



ORACLE®

Best Practices for Zero Risk, Zero Downtime Database Maintenance

Joseph Meeks
Director,
Product Management
Oracle USA

Joydip Kundu
Director,
Software Development
Oracle USA

Michael T. Smith
Principal Member of
Technical Staff
Oracle USA

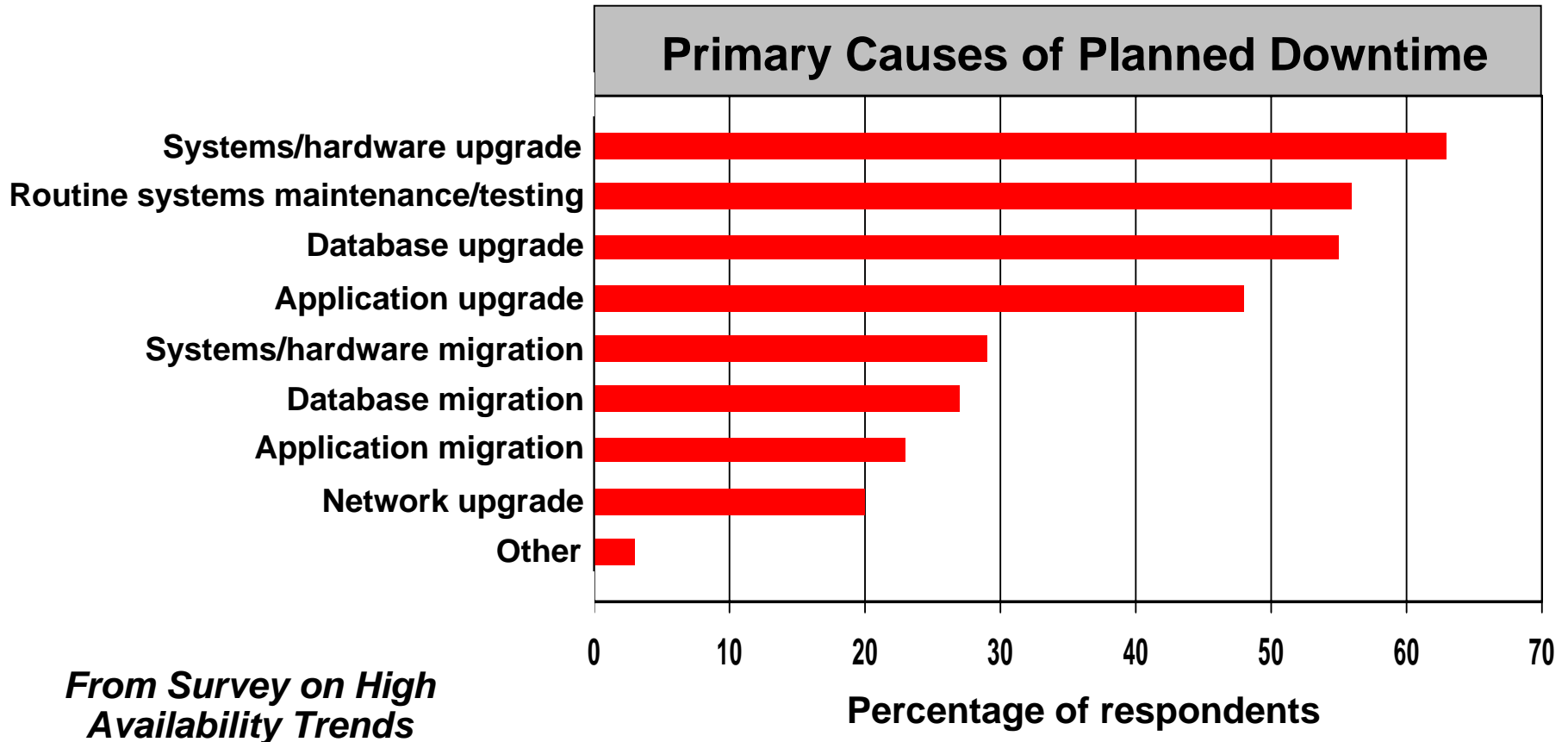
Program

- **Planned Downtime: Bad News / Good News**
- How Data Guard Minimizes Planned Downtime
- Tips & Tricks from Oracle Development
- Oracle In-Memory Database Cache: All the way to ZERO
- Wrap-up & Resources

