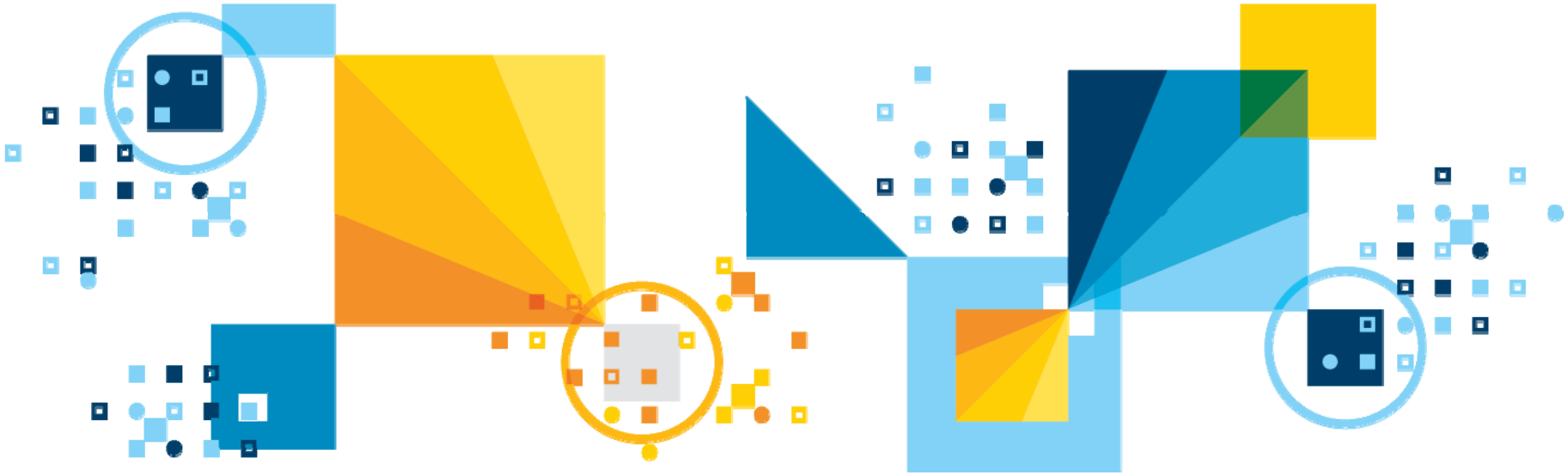
Dale McInnis

STSM / NA Data Server Tech Sales

IBM Canada Ltd.

The 5 “W”s of HADR



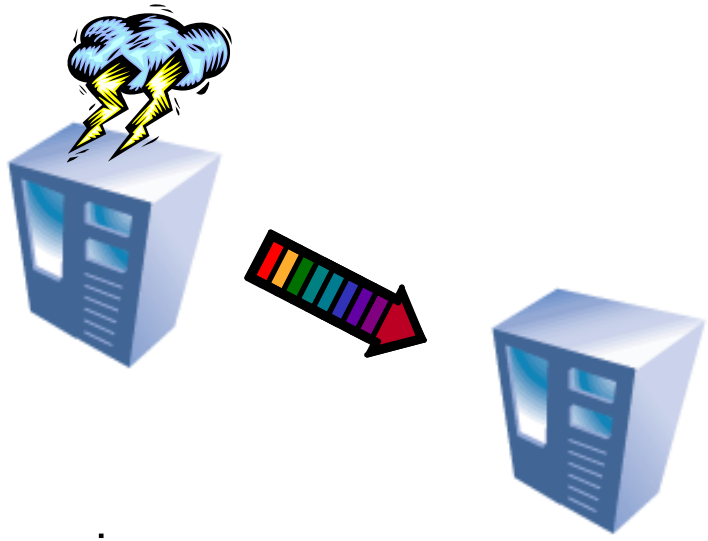
Please note

- **© IBM Corporation 2017. All rights reserved. U.S. Government Users Restricted Rights - use, duplication, or disclosure restricted by GSA ADP Schedule Contract with IBM Corporation.**
- **IBM, the IBM logo, ibm.com and DB2 are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or TM), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks make also be registred or common law trademarks in other countries. A current list of IBM trademaks is available on the Web at "Copyright and trademark information at : www.ibm.com/legal/copytrade/shtml.**
- **The information contained in this presentation is provided for informational purpose only. While efforts were made to verify the completeness and accuracy of the information contained in this presentation, it is provided "as is" without warranty of any kind, expressed or implied. IBM shall not be responsible for any damages araising out of the use of, or otherwise related to, this presentation or any other documentation.**
- **The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. Nothing contained in this presentation is intended to, nor shall have the effect of, creating any warranties or representations from IBM (or its suppliers or licensors), or altering the terms and conditions of any agreement or license governing the use of IBM products and/or software.**
- **Any statements of performance are based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multi-programming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated.**
- **IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.**

Agenda

- **WHAT is HADR**
- WHY should I use HADR
- WHERE should I deploy it
- WHEN should I use HADR
- WHO is using HADR

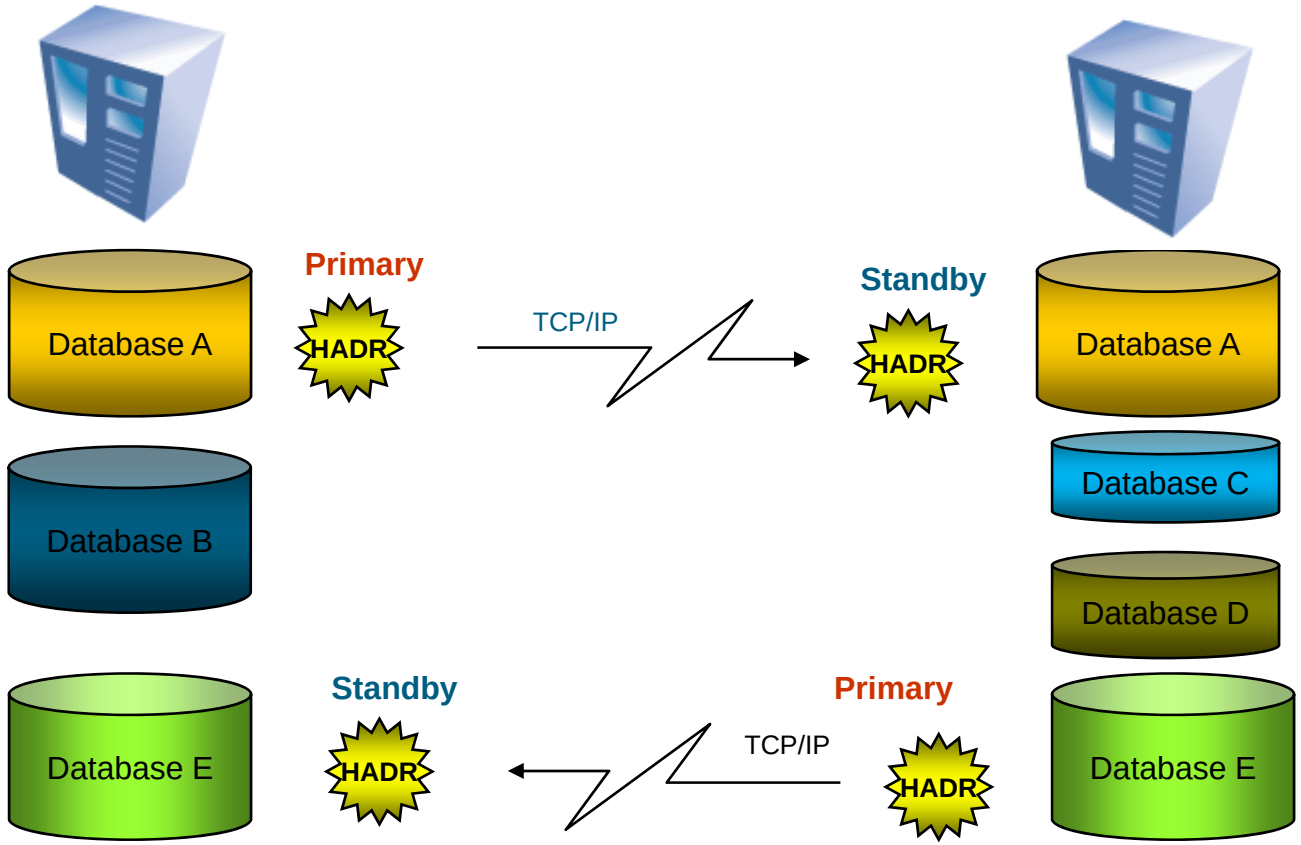
WHAT is HADR



- **Two active machines**
  - Primary
    - Processes transactions
    - Ships log entries to the other machine
  - Standby
    - Cloned from the primary
    - Receives and stores log entries from the primary
    - Re-applies the transactions
- **If the primary fails, the standby can take over the transactional workload**
  - The standby becomes the new primary
- **If the failed machine becomes available again, it can be resynchronized**
  - The old primary becomes the new standby

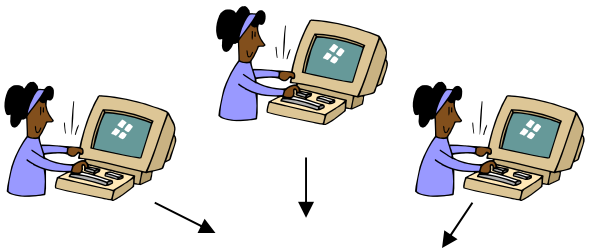
Scope of Action

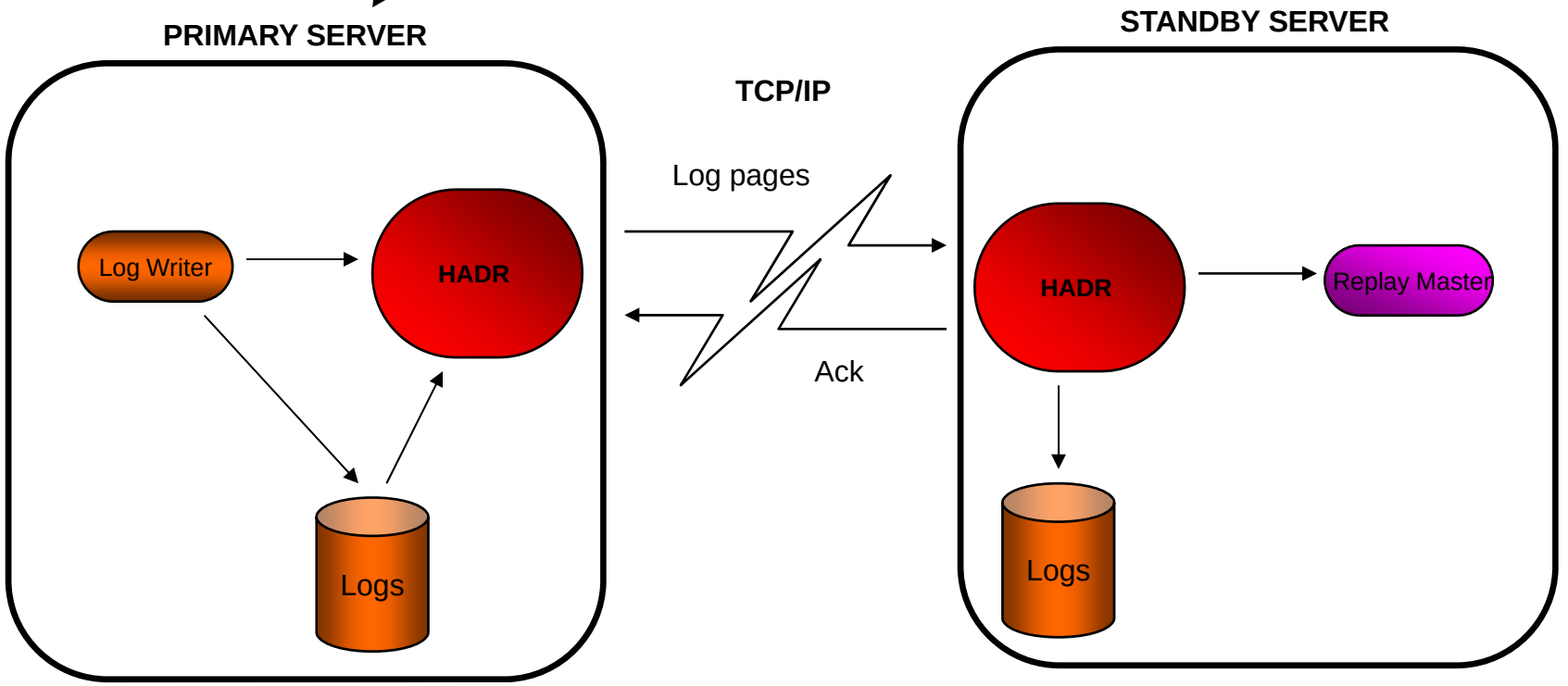
HADR replication takes place at the database level.



