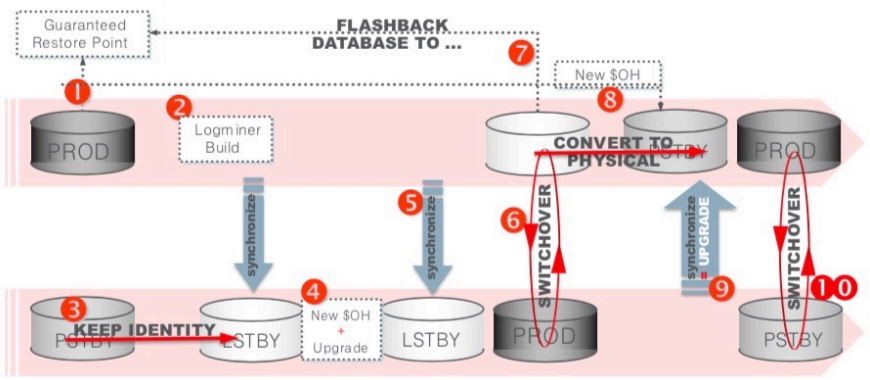
- **Überblick**
  - In einer Standby-Umgebung erfolgt der Upgrade in einer (temporären) logischen Standby-Datenbank
  - Änderungen, die während des Upgrades anfallen, werden anschließend via "SQL Apply" nachgefahren

- Tool-Unterstützung (MOS-Note 949322.1)
- Nach dem Upgrade ist der Parameter COMPATIBLE noch auf dem Ausgangswert
  ➔ für Wechsel erneute Downtime erforderlich
- Seit Oracle 12c Release 1 unterstützt das Package DBMS_ROLLING den Upgrade mittels transienter logischer Standby-Datenbank (Active DataGuard erforderlich)

Transient Logical Standby Database (2)

trivadis

- **Ablauf**
  1. Physische Standby-Datenbank anlegen
  2. Guaranteed Restore-Point anlegen
  3. Physische Standby-Datenbank in logische Standby-DB konvertieren (Single-Instance-Modus)
  4. Upgrade der logischen Standby-Datenbank
  5. SQL-Apply zum Nachfahren der Transaktionen
  6. Switchover (Beginn der Downtime für die Applikation)
  7. Migration der nicht-unterstützten Objekte (anschl. Ende der Downtime)
  8. Flashback der (ehemaligen) Primär-Datenbank zum Restore-Point (Schritt 2.) (Single-Instance-Modus)
  9. Konvertierung der ehemaligen Primär-DB zu einer physischen Standby-Datenbank
  10. Redo-Apply starten



Quelle: mikedietrichde.com / Präsentation "Upgrade to 19c"

Transient Logical Standby Database (3)

trivadis

- **Vorteile**
  - Kurze Downtime
- **Nachteile**
  - Komplexität
  - Logische Standby-Datenbanken unterstützen nicht alle Datentypen
    ➔ EDS (extended datatype support) als Workaround
    ➔ oder separate Migration dieser Tabellen

DBA_LOGSTDBY_UNSUPPORTED_TABLE	All unsupported tables
DBA_LOGSTDBY_UNSUPPORTED	All unsupported columns
DBA_LOGSTDBY_NOT_UNIQUE	Tables without PK or Unique Key

Transient Logical Standby Database (4)

trivadis

- In unserem Fall ..
  - Keine der Datenbanken kann zu 100% mittels transienter logischer Standby-Datenbank aktualisiert werden
  - Auch nach Housekeeping ("CHAINED_ROWS", "PLAN_TABLE", Schema "SQLTXPLAIN", nicht mehr benötigte Tabellen etc.) verbleiben nicht unterstützte Tabellen (Hauptproblem: fehlende PKs/UKs) ➔ Änderungen am DB-Modell wären erforderlich

- **Einhalten der 30 Minuten SLA-Grenze beim Upgrade eher unwahrscheinlich**

Transportable Tablespaces (1)

trivadis

- **Überblick**
  - Zieldatenbank in gewünschter Version anlegen
  - Metadaten der Ausgangsdatenbank werden exportiert (DB im Read-Only-Modus)
  - Datendateien werden auf das Zielsystem transferiert
  - Metadaten werden auf dem Zielsystem importiert und die Datendateien werden "eingehängt"

- Funktioniert auch "Cross-Endianess" (xTTS)