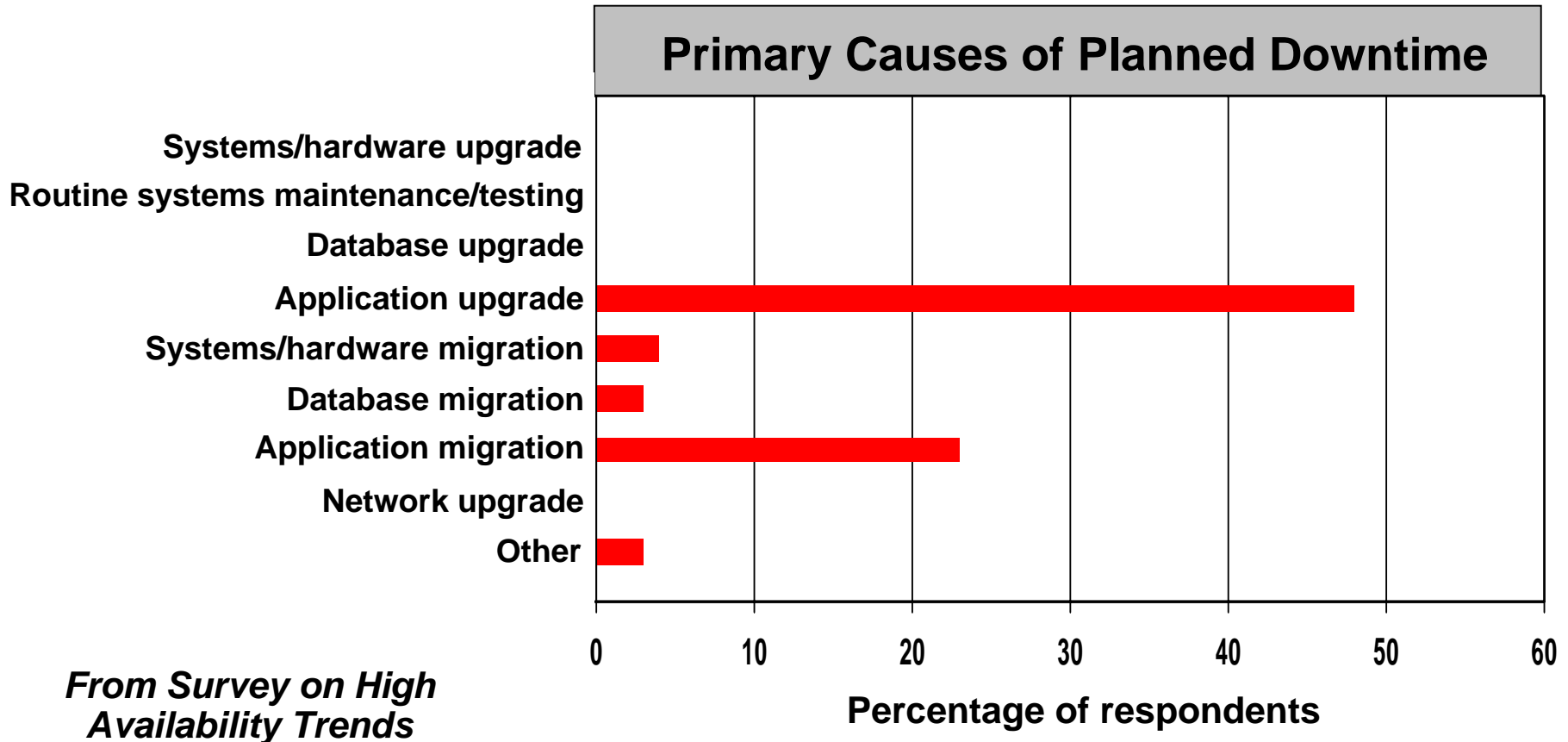


Bad News: Lots of Problems

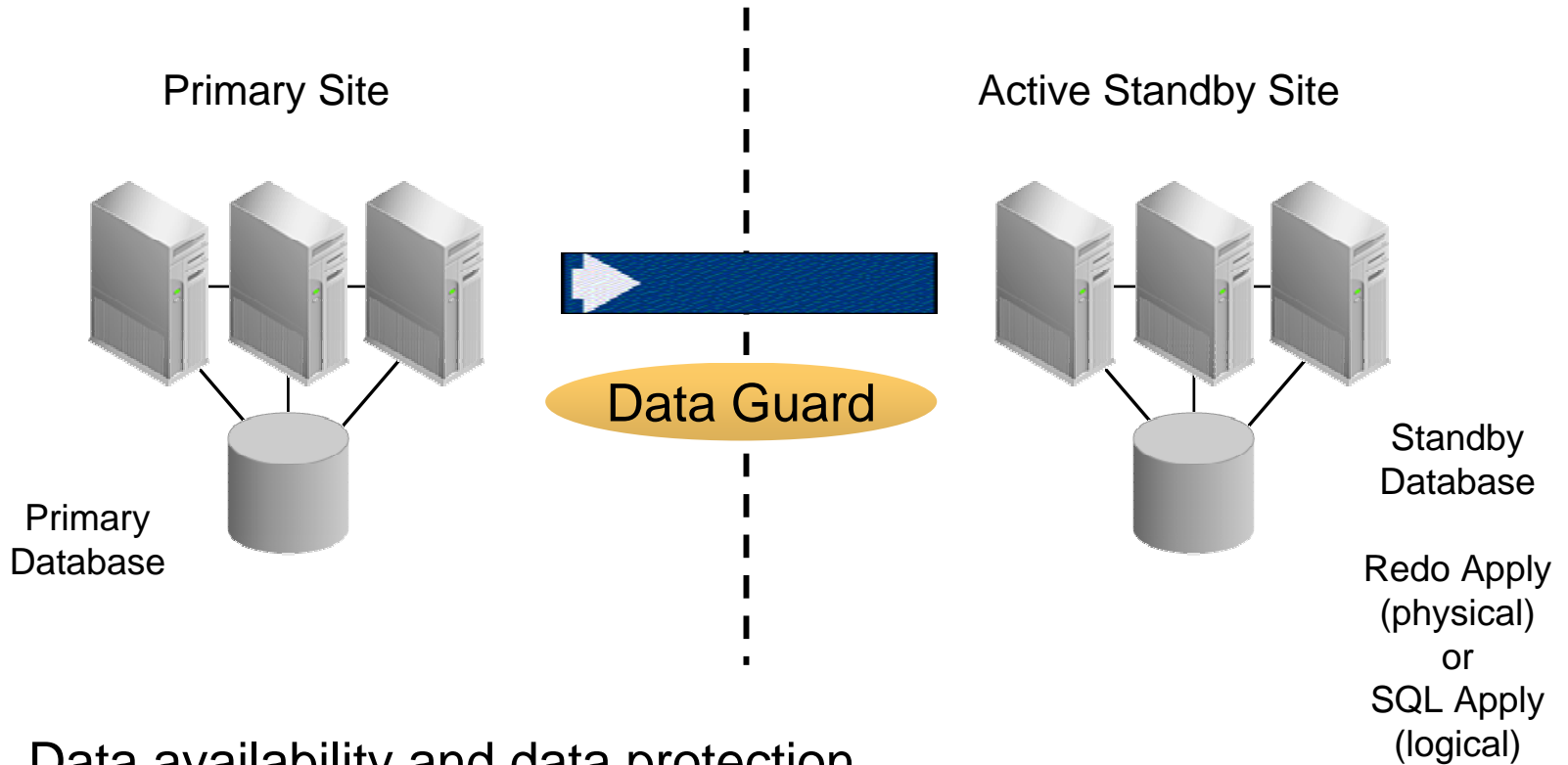
“80 Percent of all Downtime is Planned Downtime”



Good News: You Can Make Life Less Challenging Using a Tool You Already Own!



Oracle Data Guard



- Data availability and data protection
- Loosely coupled architecture
- Ideal for minimizing planned downtime

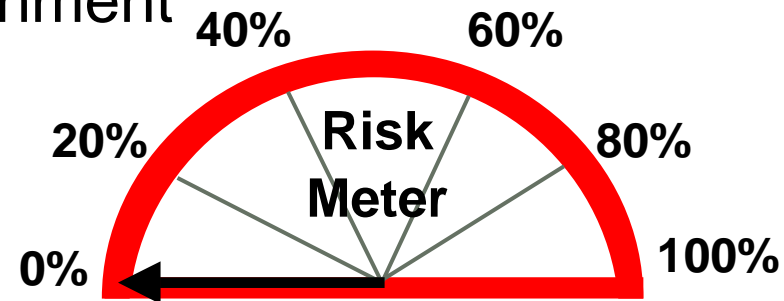
Reducing Planned Downtime with Data Guard

High Level Overview

Availability

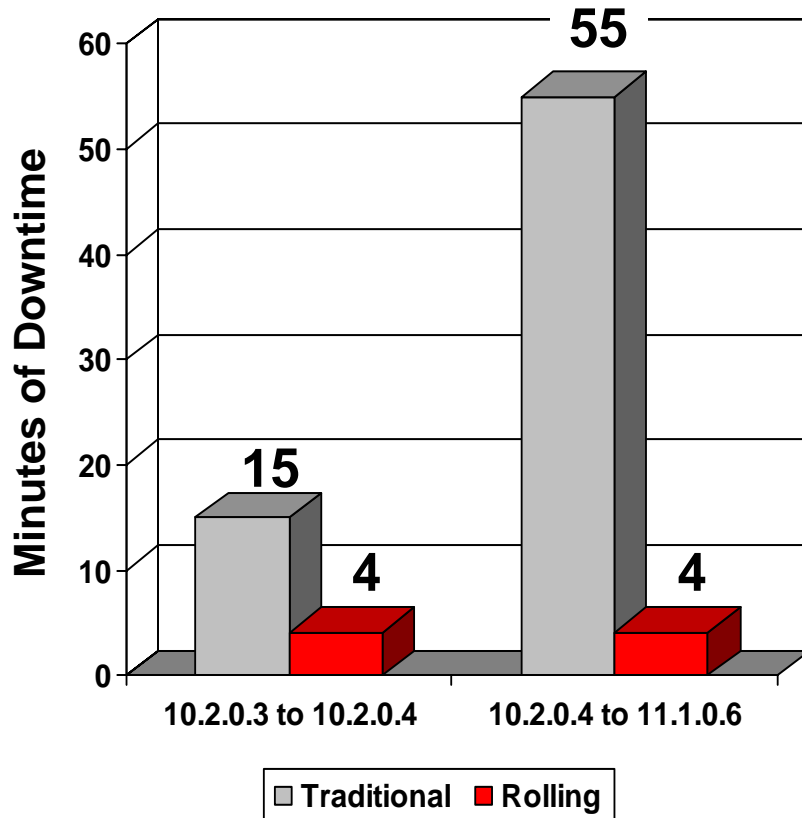


1. Data Guard primary with standby database
2. Defer redo shipping
3. Implement changes at standby database
4. Data Guard resynchronizes standby with primary
5. Validate changes
6. Switchover
7. Production on new environment



United Parcel Service

Database Rolling Upgrade Reduced Downtime by 93%



- UPS planned maintenance
 - Downtime SLA < 15 minutes
 - Oracle Database 11g upgrade tests showed that traditional upgrade methods would not meet SLA
- Data Guard Rolling Upgrade Proof of Concept
 - Total application downtime < 4 minutes

Bielefeld University – Germany

Database Rolling Upgrade using Physical Standby

“Upgrade to Oracle Database 11g with the full HA stack and patch in 2 minutes – That’s no joke.”

Dr. Lars Koller, Bielefeld University

- Physical standby user (transient logical standby)
- Total downtime less than 2 minutes



Oracle IT

Using Data Guard to Reduce Planned Downtime

- For all mission critical systems
 - oracle.com
 - Internal business applications
 - Internal collaboration applications - Beehive
 - Oracle code repository and release management
 - Internal Enterprise Manager Grid Control repository
- Example: Oracle Beehive
 - 32bit to 64bit migration
 - Migrating to ASM
 - Testing new features – e.g. flashback database
 - Migrating to Exadata storage
 - Regular bi-monthly switchovers to test standby readiness

Program

- Planned Downtime: Bad News / Good News
- **How Data Guard Minimizes Planned Downtime**
- Tips & Tricks from Oracle Development
- Oracle In-Memory Database Cache: All the way to ZERO
- Wrap-up & Resources



Two Areas Where Data Guard Helps

- Technology refresh and migrations
 - Systems, operating system, network, data center
 - Implementing major database changes
 - Objective: move from old environment to new

- Database rolling upgrades
 - Upgrading to new database release or patchset
 - Use either logical or physical standby
 - Objective: upgrade primary and standby to new Oracle version

Technology Refresh and Migrations

Using Redo Apply – Physical Standby

- Data center moves
- Technology refresh
- 32bit ↔ 64bit
- Windows ↔ Linux
- Single node → Oracle RAC
- Migrating to ASM
- Testing new features – e.g. flashback database
- Migrating to Exadata storage
- Operating system and/or hardware maintenance on single node (non-RAC) databases

Technology Refresh and Migrations

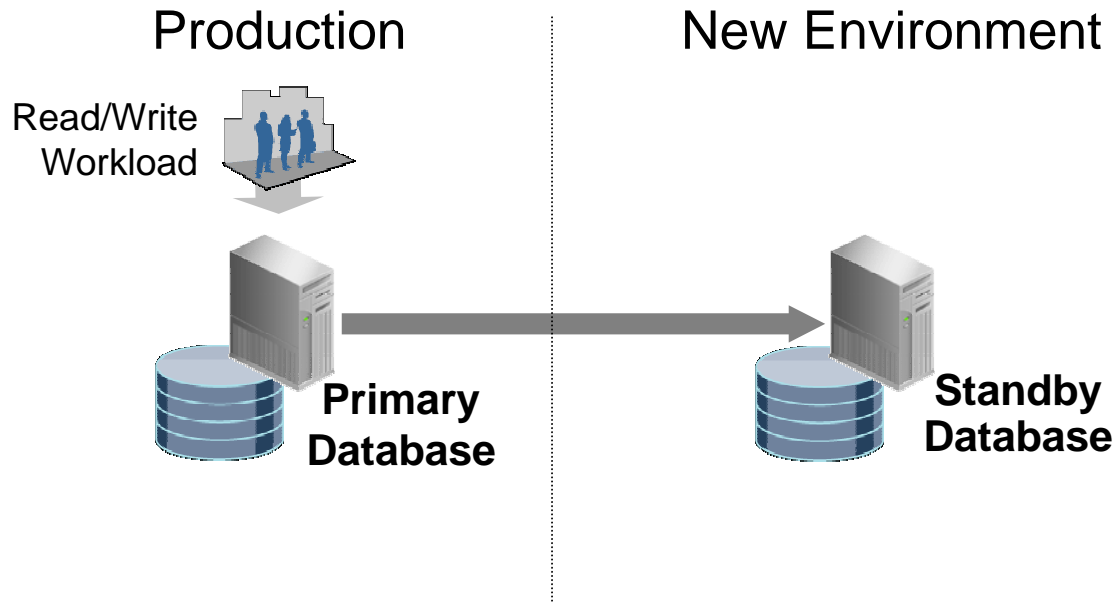
Using SQL Apply – Logical Standby

- Various database changes
 - Example: ASSM, initrans, blocksize ...
- Index and storage changes
- Implementing Advanced Compression (11.2)
- Migrating to Secure Files (11.2)
- Migrating to Exadata - when changing database extent size