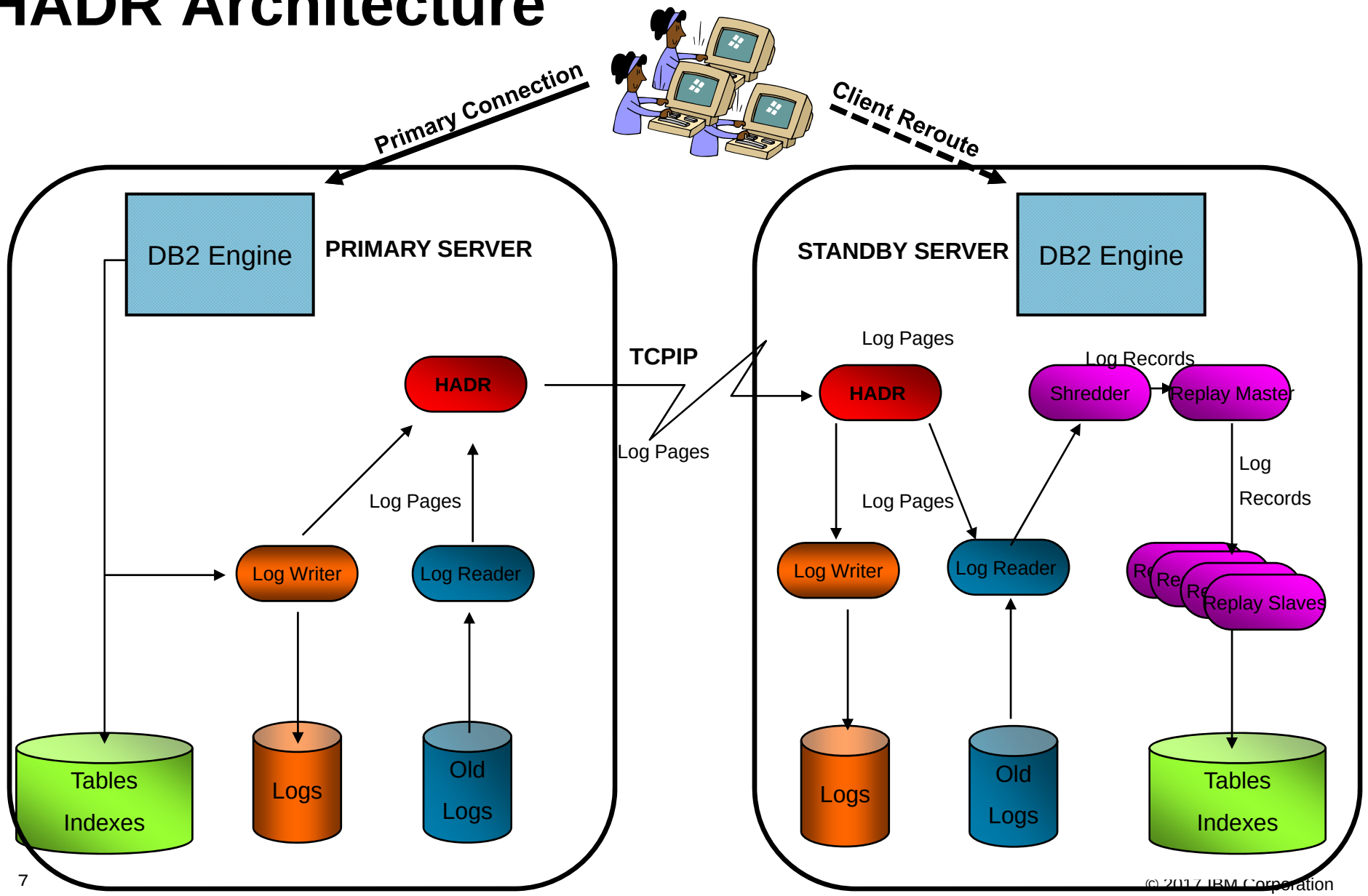
HADR replicate changes from the primary to the standby

- Transactions generate log records on the primary
- Primary sends log pages to the standby
- Standby receives log pages
- Standby writes received log pages to disk and sends Acks
- Standby replays written log pages





HADR Architecture

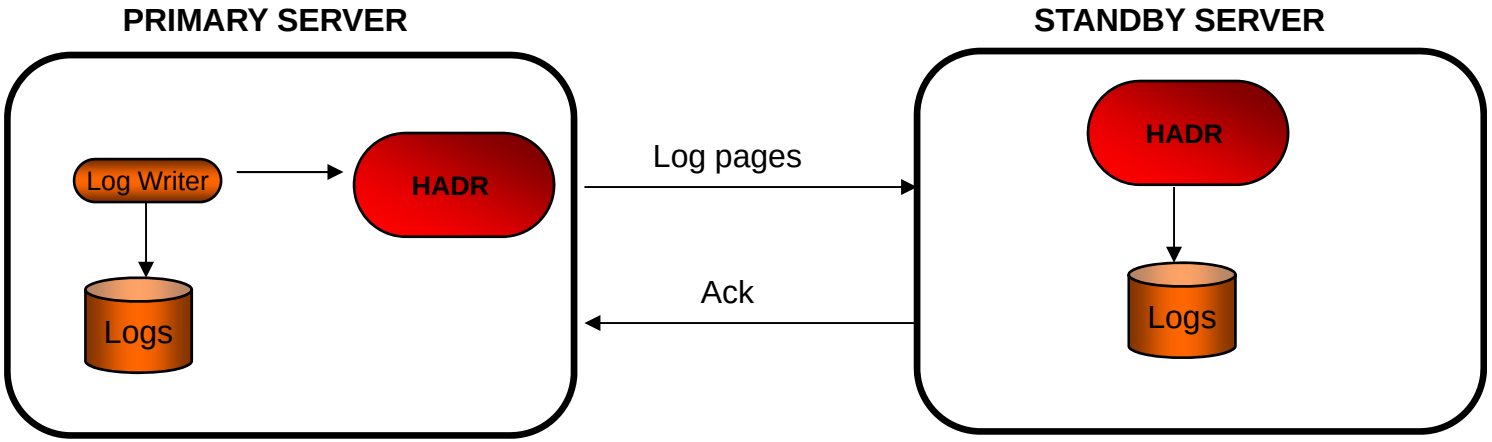


What's replicated, what's not?

- **Logged operations are replicated**
  - Example: DDL, DML, table space and buffer pool creation/deletion.
- **Not logged operations are not replicated.**
  - Example: database configuration parameter. not logged initially table, UDF libraries.
- **Index pages are not replicated unless LOGINDEXBUILD is enabled**
  - Ensure logsecond is maxed out as index rebuild is a single transaction
- **How do I prevent non-logged operations?**
  - Enable BLOCKNONLOGGED db cfg parameter

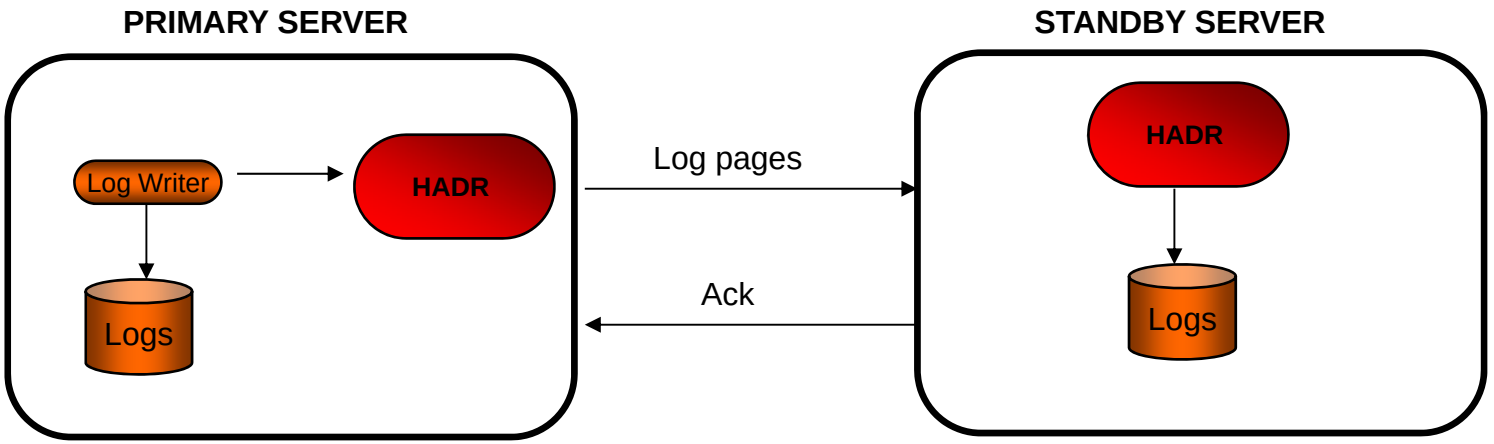
Synchronization Modes

- **SYNC mode:**
  - Logs are first written to the primary and are only then sent to standby (Serially)
  - Two on-disk copies of the log data are required for transaction commit
  - Total log write time = primary log write + log send + standby log write + ack message
  - Best data protection
  - But the cost of replication is higher than all other modes



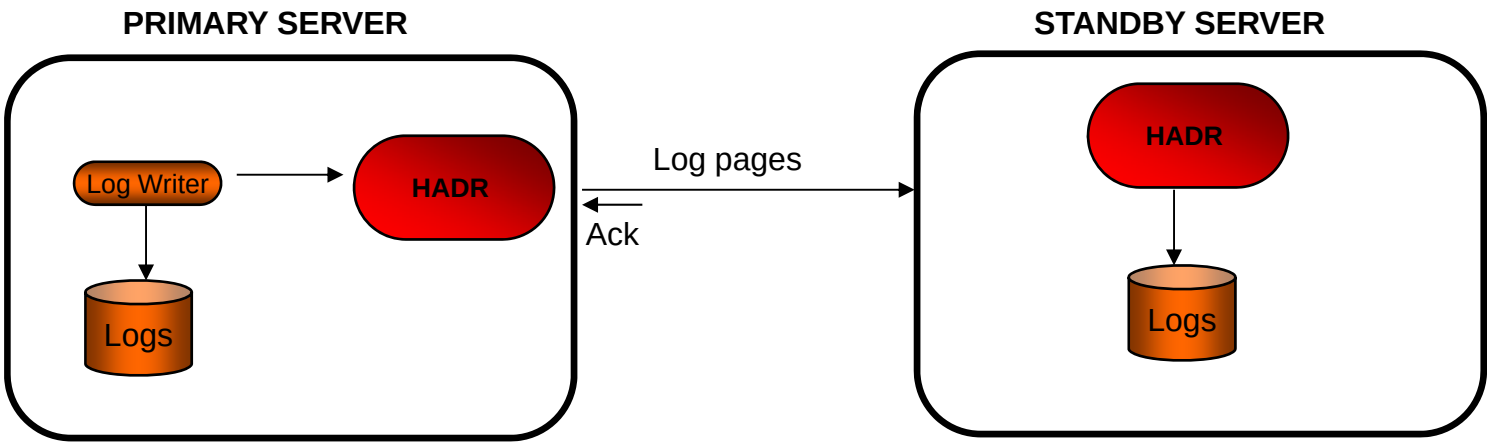
Synchronization Modes

- **NEARSYNC mode:**
  - Writing logs on the primary and sending logs to standby are done in parallel
  - Standby sends ack message as soon as it receives the logs
  - On a fast network, log replication results in little or no overhead to primary
  - Total log write time = Max (primary log write, log send + ack message)
  - Exposure to the relatively rare ‘double failure’ scenario
    - primary fails and the standby fails before it has a chance to write received logs to disk
  - Good choice for many applications
  - providing near synch protection at lower performance cost



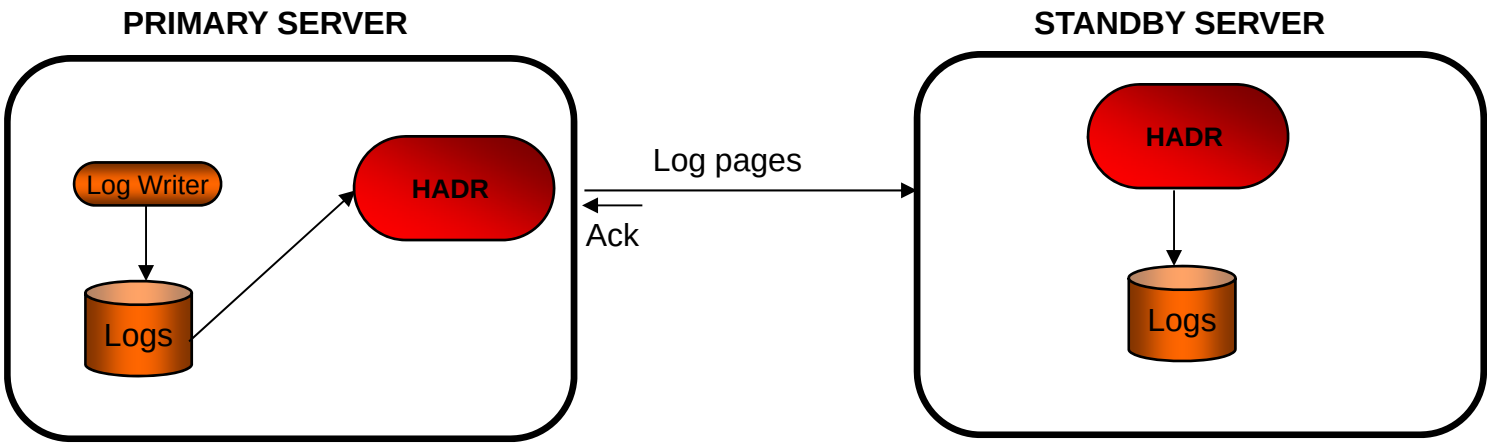
Synchronization Modes

- **ASYNC mode**
  - Writing logs on the primary and sending logs to standby are done in parallel
  - Does not wait for ack messages from the standby
    - Just the ack that the message has been sent
  - If the primary database fails, there is a higher chance that logs in transit are lost
  - Total log write time = Max (Primary log write rate, Submit log for sending)
  - Well suited for WAN application since network transmission delay does not impact performance