Quell-DB: 11.2.0.3+
Ziel-DB: 12.1+

- "Transportable Database" (Full Transportable Export/Import) unterstützt seit Oracle 12c TTS für die komplette Datenbank (alle Applikationstablespaces)

Transportable Tablespaces (2)

trivadis

- **Ablauf**
  1. Benutzer in der Zieldatenbank anlegen
  2. Zu transferierende Tablespaces in den Read-Only-Modus setzen (➔ Downtime der Applikation)
  3. Metadaten exportieren
  4. Datendateien auf den Zielserver transferieren
     ➔ kann bereits parallel zum laufenden Betrieb der Ausgangsdatenbank erfolgen (inkrementelle Backups)
  5. Dump-Datei des Metadaten-Exports importieren
  6. (Optional) Tablespaces in der Ausgangsdatenbank wieder in den Read-Write-Modus setzen
  7. Sequenzen, Synonyme, PL/SQL-Objekte, Grants transferieren
  8. Ende der Downtime

Transportable Tablespaces (3)

trivadis

- **Vorteile**
  - Daten können im Vorfeld transferiert werden (inkrementelle Backups)
  - Tool-Unterstützung (siehe MOS-Note 1389592.1)
- **Nachteile**
  - PL/SQL-Code, Sequenzen, Synonyme, Grants, Objekte im SYSTEM-Tablespace .. müssen separat transferiert werden
  - Testmode erst ab Oracle 19c (=> in unserem Fall nicht einsetzbar)
- **Insgesamt**
  - Gut geeignet für "einfache", aber große Datenbanken
    ➔ Große Tabellen, wenig Objekte die separat transferiert werden müssen

Transportable Tablespaces (4)

trivadis

- In unserem Fall ..
  - Kein einfacher Test des Metadaten-Exports möglich
    ➔ für Tests zusätzliche Infrastruktur erforderlich
  - Aufwand für die "Nacharbeiten" relativ hoch

- **Einhalten der 30 Minuten SLA-Grenze beim Upgrade eher unwahrscheinlich**

trivadis

Zwischenstand

Die Diskussionen laufen noch … (1)

trivadis

- Anschaffung weiterer Hardware zur Absicherung der Failover-Fähigkeit?

- Sind die 30 Minuten maximale Downtime in Stein gemeißelt?

- Datenbank-Upgrade
  - Transiente Logische Standby-DB:
    - insgesamt zu komplex (Komplexität des Verfahrens allgemein, viele nicht unterstützte Tabellen)
  - Transportable Tablespaces:
    - kann mit der vorhandenen Hardware bzw. den vorhandenen DB-Versionen nicht einfach getestet werden

Die Diskussionen laufen noch … (2)

trivadis

- **Stand jetzt ist die Einhaltung der "30-Minuten-Schallgrenze" auch mit einem der komplexeren, aber ggf. schnelleren Verfahren eher unwahrscheinlich**

- Wenn wir die 30-Minuten ohnehin nicht einhalten können, warum dann nicht ein einfaches Upgrade-Verfahren?
  ➔ "massiv-paralleler Upgrade mit dbupgrade"
  ➔ bei kleinen Datenbanken ggf. DataPump via Network-Link

K.I.S.S.

Keep it simple, stupid

trivadis

Weitere Informationen

Weitere Informationen

trivadis

- MAA Whitepaper: Database Rolling Upgrades for Physical Standby Databases using Transient Logical Standby 11g (Doc ID 1672625.1)
- SQL Apply Extended Datatype Support - 11.2 (Doc ID 949516.1)
- Rolling upgrade using DBMS_ROLLING - Complete Reference (Doc ID 2086512.1)
- Oracle11g Data Guard: Database Rolling Upgrade Shell Script (Doc ID 949322.1)

- Information Center: Transportable Tablespaces (TTS) for Oracle Database (Doc ID 1461278.2)
- How to Create Transportable Tablespaces Where the Source and Destination are ASM-Based (Doc ID 394798.1)
- Master Note for Transportable Tablespaces (TTS) -- Common Questions and Issues (Doc ID 1166564.1)
- Using Rman Incremental backups To Update Transportable Tablespaces. (Doc ID 831223.1)
- 11G - Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup (Doc ID 1389592.1)

Fragen & Antworten

Markus Flechtner

markus.flechtner@trivadis.com

Telefon +49 211 5866 64725

@markusdba

www.markusdba.de


Eine WELT ermöglichen,
in der intelligente IT
LEBEN und ARBEITEN
völlig selbstverständlich
erleichtert.