Technology Refresh and Migrations

Step 1

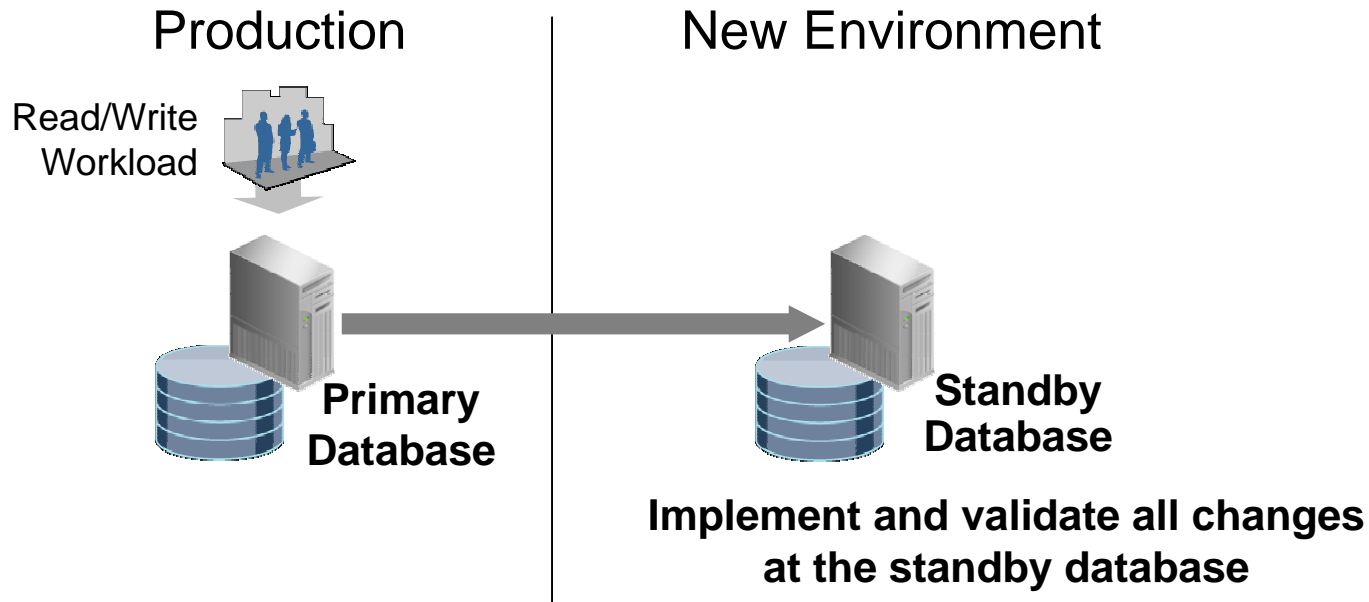
- Deploy a standby database on the new environment



Technology Refresh and Migrations

Step 2

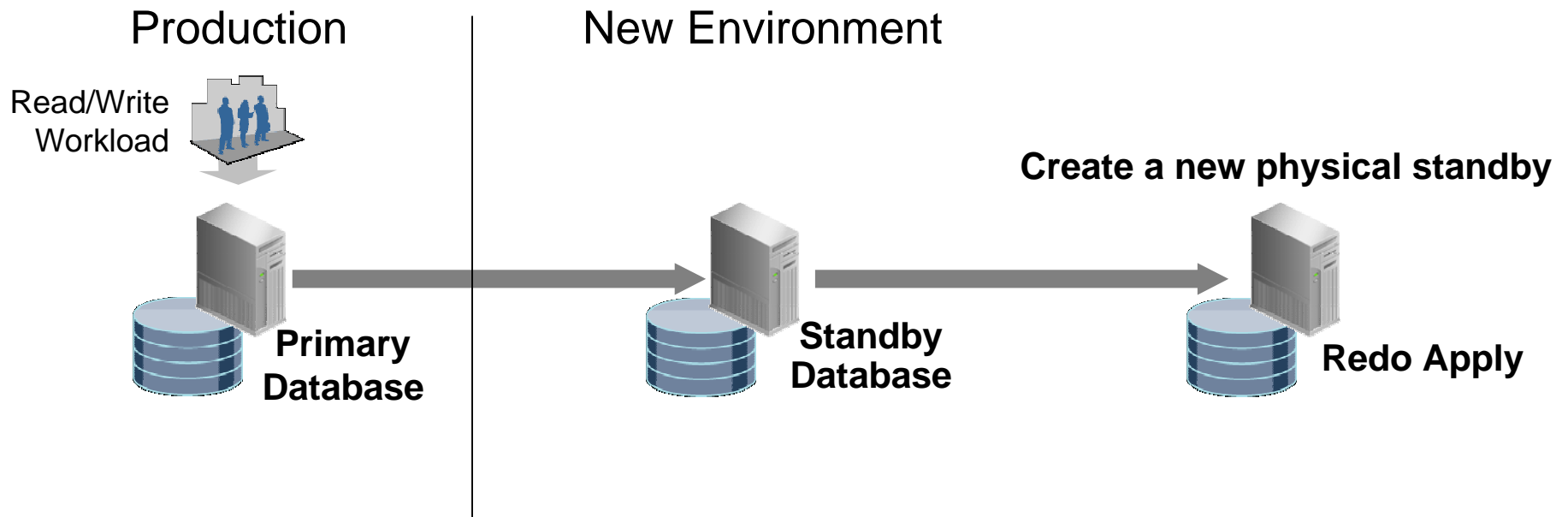
- Implement any additional changes at the standby database
- Validate changes using Real Application Testing



Technology Refresh and Migrations

Step 3

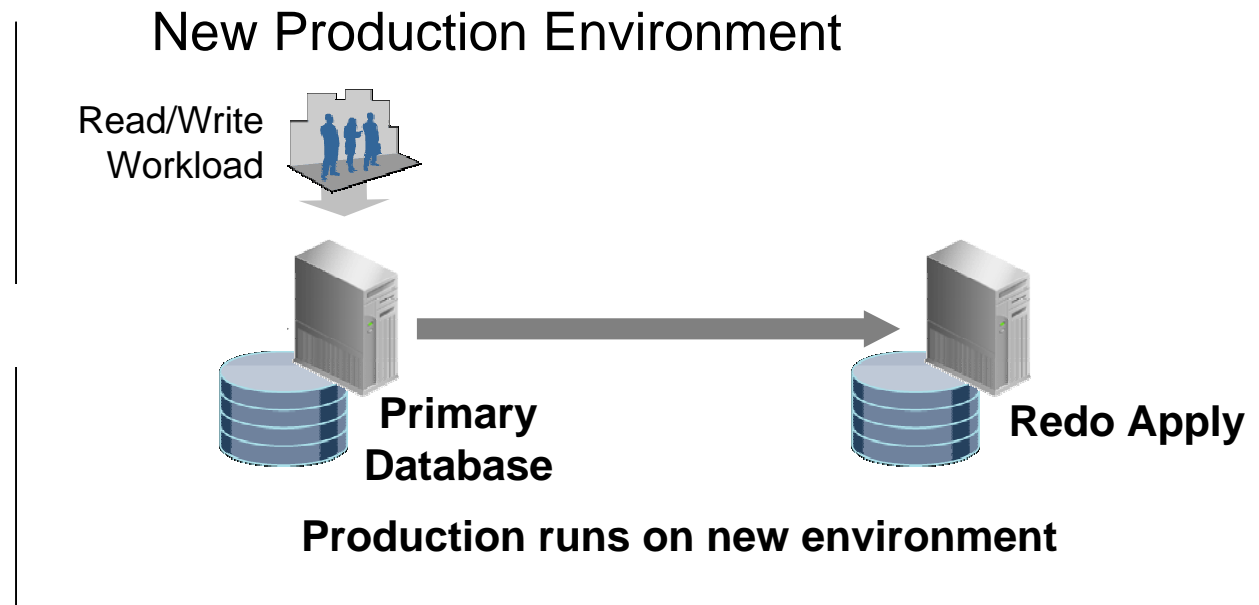
- Create a standby database for the new environment



Technology Refresh and Migrations

Step 3

- Create a standby database for the new environment
- Switch production over to the new environment
 - Retire the old primary