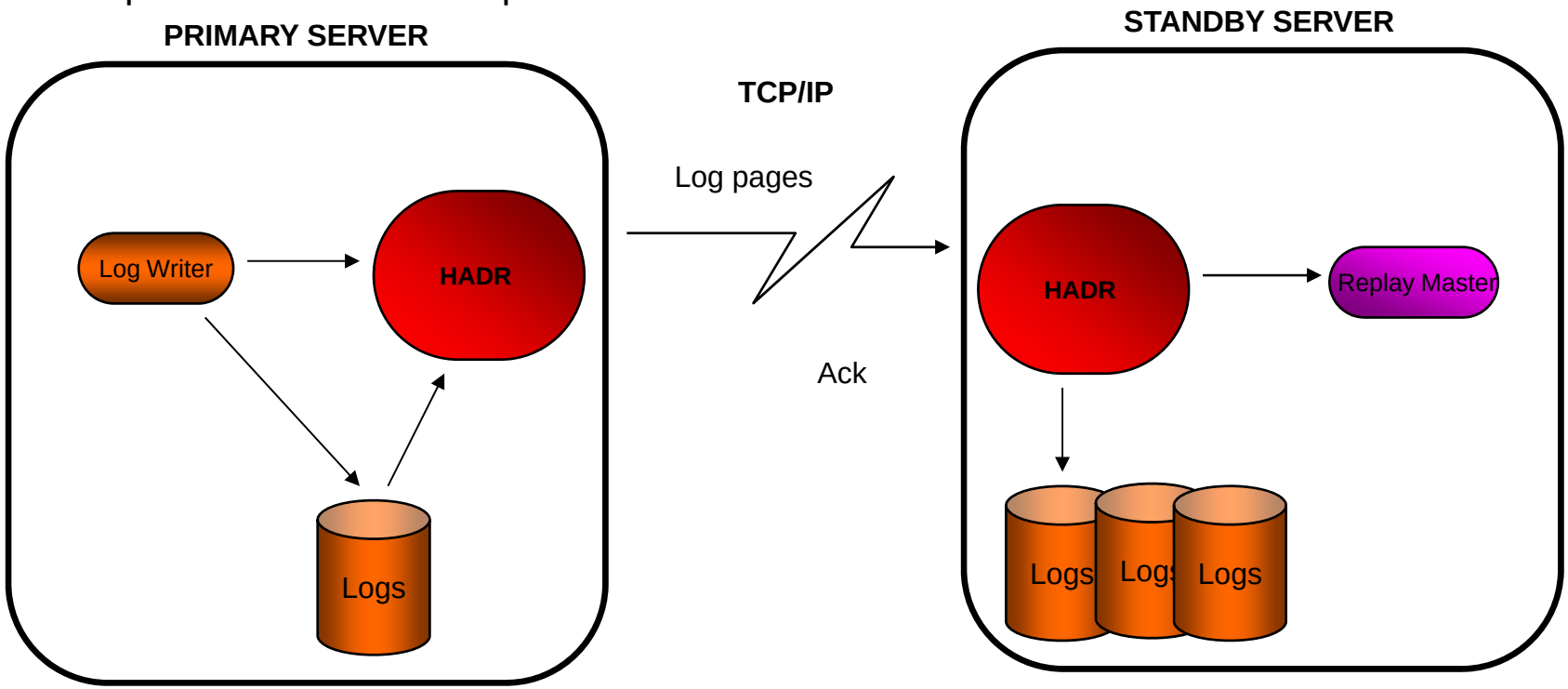
Synchronization Modes

- **SUPERASYNC mode**
  - Log writing and log shipping are completely independent
  - HADR remains in remote catchup state and never enters peer state
  - Zero impact on Log writing: Total log write time = Primary log write
  - But the log gap between the primary and the standby can grow
    - In a failover, data in the gap will be lost.
  - This mode has the least impact on primary, at the cost of the lowest data protection



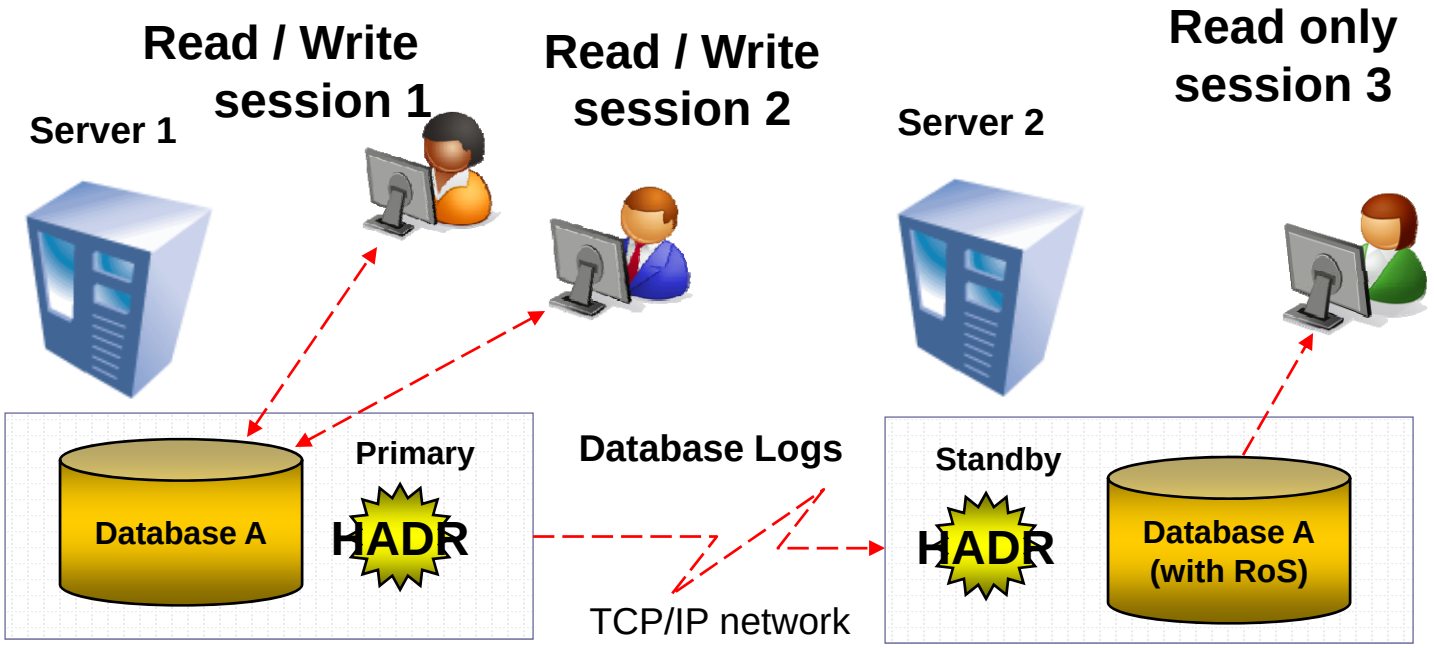
HADR Log Spooling

- **All log records are written to the active log path on the standby**
- **Synchronization Mode is with respect to the log data shipped not with respect to the replay**
- **Size of spool controlled by the HADR_SPOOL_LIMIT configuration parameter**
  - In v10.1 defaults to 0 (i.e. spooling disabled)
  - In v10.5 defaults to AUTOMATIC which is (LOGPRIMARY + LOGSECOND) log files, computed at HADR startup



HADR Read On Standby (RoS)

- Reads on Standby provides high availability, disaster recovery and allows reporting workloads.
- Improve resource utilization on your HA or DR hardware
- Offload reporting work from your primary, Increase capacity of HADR system
- Maximize Return on Investment and decrease Total Cost of Ownership
- Concurrent replay and query workload on the standby
- V 9.7 FP5+ supports returning inline LOBs / XML



Tools available to measure HADR related performance

- **Check out https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/DB2HADR/page/Perf%20Tuning**
- **HADR Simulator to measure network and disk speed**
- **DB2 Log Scanner to analyze database workload**
- **HADR Calculator to estimate performance of various HADR sync modes**

HADR Automation using TSA/MP

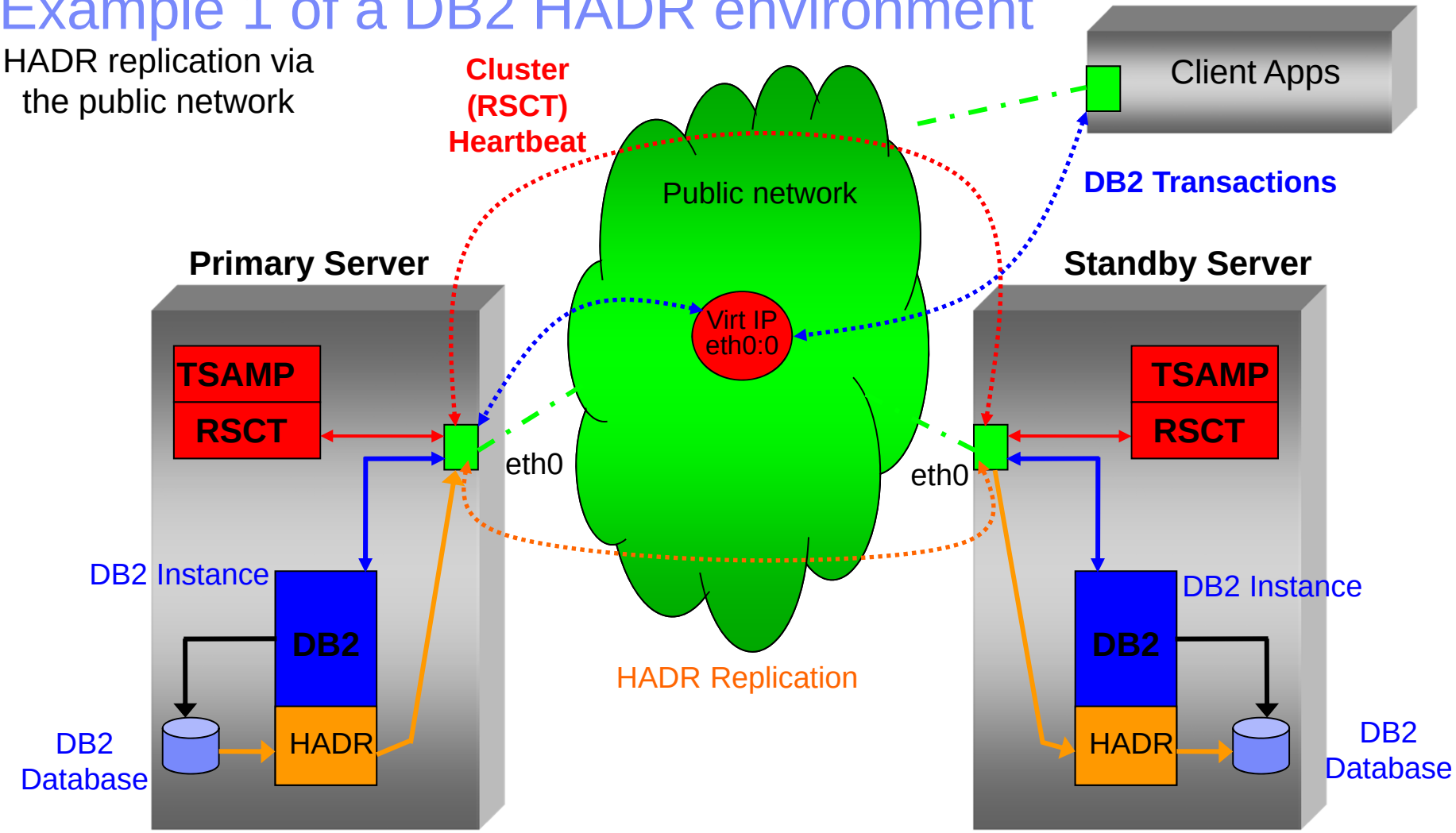
- **Follow instructions in**

http://download.boulder.ibm.com/ibmdl/pub/software/dw/data/dm-0908hadrdb2haicu/HADR_db2haicu.pdf

- **Items to watch out for (source: Ember Crooks)**
  - Ensure that you have picked either the server's short name or the server's long name and are using it consistently in each of:
    - /etc/hosts
    - HADR configuration parameters in db cfg
    - db2nodes.cfg (in $HOME/sqllib)
    - Results of the 'hostname' command
    - Ensure the hostname case is the same in all location
  - Ensure that you successfully executed the preprpnode command on both hosts
  - Ensure that you successfully executed the db2cptsa command on both hosts

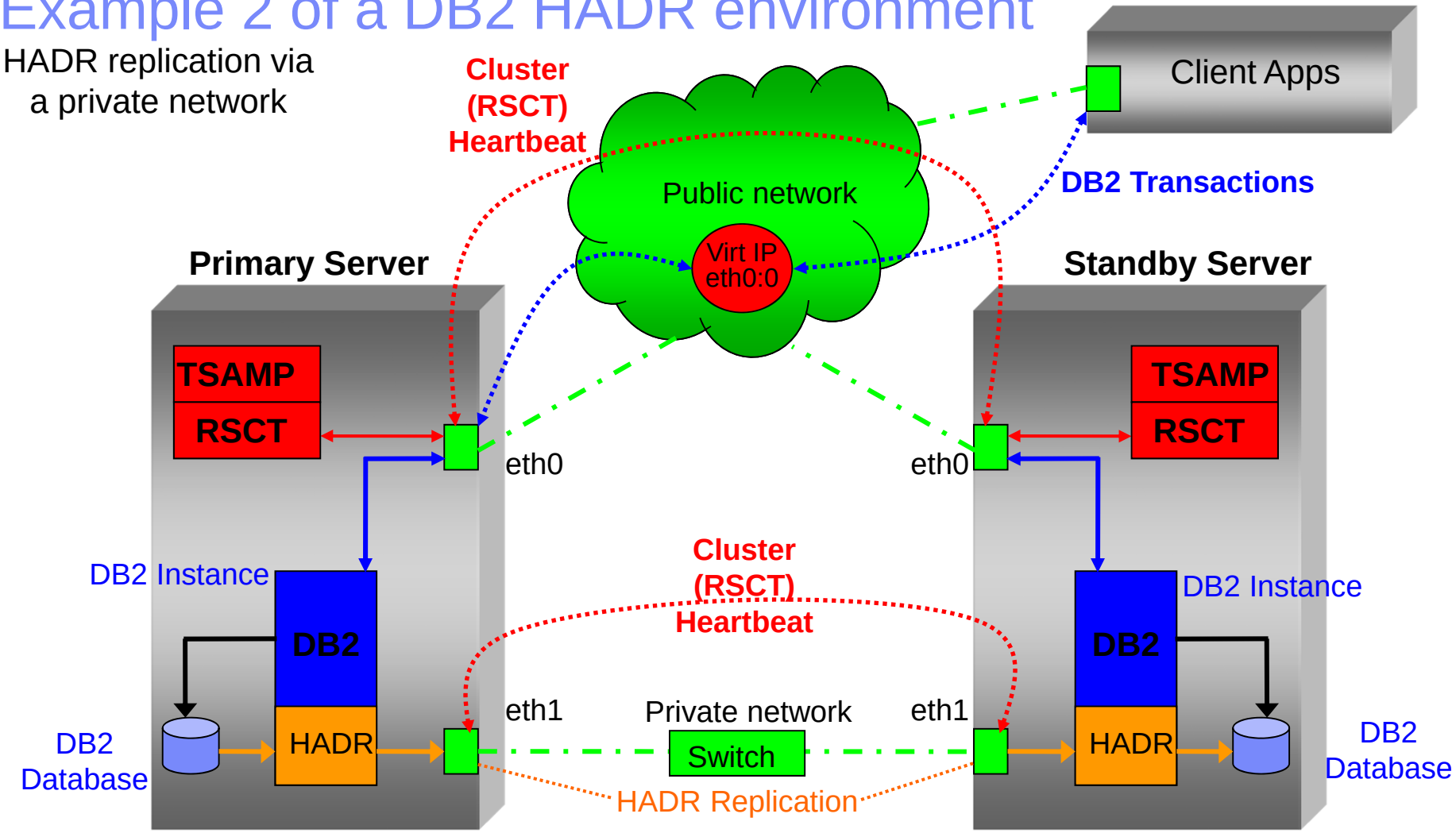
Example 1 of a DB2 HADR environment

HADR replication via the public network



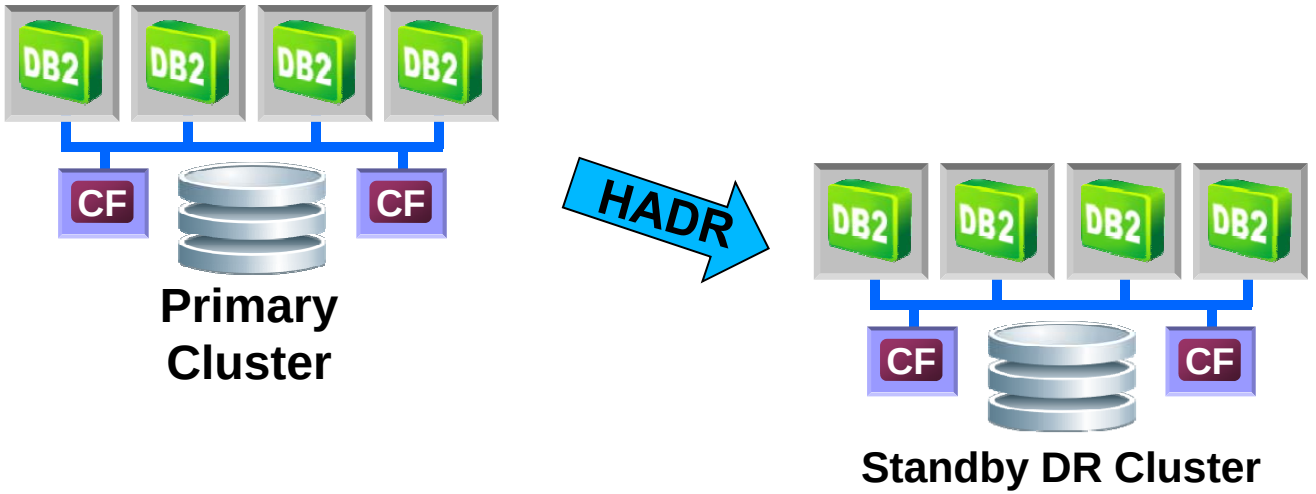
Example 2 of a DB2 HADR environment

HADR replication via
a private network

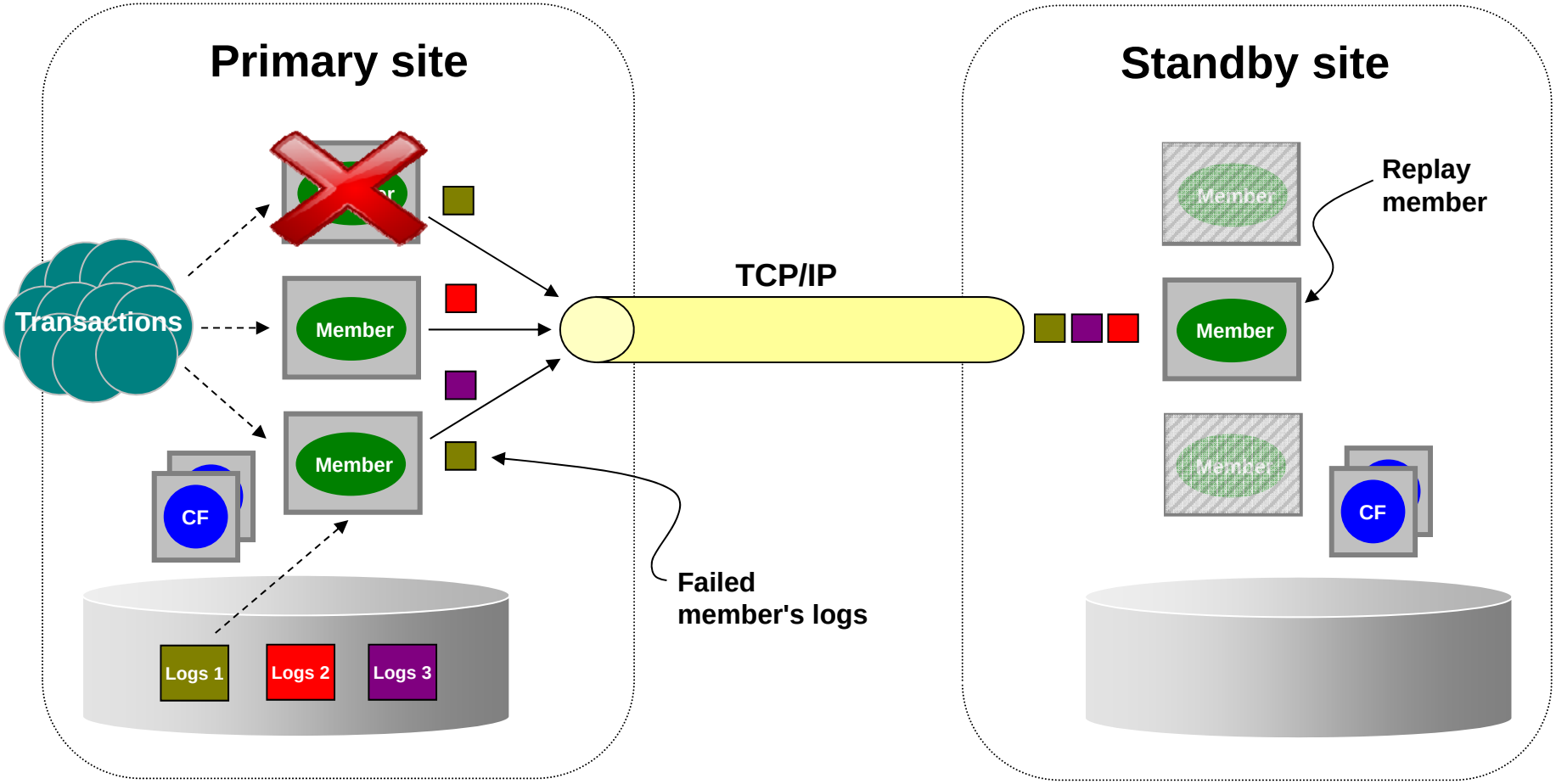


HADR in DB2 pureScale

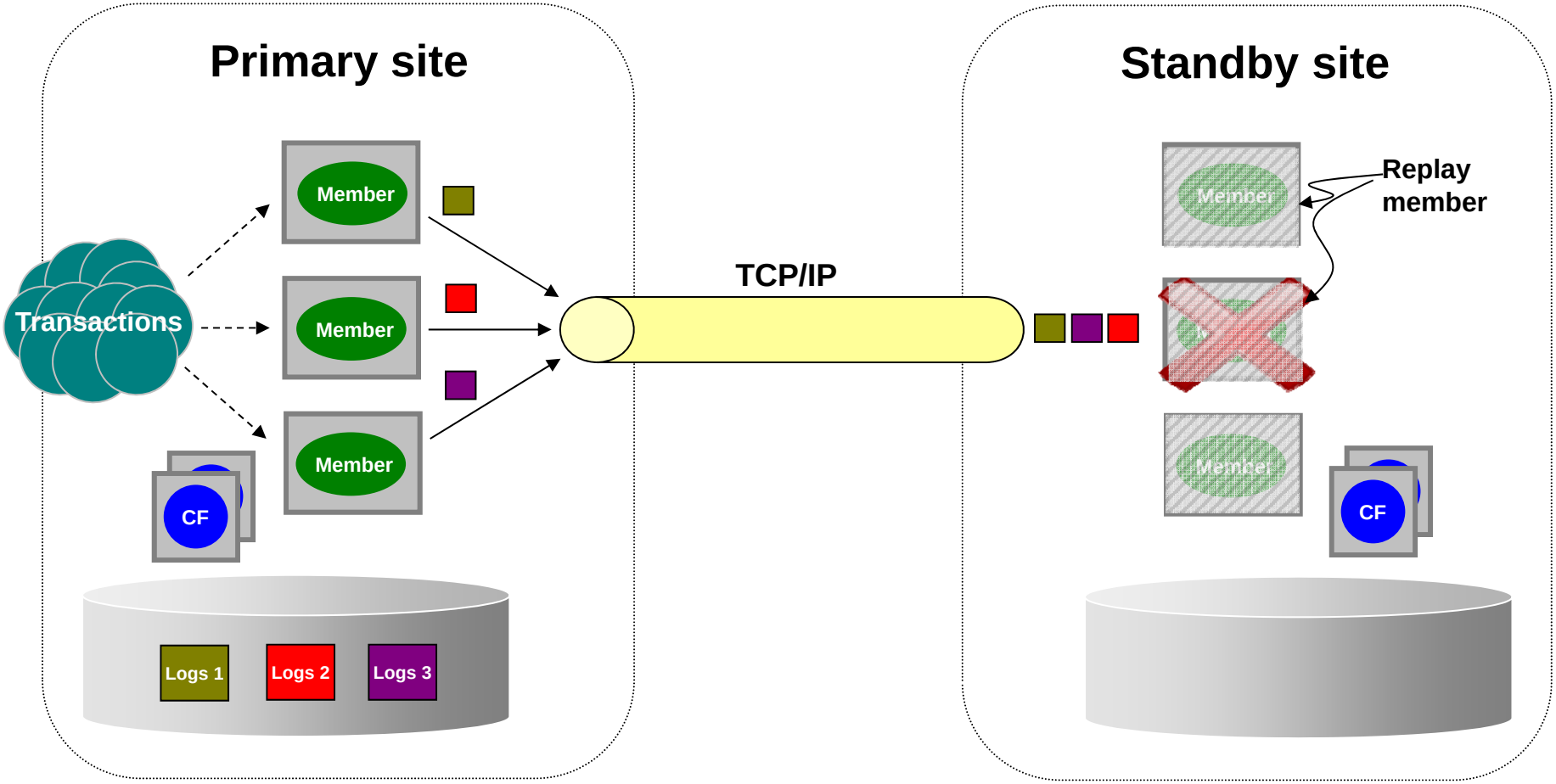
- **Integrated disaster recovery solution**
  - Simple to setup, configure, and manage
- **Support includes**
  - SYNC, NEARSYNC, ASYNC and SUPERASYNC modes
  - Time delayed apply
  - Log spooling
  - Both non-forced (role switch) and forced (failover) takeovers