Two Areas Where Data Guard Helps

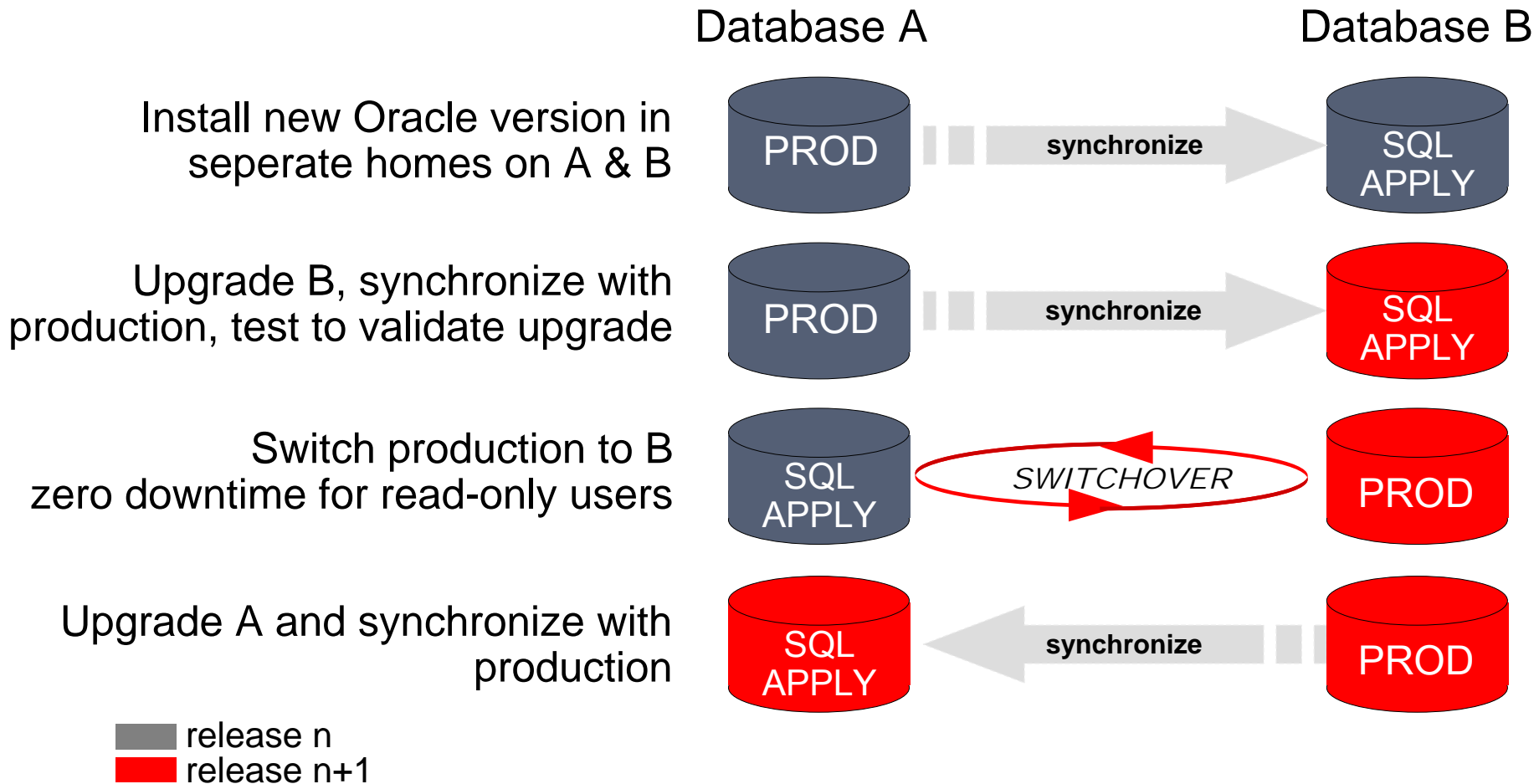
- Technology refresh and migrations
 - Systems, operating system, network, data center
 - Implementing major database changes
 - Objective: move from old environment to new
- Database rolling upgrades
 - Upgrading to new database release or patchset
 - Use either logical or physical standby
 - Objective: upgrade primary and standby to new Oracle version

Database Rolling Upgrades

- Patch-sets and new database releases
- One-off patches for single node (non-RAC) databases
- Using SQL Apply
 - Any upgrade from 10.1.0.3 onward
- Using Redo Apply – Transient Logical Standby
 - Any upgrade from 11.1.0.7 onward
 - Added convenience for physical standby users
 - No need to create new logical standby just for upgrade

Database Rolling Upgrades

Using SQL Apply - Logical Standby



Database Rolling Upgrades

Using Redo Apply – Transient Logical Standby

Install new Oracle version in separate homes on A & B, set guaranteed restore point (GRP) on A

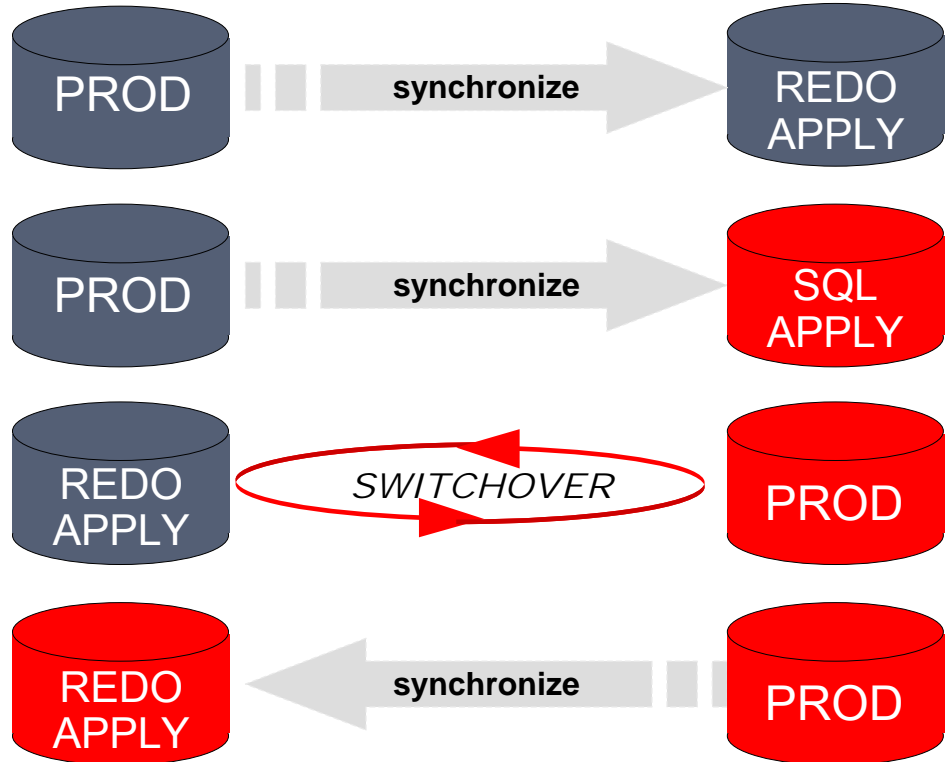
Convert B to transient logical, upgrade to new version and sync

Switchover, flashback A to GRP, mount in new/upgraded home

Upgrade A via redo stream and synchronize

Database A

Database B



release n
release n+1

Switchover is Good

- Switchover is a planned, zero data loss operation
- Guarantees standby data can not be modified independent of primary transactions
 - No split brain
 - No data corruptions
- Enforced by standby database Role
 - Write access to primary data is only available to Data Guard apply processes at the standby database
- Straightforward to execute
 - Via mouse click or simple command

Switchover

What Happens at a Database Level?

Redo Apply

- Primary drains redo pipe
- Standby applies all redo
- Flips a bit in the control file
 - Changes role to primary
- Opens standby as primary
 - No bounce required
- Done

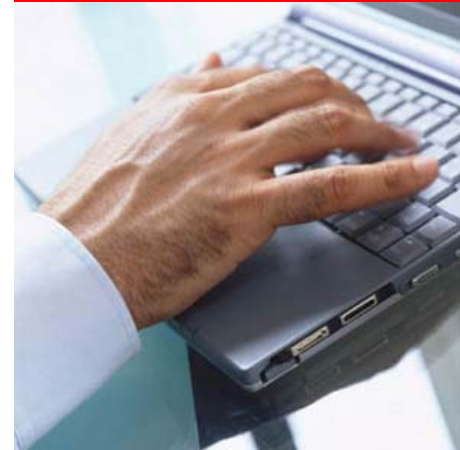
SQL Apply

- Primary drains redo pipe
- Standby applies all redo
- Flips a bit in the control file
 - Turns off the Guard
 - Changes role to Primary
- Done

Note: No open required since logical standby is already open read-write

Program

- Planned Downtime: Bad News / Good News
- How Data Guard Minimizes Planned Downtime
- **Tips & Tricks from Oracle Development**
- Oracle In-Memory Database Cache: All the way to ZERO
- Wrap-up & Resources



Database Rolling Upgrade Process

- Conceptually simple, but..
 - For a many DBAs, this would be their first encounter with a logical standby
 - Setting of parameters for performance
 - Iterative testing
 - Several parameters are specific for rolling upgrade
 - record_unsupported_operations
 - log_auto_delete must be FALSE etc..
 - Publishing services following the switchover
 - Rolling upgrade is a process, not a DDL

NEW: Rolling Upgrade Automated Workflow

Oracle Database 11g Transient Logical Standby

- Shell script available via MetaLink Note 949322.1
- Incorporates best practices for
 - Transient rolling upgrade (physical standby databases)
 - SQL Apply
 - Switchover
- Features
 - Automatic migration of OCI clients using server-side TAF
 - Can restart script after failure
 - Ability to abandon rolling upgrade mid-way
 - Customizable (by definition)
 - Repeatable testing (by definition)

Automated Workflow

Four Phases

- Phase 1: Prepare
 - Make sure flashback database is on
 - Convert target physical standby to a logical standby
- Phase 2: Upgrade Standby and Synchronize
 - User upgrades the logical standby to the target version
 - Synchronize with the primary with SQL Apply running at the higher version of the RDBMS software
- Phase 3: Switchover
 - Switchover to the logical standby
- Phase 4: Upgrade Original Primary
 - Flashback original primary so it becomes a physical
 - Follow the new primary and get upgraded automatically