HADR in DB2 pureScale: Example



HADR in DB2 pureScale: Example

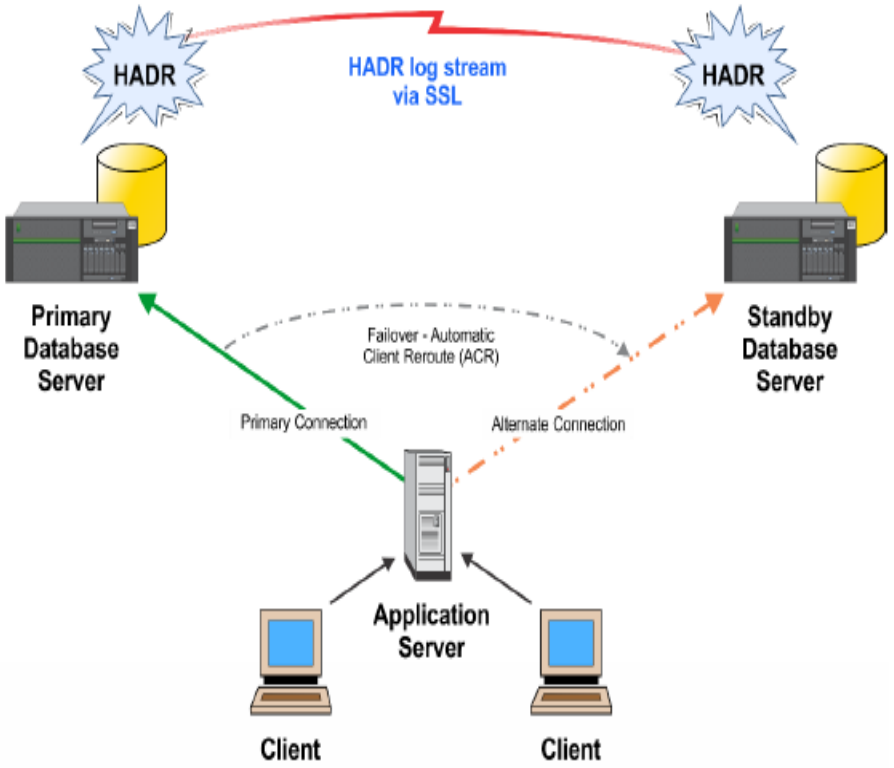


Db2 V11 recent HADR enhancements

- Security
- PureScale sync modes
- Improved version upgrade
- Faster takeover
- Standby tablespace recovery
- Monitoring standby enhancements

SSL Encryption Between HADR Primary and Standbys

- Provides integrated protection of sensitive data in the log stream
- Enabled via the **HADR_SSL_LABEL** database configuration parameter
- Supports all HADR synchronization modes
- Supports multiple standbys
- Currently supported with Linux on x86 (non-pureScale) platforms only




```
db update database configuration for <database_name> using HADR_SSL_LABEL <SSL_certificate_label_name>
```

All Sync modes supported in pureScale HADR

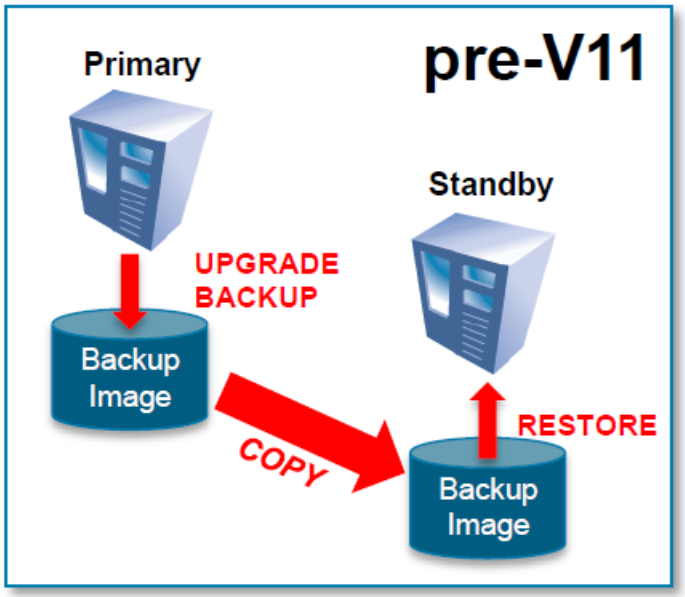
- **Combines pureScale and HADR to provide a near continuously available system with robust RPO=0 DR**
- **Other capabilities & enhancements include**
  - Seamless rolling update supported; **No TAKEOVER, no FORCEd connections, 0 outage**
    1. **On STANDBY CLUSTER** : Perform pureScale rolling update
    2. **On PRIMARY CLUSTER** : Perform pureScale rolling update
  - In V11.1, HADR log send and replay can occur **during crash recovery**
    - Allows logs written during crash recovery to replayed while crash recovery is occurring
      - Previously, log send and replay was disabled during crash recovery
    - Especially important in pureScale during **online** member crash recovery
    - Support added for both pureScale and non-pureScale

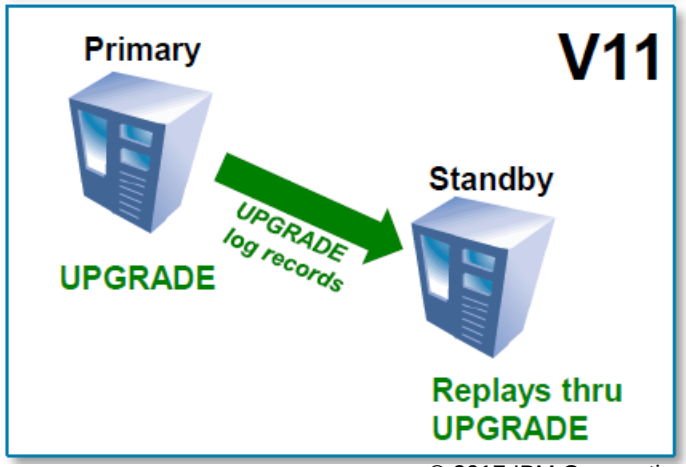
Streamlined HADR Version Upgrade

- No more need to re-initialize the standby database during version upgrades
  - Saves significant time & effort especially with large and numerous databases

- Procedure overview :

1. **PRIMARY :**
   DEACTIVATE database; DB2STOP instance
   Upgrade the instance using db2iupgrade
   - db2iupgrade invokes db2ckupgrade which will check to ensure primary and standby log positions match
   - db2ckupgrade will not issue STOP HADR or change role
2. **STANDBY :**
   DEACTIVATE database; DB2STOP instance
   Upgrade the instance using db2iupgrade
   - db2ckupgrade will no longer fail 'because it is an HADR standby'
3. **STANDBY : UPGRADE the database**
   - Returns successful – this indicates that the standby is now waiting for log data from a subsequent UPGRADE issued on the primary
   - No connections can be made to a standby database in this state (including with Reads on Standby)
   - Can be monitored via db2pd –hadr
4. **PRIMARY : UPGRADE the database**
   - Will ship log data to standby; Standby replays these log records
5. **PRIMARY : ACTIVATE the database**
   - Now primary and standby can resume normal operations





Base HADR V11 Enhancements

Improved HADR Takeover performance

- **Prior to 11.1.2.2, during Crash Recovery (or HADR Takeover by Force), connections are not allowed to the database until all recovery is complete**
  - A REDO phase to redo all transaction activity to the end of log
  - An UNDO phase to rollback transactions that had not committed

- **In Db2 11.1.2.2, we allow connections and activity into the database after the REDO phase and while the UNDO phase is executing**
  - This was a technical preview in Db2 11.1.1.1

HADR Tablespace Recovery

- If the standby database has a tablespace in an invalid or error state, a TAKEOVER by the Standby will not be fully available to applications
- Prior to Db2 11.1.2.2, the entire Standby database needed to be recovered
- In 11.1.2.2, the tablespace can be recovered online without a full database refresh
- Tablespace recovery been retrofitted into Db2 10.5fp9+
- Details: http://www-01.ibm.com/support/docview.wss?uid=swg21993389

Monitoring HADR Tablespace Status

- **Tablespace Error State**
  - When a tablespaces is in an invalid or error state on the Standby database, the **HADR_FLAGS** field will display the value **STANDBY_TABLESPACE_ERROR**
- **Monitoring with db2pd**
  - The **HADR_FLAGS** field can be monitored by using the **db2pd -hadr** command on the Primary or Standby database

```
db2pd –db HADRDB1 –hadr
… HADR_FLAGS = STANDBY_TABLESPACE_ERROR TCP_PROTOCOL …
```

- **Monitoring with Table Function**
  - The **MON_GET_HADR()** table function will display the current status on either the Primary database or Standby database with Reads on Standby enabled

```
SELECT STANDBY_ID, HADR_FLAGS FROM
       TABLE(MON_GET_HADR(NULL))

STANDBY_ID
---------- ----------------------------------
         1 STANDBY_TABLESPACE_ERROR TCP_PROTOCOL
```

Agenda

- WHAT is HADR
- **WHY should I use HADR**
- WHERE should I deploy it
- WHEN should I use HADR
- WHO is using HADR

Why should I use HADR

Several main reasons

- **SIMPLICITY:**
  - ultra easy to deploy
  - Single command to move processing from one node to another
- **PEACE OF MIND:**
  - with automation enabled a failure anywhere in the stack will be result in only a 30 second interruption in service
- **SINGLE TECHNOLOGY FOR HA AND DR**
  - With multiple standbys, HADR can be used for local high availability as well as disaster recovery

HADR Setup Fits on One Slide

Primary Setup

db2 backup db hadr_db to backup_dir

db2 update db cfg for hadr_db using

HADR_LOCAL_HOST	host_a
HADR_LOCAL_SVC	svc_a
HADR_TARGET_LIST	host_b:svc_b
HADR_REMOTE_INST	inst_b
HADR_TIMEOUT	120
HADR_SYNCMODE	ASYNC

db2 start hadr on database hadr_db as primary

Standby Setup

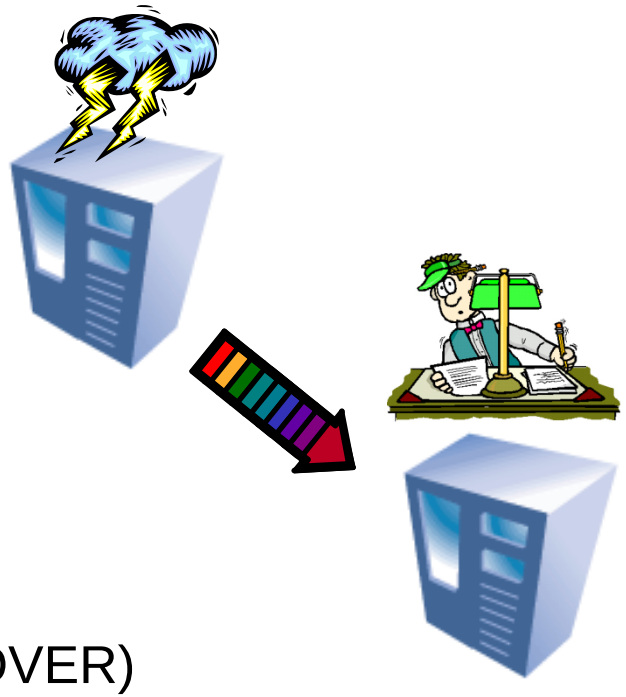
db2 restore db hadr_db from backup_dir

db2 update db cfg for hadr_db using

HADR_LOCAL_HOST	host_b
HADR_LOCAL_SVC	svc_b
HADR_TARGET_LIST	host_a:svc_a
HADR_REMOTE_INST	inst_a
HADR_TIMEOUT	120
HADR_SYNCMODE	ASYNC

db2 start hadr on database hadr_db as standby

Failing Over : Simple “TAKEOVER” Command



- Normal TAKEOVER
  - Primary and standby switch roles as follows:
    1. Standby tells primary that it is taking over.
    2. Primary forces off all client connections and refuses new connections.
    3. Primary rolls back any open transactions and ships remaining log, up to the end of log, to standby.
    4. Standby replays received log, up to end of the log.
    5. Primary becomes new standby.
    6. Standby becomes new primary
- Emergency TAKEOVER (aka ‘Forced’ TAKEOVER)
  - The standby sends a notice asking the primary to shut itself down.
  - The standby does NOT wait for any acknowledgement from the primary to confirm that it has received the takeover notification or that it has shut down
  - The standby stops receiving logs from the primary, finishes replaying the logs it has already received, and then becomes a primary.