Phase 1: Prepare

```
Phase 1: Convert standby to logical  
Continue? [y/n]: y
```

```
Flashback enabled on both databases.
```

```
Converting BOSTON to a logical standby
```

```
Checking status of managed recovery process (MRP0) on BOSTON  
MRP0 is running on BOSTON
```

```
Stopping MRP0 on BOSTON ...
```

```
MRP0 successfully stopped on BOSTON
```

```
Taking Guaranteed Restore Points on CHICAGO and BOSTON ...
```

```
ru_primary_before_upgrade (GRP) taken on CHICAGO
```

```
ru_stdby_before_upgrade (GRP) taken on BOSTON
```

Phase 1: Prepare - continued

```
Taking data dictionary snapshot in the redo stream on CHICAGO ...  
Dictionary snapshot successfully logged in the redo stream  
Thread 1, sequence 7 contains the data dictionary snapshot
```

```
Converting BOSTON to a logical standby ...  
Issuing alter database recover to logical standby keep identity ...  
BOSTON successfully converted to a logical standby
```

```
Setting SQL Apply parameters ...  
MAX_SGA set to 500M  
MAX_SERVERS set to 50  
RECORD_UNSUPPORTED_OPERATIONS set to TRUE  
LOG_AUTO_DELETE set to FALSE
```

```
Starting SQL Apply on BOSTON ...  
SQL Apply successfully started
```

```
Waiting for dictionary load to complete on BOSTON ...  
Dictionary load progress 40% done  
Dictionary load completed  
  
BOSTON is currently 5 minutes behind CHICAGO  
Estimated catch-up time 1 minute
```

```
Phase 1 of rolling upgrade successfully completed
```

Phase 2: Upgrade Standby and Synchronize

```
Phase 2: Ready to upgrade the logical standby  
Continue? [y/n]: y
```

```
Stopping SQL Apply in BOSTON ...  
SQL Apply successfully stopped.
```

```
Please upgrade BOSTON using Upgrade instructions provided in  
Oracle Upgrade Guide of the target version. Once you have upgraded  
BOSTON, please say "y" to continue.
```

```
Continue? [y/n]: y
```

```
Checking for target database upgrade software version...  
BOSTON is running Oracle Version 11.2.0.1  
with redo compatibility set to 11.0
```

Phase 2: Upgrade Standby and Synchronize - continued

```
Starting SQL Apply on BOSTON ...  
SQL Apply successfully started.
```

```
BOSTON is currently 55 minutes behind CHICAGO  
Estimated catch up time is 20 minutes
```

```
Checking DBA_LOGSTDBY_EVENTS for unsupported operations ...  
No unsupported operations found
```

```
Waiting (will check again in 5 minutes)...
```

```
BOSTON is currently 35 minutes behind CHICAGO  
Estimated catch up time is 10 minutes
```

```
Checking DBA_LOGSTDBY_EVENTS for unsupported operations ...  
No unsupported operations found
```

```
Waiting (will check again in 5 minutes) ...
```

Phase 2: Upgrade Standby and Synchronize - continued

BOSTON is currently 30 seconds behind CHICAGO

Estimated catch up time is 3 seconds

Checking DBA_LOGSTDBY_EVENTS for unsupported operations ...

No unsupported operations found

BOSTON is within 2 minutes of CHICAGO

DBA_LOGSTDBY_EVENTS table show no unsupported operations

Phase 2 of rolling upgrade successfully completed

Phase 3: Switchover

```
Phase 3: Switchover to upgraded logical standby  
Continue? (y/n): y
```

```
Starting switchover sequence: BOSTON will become the new primary
```

```
Checking viability of switchover ...  
Checked V$DATABASE.SWITCHOVER_STATUS on CHICAGO: ok for switchover  
Checked V$ARCHIVE_DEST on CHICAGO: ok for switchover  
Checked V$DATAGUARD_STATS on BOSTON: ok for switchover  
Sentinel DML reflected on BOSTON: ok for switchover  
Switchover operation deemed viable
```

```
Stopping services on CHICAGO ...  
All services stopped
```

```
Disconnecting user sessions on CHICAGO ...  
All user sessions disconnected
```

```
Switching over CHICAGO to logical standby role ...  
CHICAGO is now logical standby
```

Phase 3: Switchover - continued

Switching over BOSTON to the primary database ...
BOSTON is now the primary database

OCI-based services using server-side TAF have been restarted on BOSTON

Shutting down CHICAGO for upgrade ...
shutdown completed

Phase 3 of rolling upgrade successfully completed

Phase 4: Upgrade Original Primary

```
Phase 4: Upgrade the original primary
Continue? (y/n): y
```

```
Mounting CHICAGO for flashback ...
Database mounted
```

```
Flashing back CHICAGO to Guaranteed Restore Point (ru_primary_before_upgrade)...
Flashback completed
```

```
Converting CHICAGO back to a physical standby...
CHICAGO is now a physical standby of BOSTON
```

```
Starting media recovery on CHICAGO ...
MRP0 successfully started on CHICAGO
```

```
CHICAGO is currently 75 minutes behind BOSTON
Estimated catch up time is 30 minutes
```

```
Waiting..
```

Phase 4: Upgrade Original Primary - continued

CHICAGO is currently 35 minutes behind BOSTON
Estimated catch up time 15 minutes

```
Checking to see if CHICAGO has mined through upgrade redo...  
CHICAGO has now been upgraded automatically to the 11.2.0.1 release
```

```
Phase 4 of rolling upgrade completed
```

```
Continue to drop all Guaranteed Restore Points and other metadata? {y/n} y
```

```
Removing Guaranteed Restore Points ...  
Restore points removed
```

```
Dropping tables and triggers created as part of rolling upgrade process ...  
All supporting tables and triggers dropped
```

```
Rolling upgrade process completed.
```

```
Current configuration: primary database is BOSTON, standby database is CHICAGO
```

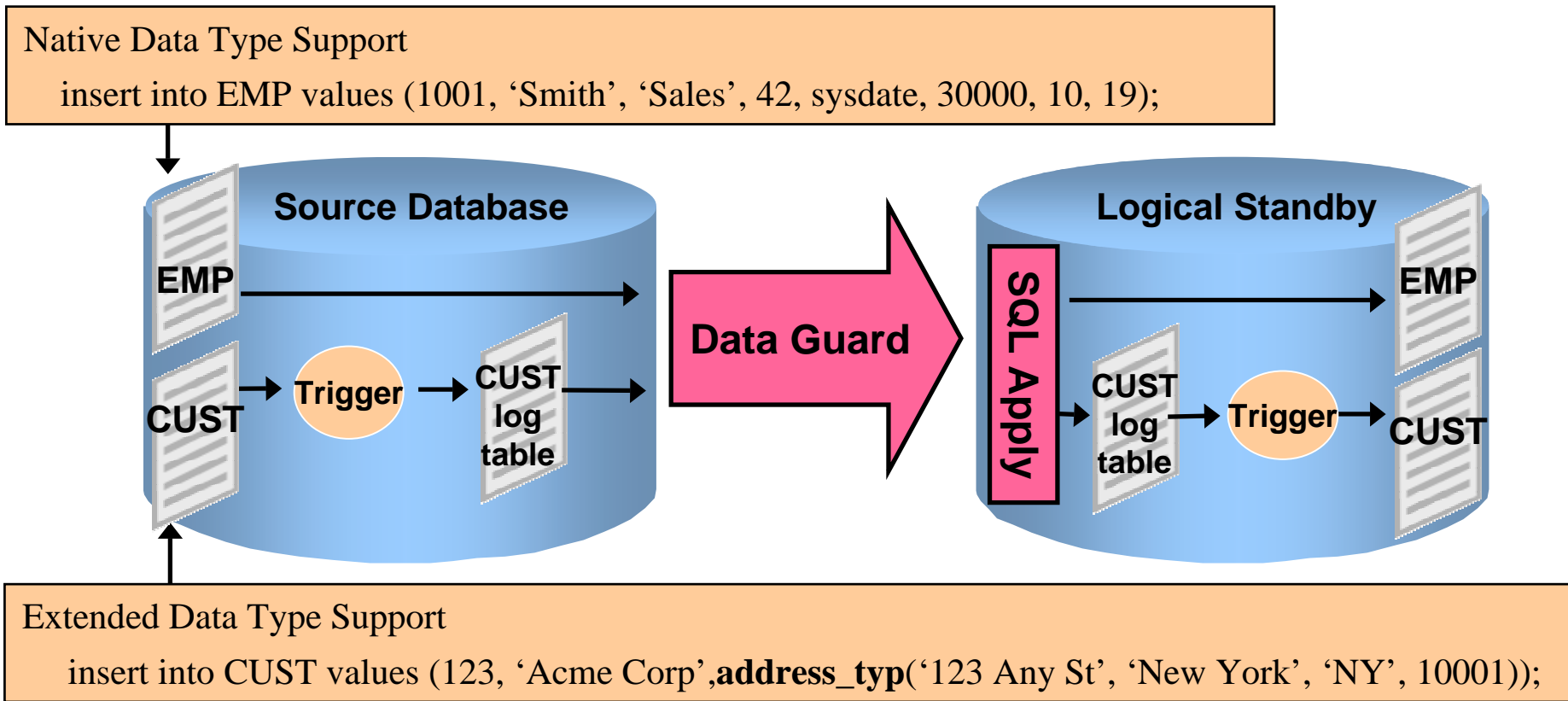
```
You can do a switchover if you want to get back to your original configuration.  
Rolling Upgrade Workflow will now exit.
```