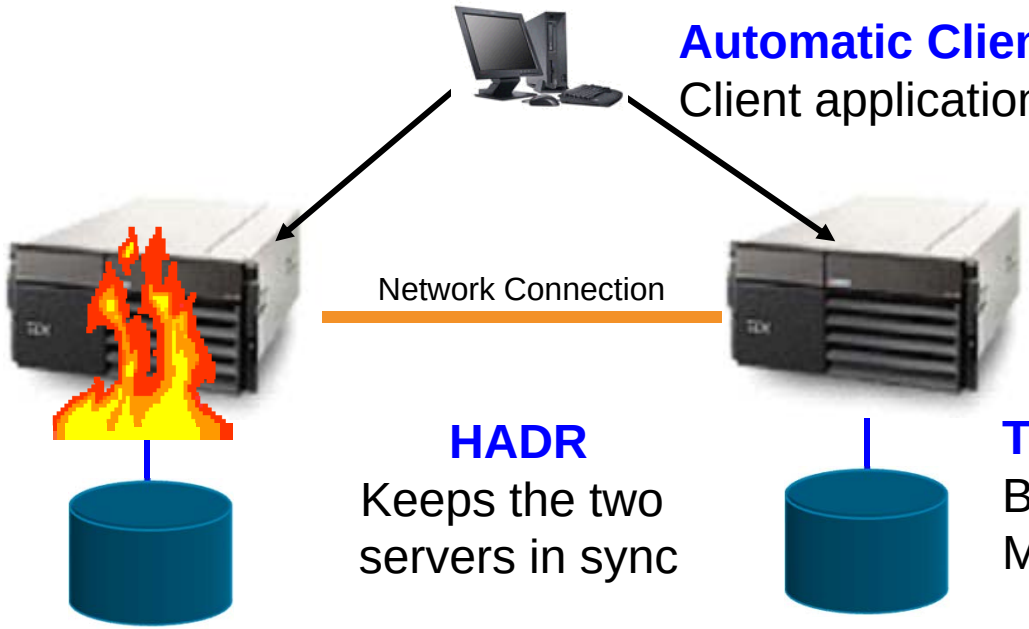
```
TAKEOVER HADR ON DATABASE <dbname>
          <USER <username> [USING <password>]]      [BY FORCE]
```

Primary Reintegration

- After primary failure and takeover, allow old primary to reintegrate as a standby with the new primary (saves user from having to reinitialize standby from scratch)
- Differentiating feature for DB2 HADR – competitors do not support this
- Reintegration possible if old primary can be made consistent with new primary
- Some conditions to satisfy, e.g. old primary crashed in peer state and had no disk updates that were not logged on old standby; some other details.
- Successful reintegration is most likely in SYNC mode, least likely in ASYNC and SUPERASYNC modes
- Synchronization with tail of the log file

High Availability Disaster Recovery (HADR)

- **Delivers fast failover at low cost**
- **Redundant copy of the database to protect against site or storage failure**
- **Support for Rolling Upgrades (Fix Packs *and* Versions!)**
- **Failover typically under 15 seconds**
- **100% performance after primary failure**
- **Easy to setup, configure, monitor, and manage**



Automatic Client Reroute
Client application transparently resumes on Standby

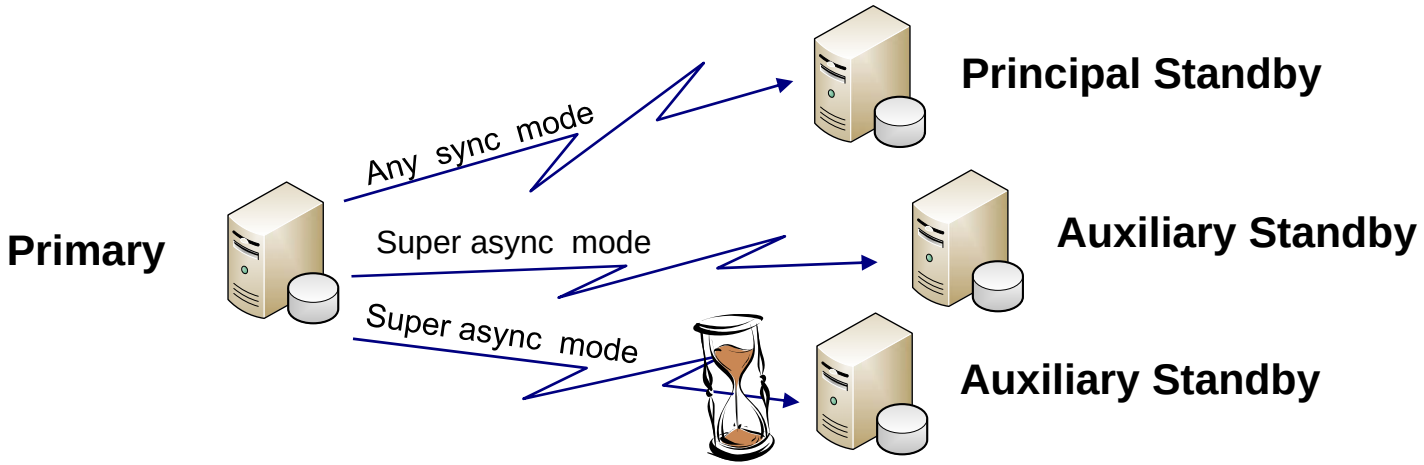
HADR
Keeps the two servers in sync

TSA for server monitoring
Built in cluster manager
Monitors primary, performs takeover

Primary Server

Standby Server

HADR Multiple Standbys, Time Delay



- **1 standby for HA, and up to 2 other standbys for DR**
  - Rolling fix pack updates of standbys and primary without losing HA
- **Reads on Standby (RoS) supported on all standbys**
- **TSA Automation for takeover is only to Principal Standby**
- **Takeover (forced and non-forced) supported from any standby**
  - After takeover, configuration parameters on new primary's standbys will be changed automatically so they point to the new primary
- **Time delay (db-cfg parameter) helps recover from application errors**
  - Must be noticed before time delay on standby results in changes being replayed

Agenda

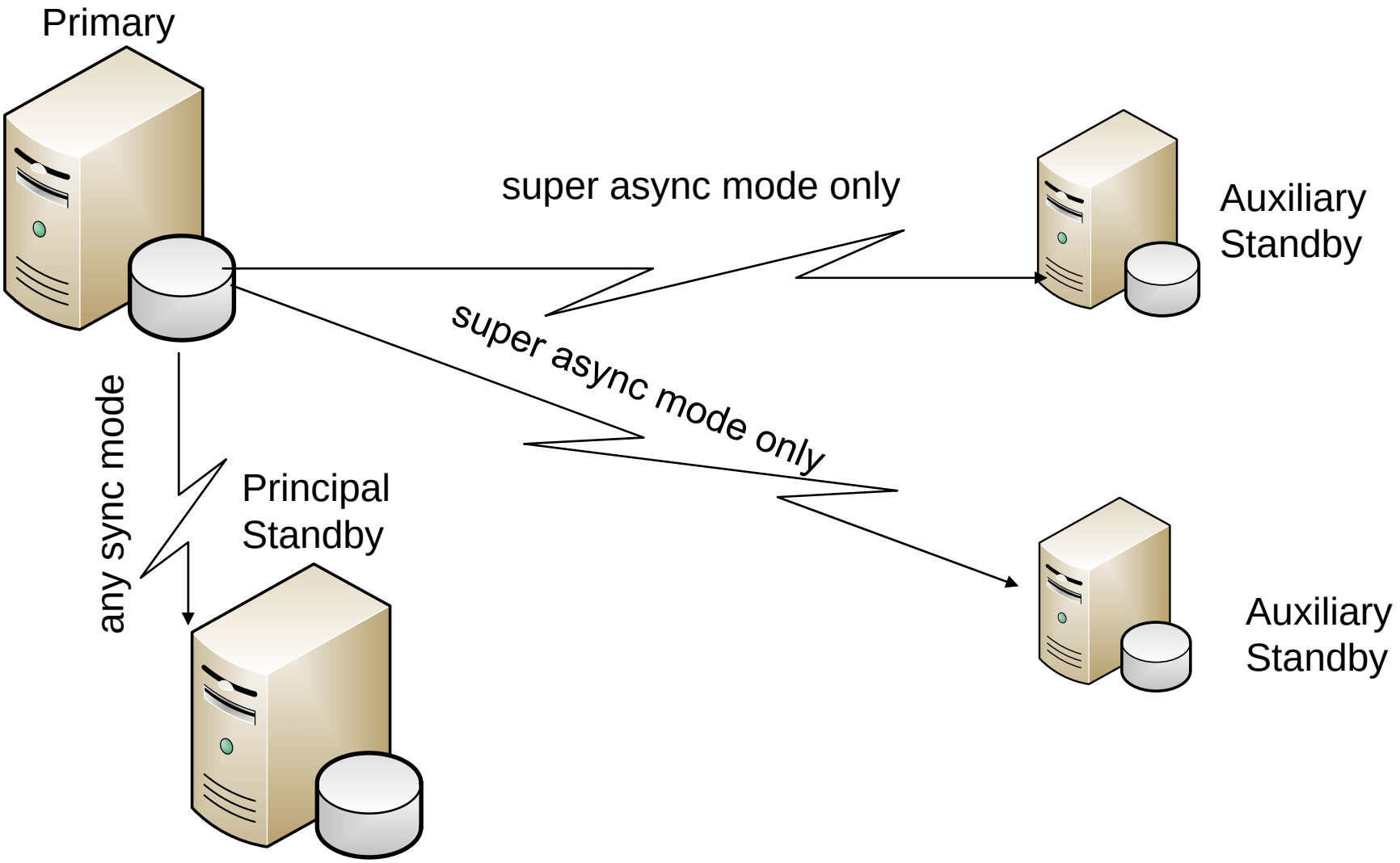
- WHAT is HADR
- WHY should I use HADR
- **WHERE should I deploy it**
- WHEN should I use HADR
- WHO is using HADR

Where should I deploy HADR?

- Db2 HADR services 2 main purposes
  - Very fast high availability, typically to a co-located system and typically automated via TSA- MP
  - Disaster recovery, typically to a geographical separated system

- Why not use the same technology for BOTH!