Considerations When Using SQL Apply

- Requires primary key or unique index
- DDLs are applied serially
 - Work around by not issuing maintenance DDLs during upgrade window
- Batch updates are expensive
 - Do not run batch jobs during rolling upgrade window
- Restart in the middle of a large load can be expensive
 - SQL Apply allows parameter changes without having to stop
- See MetaLink Note 603361.1
 - Developer and DBA Tips for Pro-Actively Optimizing SQL Apply
- Extended Datatype Support is required for unsupported types
 - SQL Apply does not support native redo-based replication of all datatypes
 - SDO_GEOMETRY, XML-OR/Binary/XML, objects and collections

SQL Apply Extended Datatype Support

Data Guard 11g Release 1



Provides sample triggers and log table definitions for several unsupported datatypes

Customers can follow examples and implement extended support for datatypes not currently supported by SQL Apply

Best Practices for Extended Datatype Support in Oracle 11g Release 1: MetaLink Note 559353.1

SQL Apply Extended Datatype Support

Enhanced for Data Guard 11g Release 2 (DBMS_LOGSTDBY)

- eds_add_table
 - Automatically generates log table definitions and triggers for tables containing type SDO_GEOMETRY
 - For transient logical, call at primary database
 - replicated to physical standby via redo stream
 - If logical standby, call at the primary first and then at logical
- eds_drop_table
 - Calling at the primary will automatically drop log tables and triggers from all standbys

SQL Apply Extended Datatype Support

Enhanced for Data Guard 11g Release 2 (DBMS_LOGSTDBY)

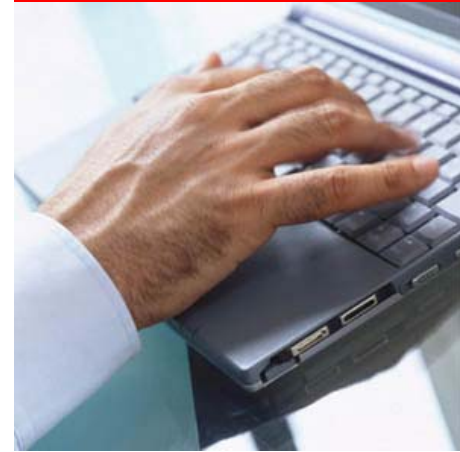
- Switchover-friendly
- Does not (yet) handle DDLs transparently
- DBA_LOGSTDBY_EDS view
- Additional data type support to follow:
 - XML-OR/Binary/XML in a future patchset
 - Subsequent patchsets will address objects and collections
- As native redo based support is added in future releases, you will be able to transparently transition to native support with zero downtime
- Documented in MetaLink Note 949516.1: SQL Apply Extended Datatype Support – Data Guard 11g Release 2

Considerations When Using EDS

- Applicable if rate of change to EDS-replicated tables is manageable
 - Every dml on an EDS replicated table fires a trigger that initiates a second dml on its shadow table
- DDL that changes the shape of the table, or adds unique indexes must be DBA controlled
 - so that you can take compensating actions

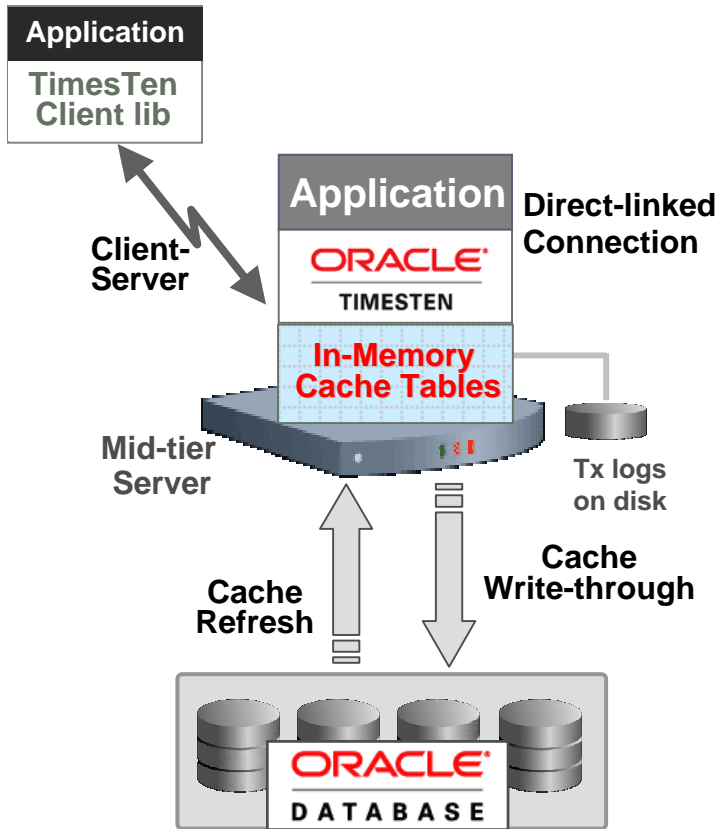
Program

- Planned Downtime: Bad News / Good News
- How Data Guard Minimizes Planned Downtime
- Tips & Tricks from Oracle Development
- **Oracle In-Memory Database Cache: All the way to ZERO**
- Wrap-up & Resources



Oracle TimesTen In-Memory Database Cache

An Oracle Database Option

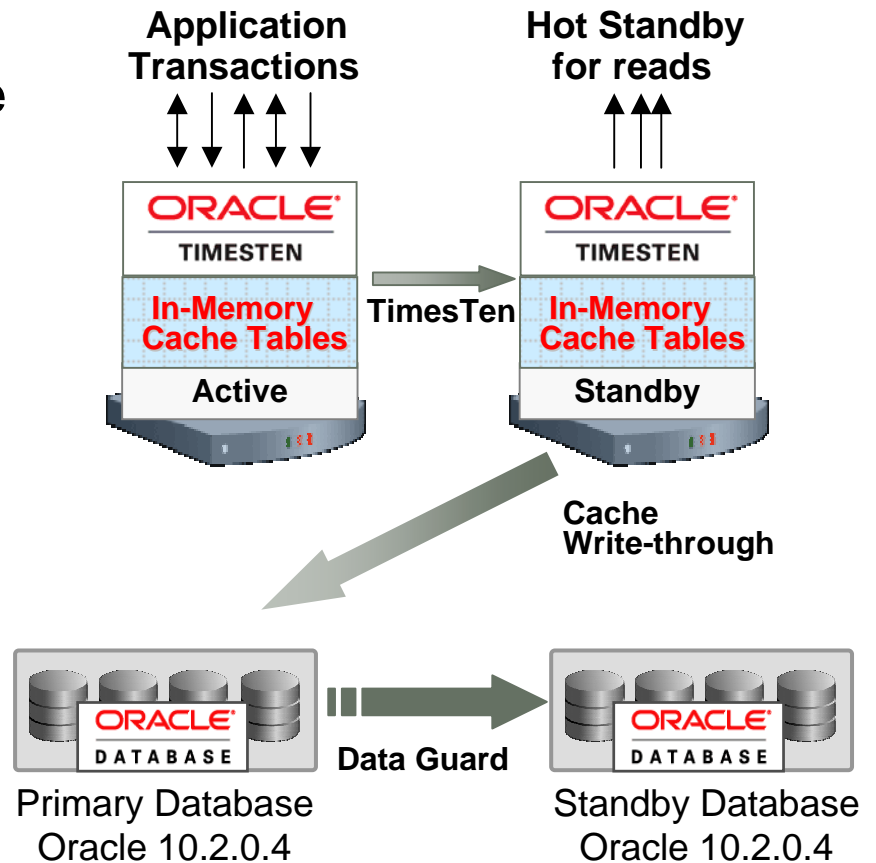


- High-performance, in-memory database cache
 - Cache individual tables and related tables
 - Cache all or subset of rows and columns
- Read-only and updateable caches
 - Accessed as regular SQL database tables
 - Joins/search, insert/update/delete
- Automatic data synchronization
 - TimesTen to Oracle
 - Oracle to TimesTen
- Supports real-time transactional replication
 - Between two in-memory cache databases (active-standby)

Zero Downtime Database Upgrade

Availability

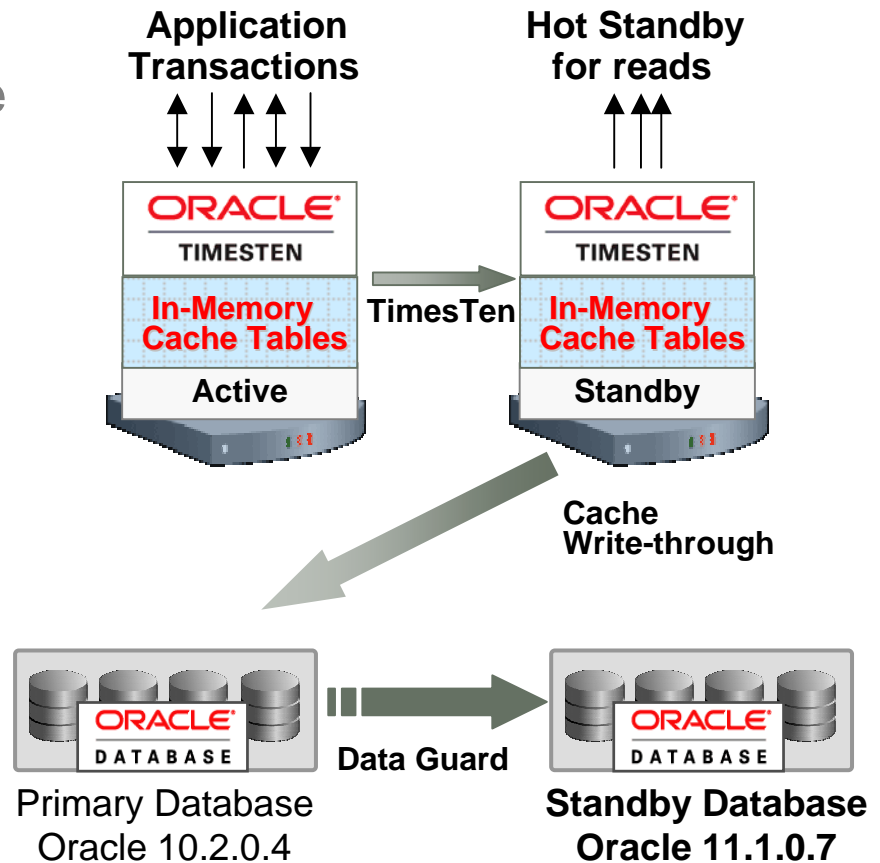
- Applications access cache
- Cache replicated for HA
- Cache write-through to Oracle Database