HADR Multiple Standby Overview

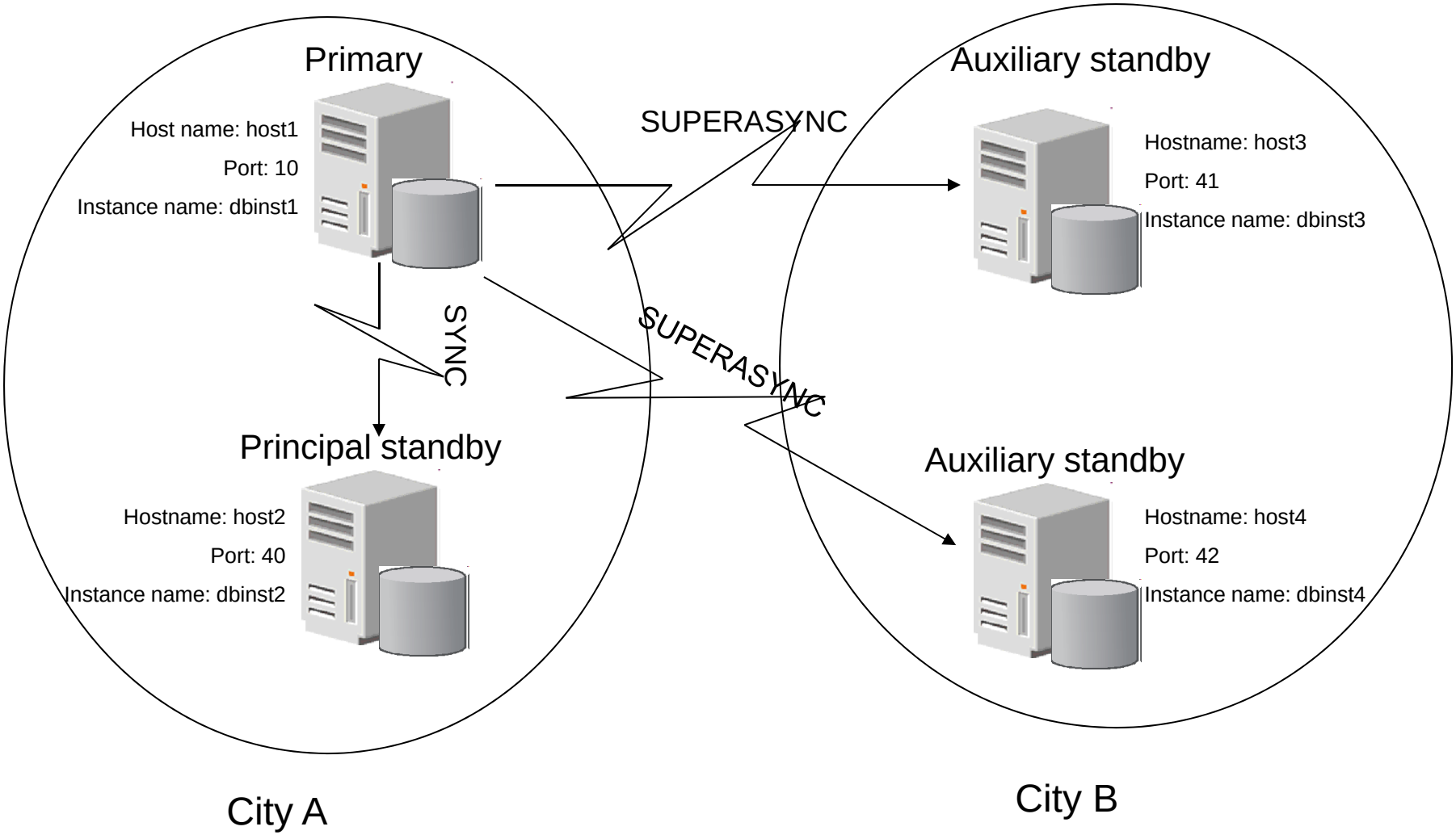


HADR Multiple Standby Features

- **Principal Standby (PS) equivalent to standby today**
  - PS supports any sync mode
  - Can automate takeover using integrated TSA

- **Support for up to two(2) Auxiliary Standbys (AS)**
  - AS supports super async mode only
  - No automated takeover supported
  - Always feed from the current primary
  - Can be added dynamically

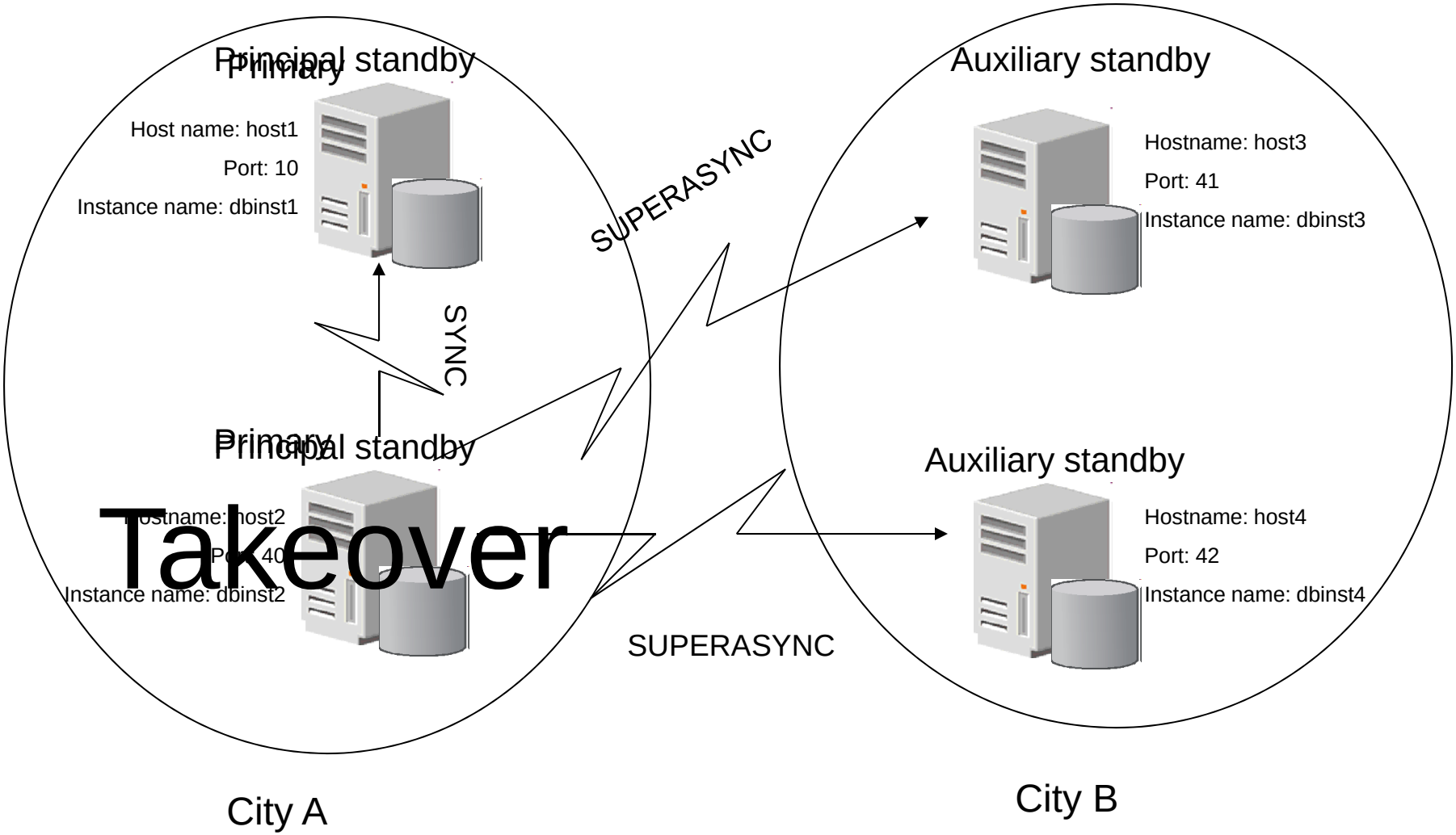
HADR Multiple Standby Example



Configuration values for each host

Configuration parameter	Host1	Host2	Host3	Host4
Hadr_target_list	host2:40 | host3:41 | host4:42	host1:10 | host3:41 | host4:42	Host2:40 | host1:10 | host442	host3:41 | host1:10 | host2:40
Hadr_remote_host	host2	host1	host1	host1
Hadr_remote_svc	40	10	10	10
Hadr_remote_inst	dbinst2	dbinst1	dbinst1	dbinst1
Hadr_local_host	host1	host2	host3	host4
Hadr_local_svc	10	40	41	42
Operational Hadr_syncmode	sync	sync	Near sync	Async
Effective Hadr_syncmode	N/A	sync	superasync	superasync

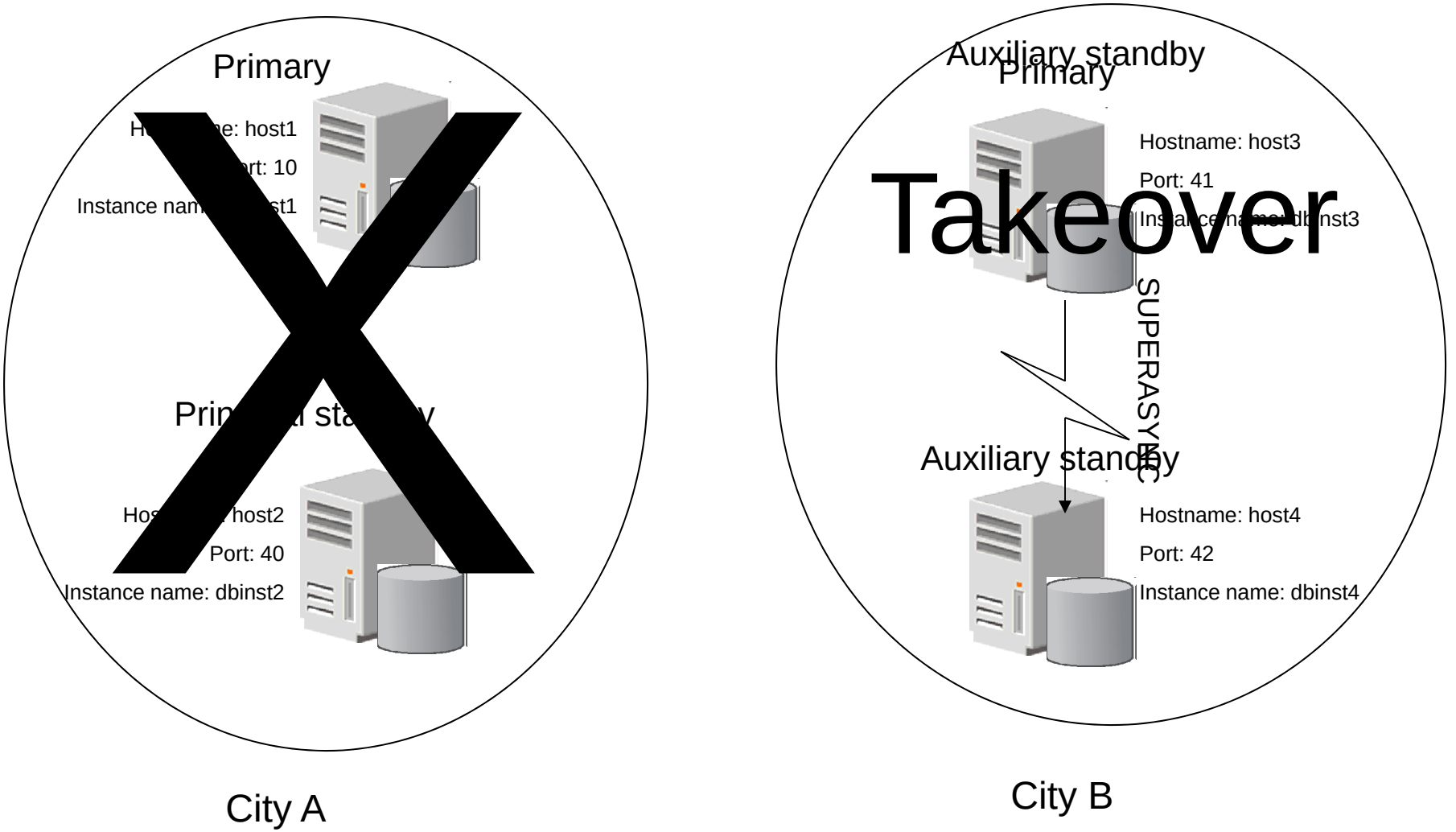
HADR Multiple Standby Role Reversal Example



After issuing takeover on host2 (auto reconfigured)

Configuration parameter	Host1	Host2	Host3	Host4
Hadr_target_list	host2:40 | host3:41 | host4:42	host1:10 | host3:41 | host4:42	Host2:40 | host1:10 | host4:42	host3:41 | host1:10 | host2:40
Hadr_remote_host	host2	host1	host2	host2
Hadr_remote_svc	40	10	40	40
Hadr_remote_inst	dbinst2	dbinst1	dbinst2	dbinst2
Hadr_local_host	host1	host2	host3	host4
Hadr_local_svc	10	40	41	42
Operational Hadr_syncmode	sync	sync	Near sync	Async
Effective Hadr_syncmode	sync	N/A	superasync	superasync