Zero Downtime Database Upgrade

Availability

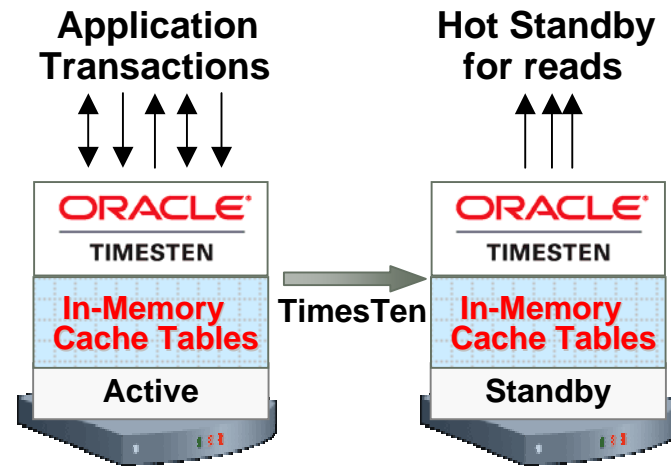
- Applications access cache
- Cache replicated for HA
- Cache write-through to Oracle Database
- Upgrade Standby Database



Zero Downtime Database Upgrade

Availability

- Applications access cache
- Cache replicated for HA
- Cache write-through to Oracle Database
- Upgrade Standby Database
- Switchover



Standby Database
Oracle 10.2.0.4

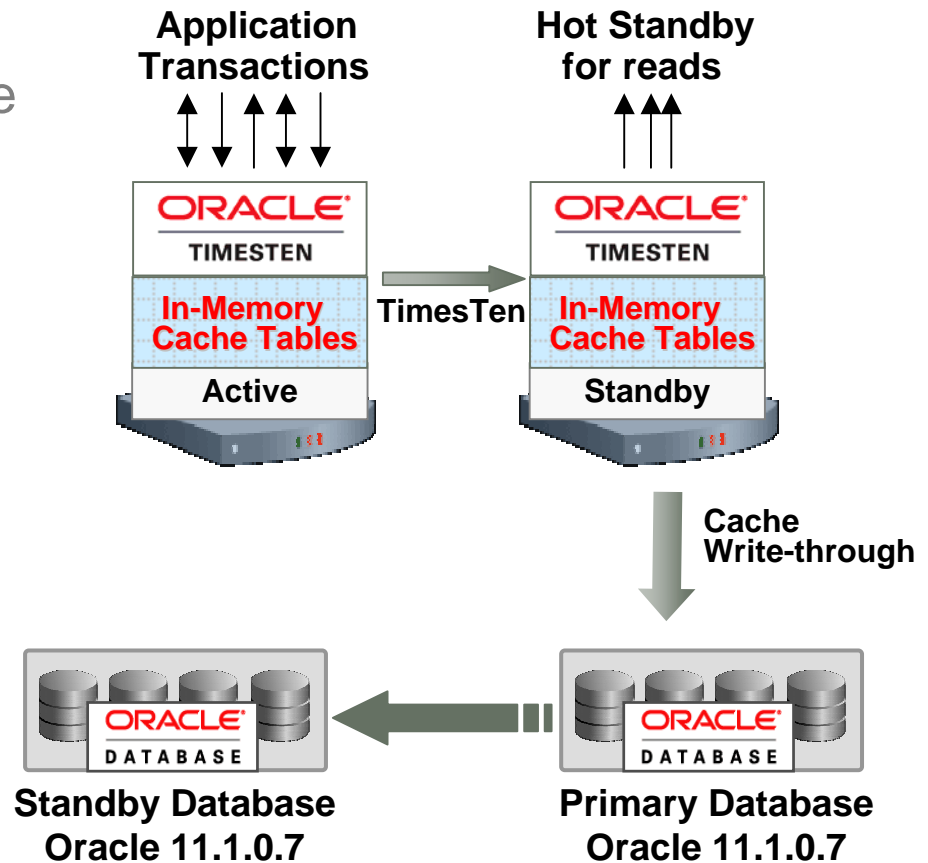


Primary Database
Oracle 11.1.0.7

Zero Downtime Database Upgrade

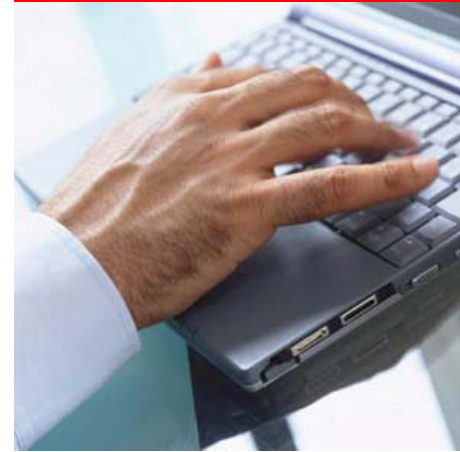
Availability

- Applications access cache
- Cache replicated for HA
- Cache write-through to Oracle Database
- Upgrade Standby Database
- Switchover
- Upgrade original primary



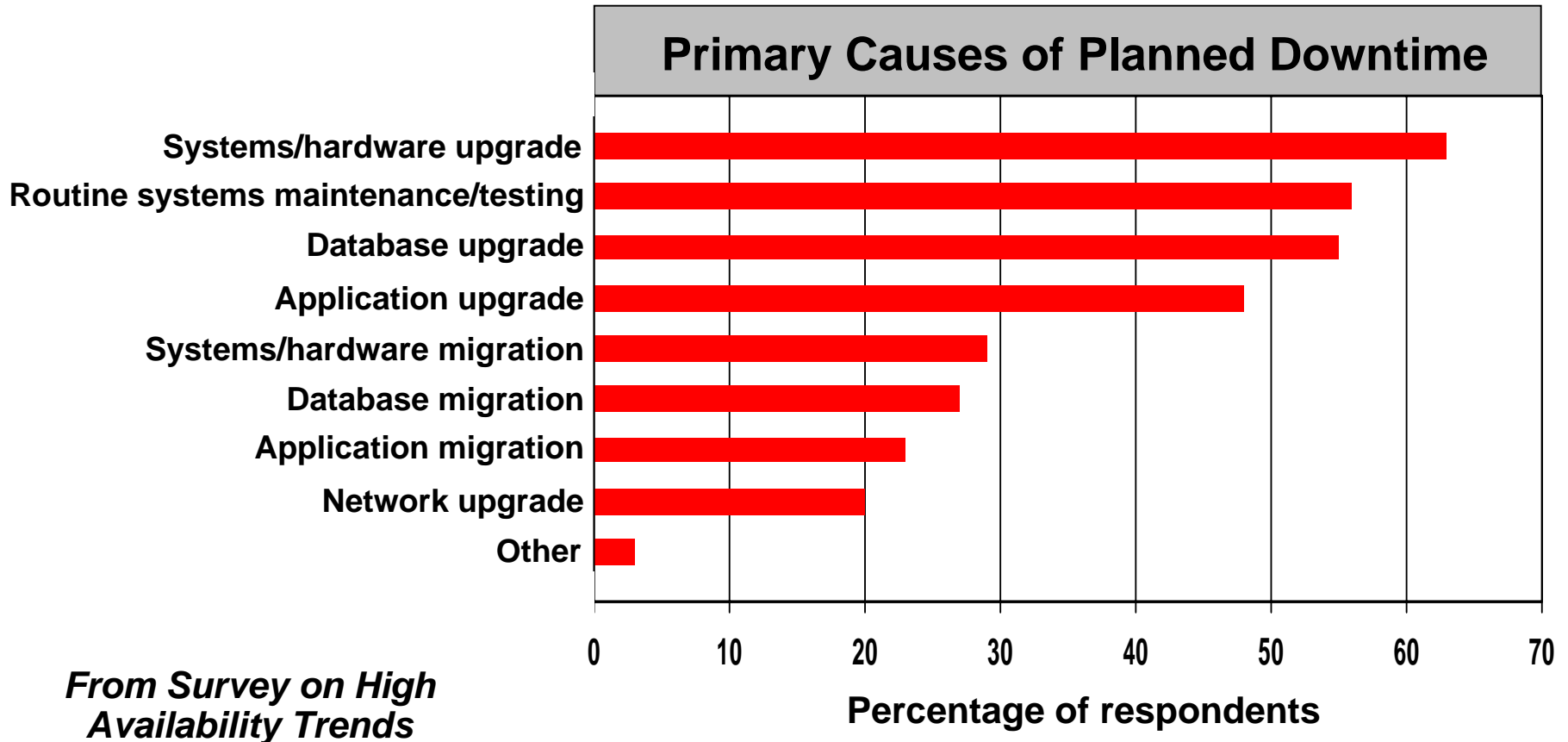
Program

- Planned Downtime: Bad News / Good News
- How Data Guard Minimizes Planned Downtime
- Tips & Tricks from Oracle Development
- Oracle In-Memory Database Cache: All the way to ZERO
- **Wrap-up & Resources**

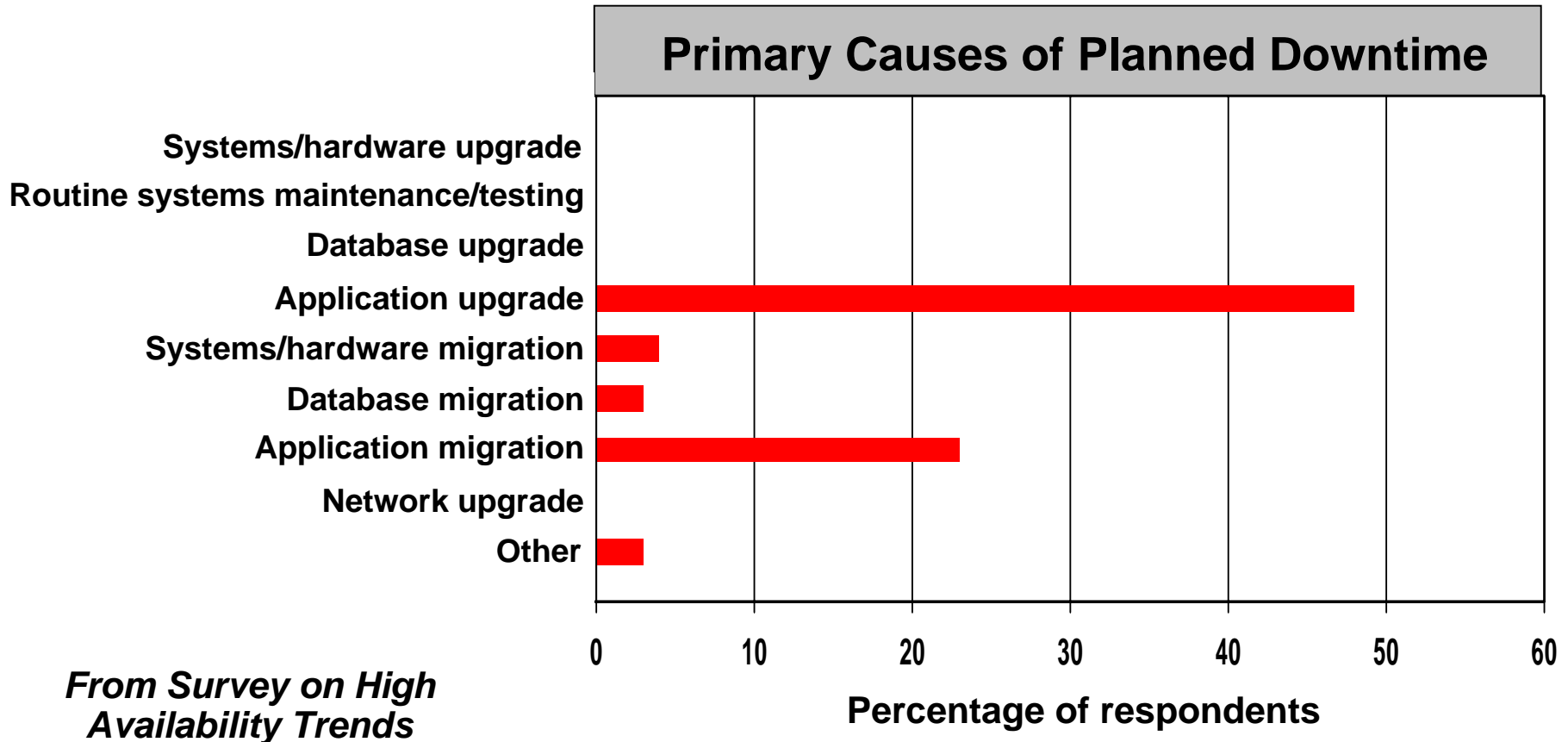


Bad News: Lots of Problems

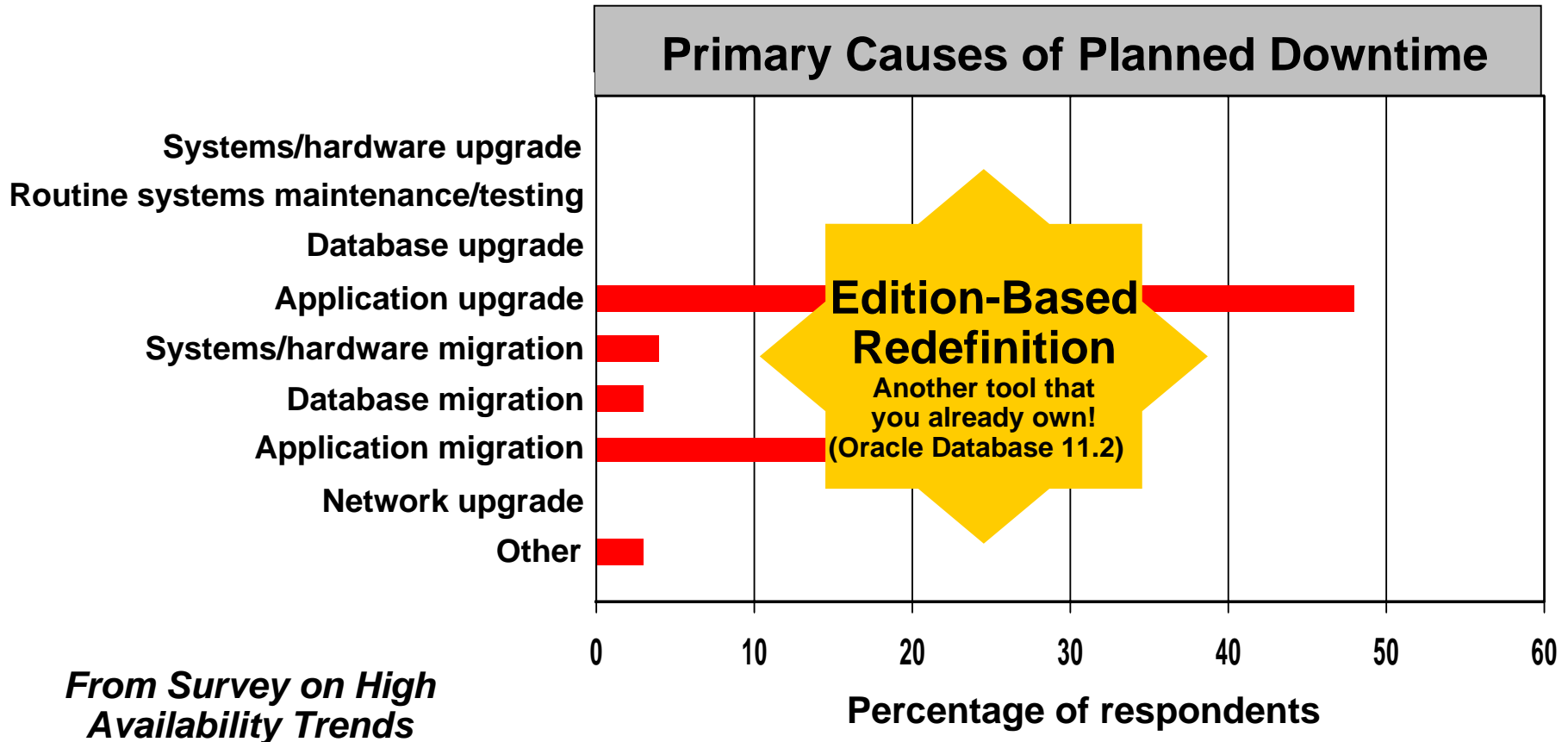
“80 Percent of all Downtime is Planned Downtime”



Good News: You Can Make Life Less Challenging Using Data Guard: A Tool You Already Own!



Good News: You Can Make Life Less Challenging Using Data Guard: A Tool You Already Own!



Key Best Practices Documentation

- HA Best Practices
http://www.oracle.com/pls/db111/portal.portal_db?selected=14&frame=
- Supported Mixed Platforms in a Data Guard Config: MetaLink Note 413484.1
- Database Rolling Upgrade Best Practices using Physical Standby
http://www.oracle.com/technology/deploy/availability/pdf/maa_wp_11g_transientlogicalrollingupgrade.pdf
- Automated Workflow for Database Rolling Upgrades using Transient Logical Standby: MetaLink Note 949322.1
- Switchover Best Practices: MetaLink Note 751600.1
- Developer and DBA Tips for Pro-Actively Optimizing SQL Apply: MetaLink Note 603361.1
- Data Guard 11g Release 2 Extended Datatype Support for SQL Apply: MetaLink Note 949516.1
- Using your Data Guard Standby for Real Application Testing
<http://www.oracle.com/technology/deploy/availability/pdf/oracle-openworld-2008/298770.pdf>
- Edition-Based Redefinition
http://www.oracle.com/technology/deploy/availability/pdf/edition_based_redefinition.pdf

How Application Clients Move to the New Primary

- After the switchover to the new version
 - Application connections remain on the new logical / original primary
 - Session have read only access to the data
 - Database Guard prevents any updates
- To transition clients to the new primary
 - Database service used by the application is started on the new primary
 - Shutdown new logical / original primary to convert to physical standby
 - Clients automatically reconnect and find service running on new primary


For More Information

search.oracle.com

or

oracle.com



The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

ORACLE®