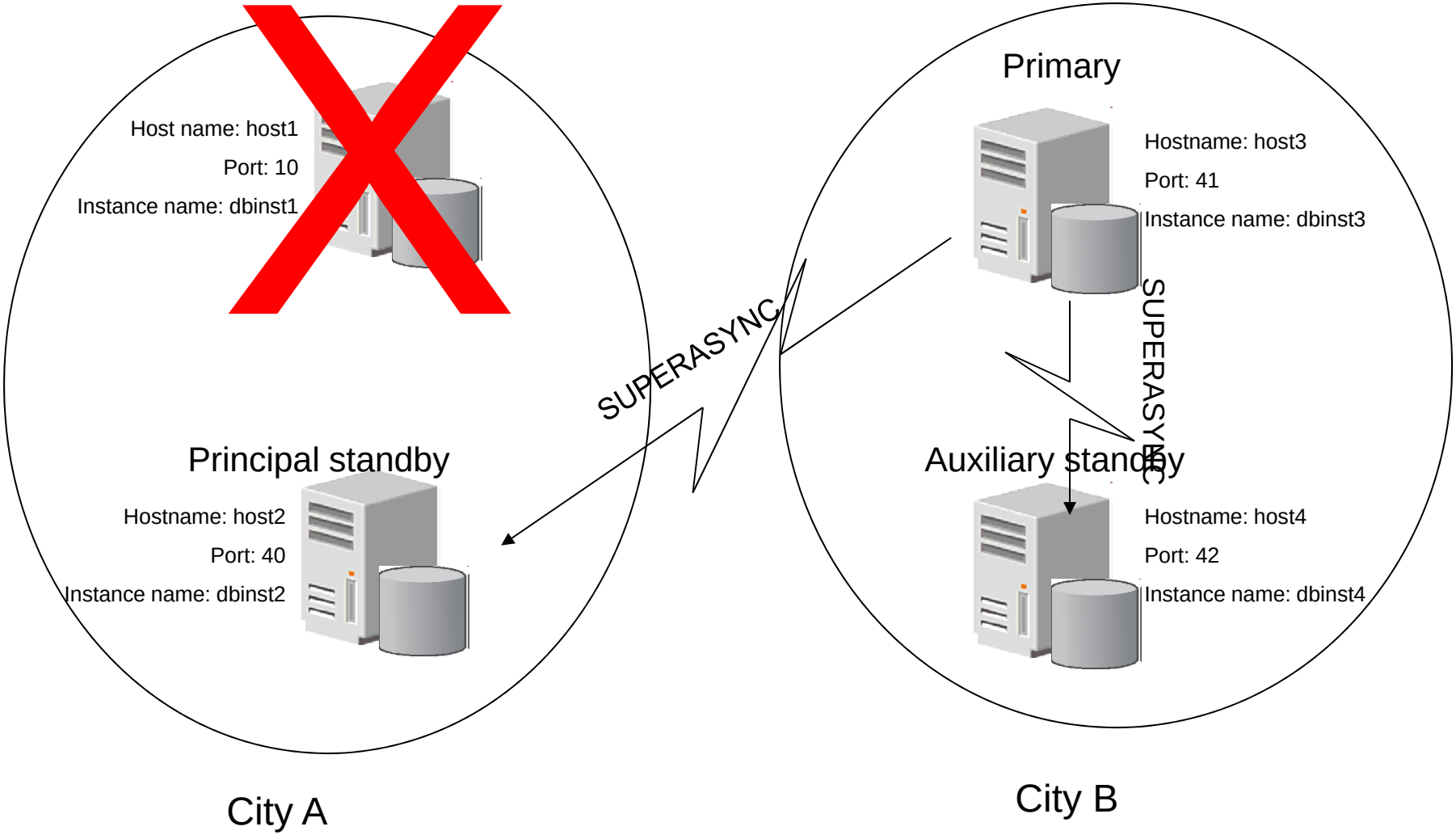
HADR Multiple Standby Forced Takeover Example



After issuing takeover on host3 (host 1+2 are down)

Configuration parameter	Host1	Host2	Host3	Host4
Hadr_target_list	host2 host3 host4	host1 host3 host4	host2:40 | host1:10 | host4:42	host3:41 | host1:10 | host2:40
Hadr_remote_host	host2	host1	host2	host3
Hadr_remote_svc	40	10	40	41
Hadr_remote_inst	dbins	dbins	dbinst2	dbinst3
Hadr_local_host	host1	host2	host3	host4
Hadr_local_svc	10	40	41	42
Operational Hadr_syncmode	sync	sync	Near sync	Async
Effective Hadr_syncmode	N/A	sync	n/a	superasync

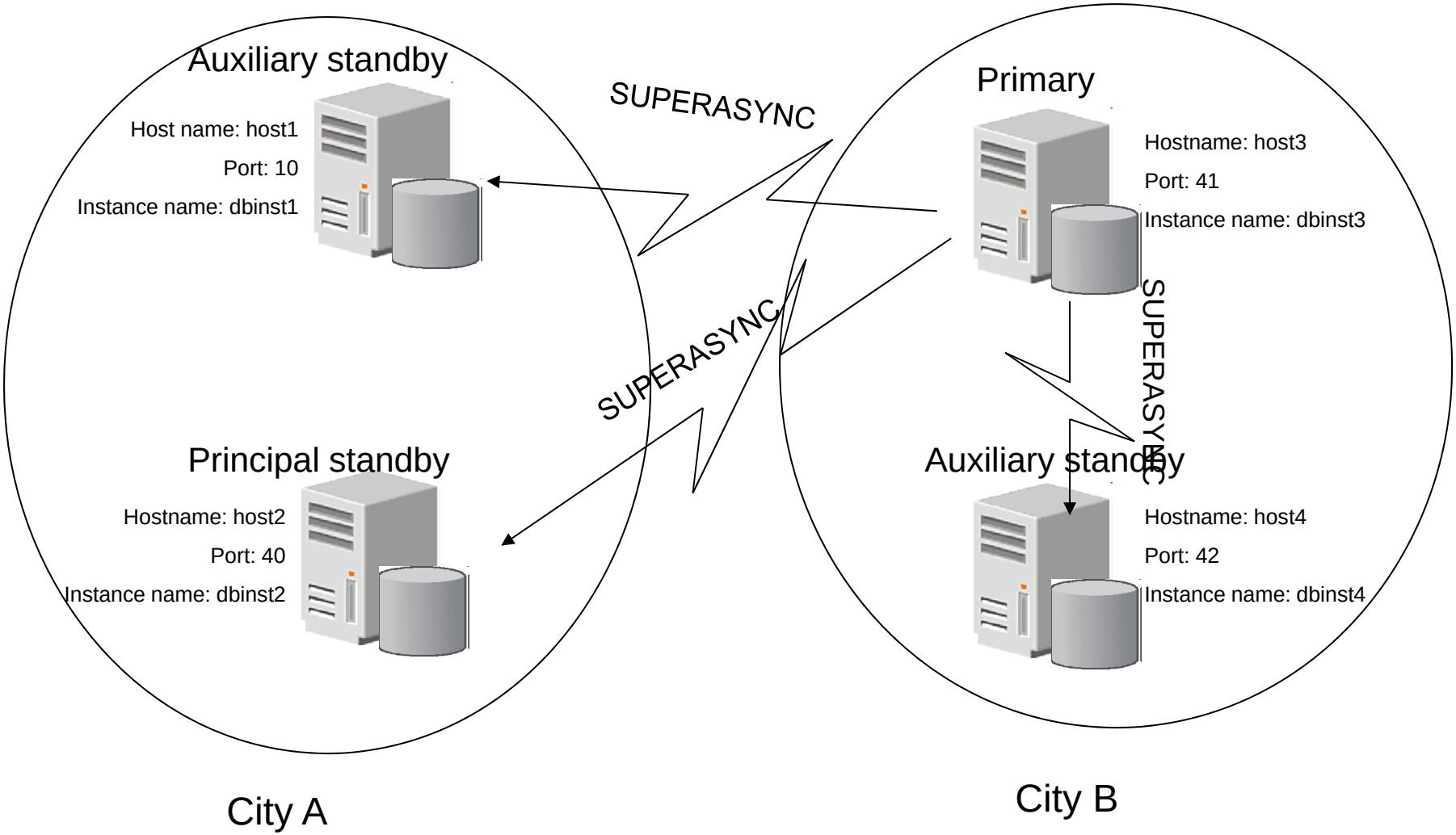
HADR Multiple Standby Example



After issuing takeover on host3 (host 2 online)

Configuration parameter	Host1	Host2	Host3	Host4
Hadr_target_list	host2 host3 host4	host1:10 | host3:41 | host4:42	host2:40 | host1:10 | host4:42	host3:41 | host1:10 | host2:40
Hadr_remote_host	host2	host3	host2	host3
Hadr_remote_svc	40	41	40	41
Hadr_remote_inst	dbins	dbinst3	dbinst2	dbinst3
Hadr_local_host	host1	host2	host3	host4
Hadr_local_svc	10	40	41	42
Operational Hadr_syncmode	sync	sync	Near sync	Async
Effective Hadr_syncmode	N/A	Nearsync	n/a	superasync

HADR Multiple Standby Example



After issuing takeover on host3 (host 1 online)

Configuration parameter	Host1	Host2	Host3	Host4
Hadr_target_list	host2:40 | host3:41 | host4:42	host1:10 | host3:41 | host4:42	host2:40 | host1:10 | host4:42	host3:41 | host1:10 | host2:40
Hadr_remote_host	host3	host3	host2	host3
Hadr_remote_svc	41	41	40	41
Hadr_remote_inst	dbinst3	dbinst3	dbinst2	dbinst3
Hadr_local_host	host1	host2	host3	host4
Hadr_local_svc	10	40	41	42
Operational Hadr_syncmode	sync	sync	Near sync	Async
Effective Hadr_syncmode	superasync	Nearsync	n/a	superasync

Agenda

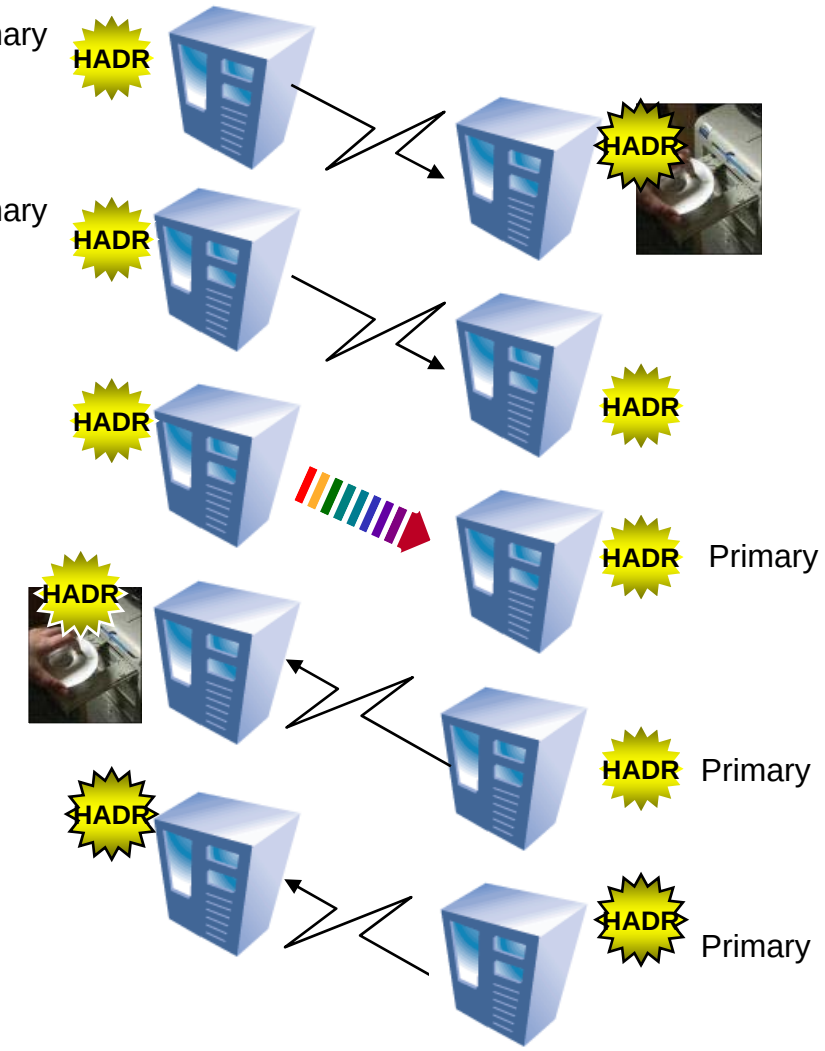
- WHAT is HADR
- WHY should I use HADR
- WHERE should I deploy it
- **WHEN should I use HADR**
- WHO is using HADR

When should I deploy HADR

- **If you require ultra easy HA and/or DR**
  - Use a single technology for both HA and DR
- **Once deployed other benefits arise**
  - Minimize outage from a software upgrade
  - Rehosting
    - Upgrade of Hardware
    - Upgrade of Operating System
  - Migration to pureScale

Software upgrades on the fly

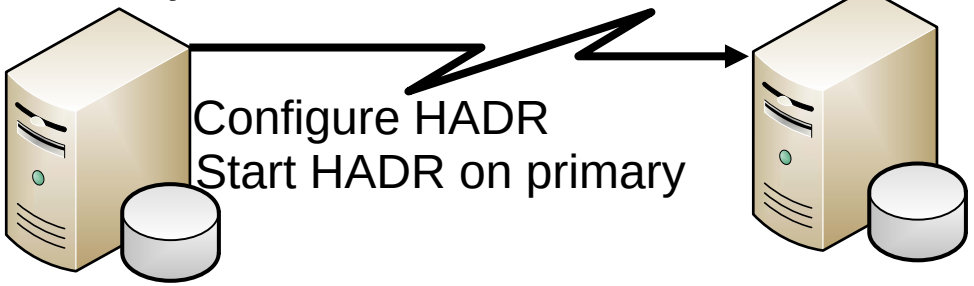
1. **HADR in peer state**
2. **Deactivate HADR on the Standby**
3. **Upgrade the standby**
4. **Start the standby again**
   - Let it catch-up with primary
5. **Issue a normal TAKEOVER**
   - The primary and standby change roles
6. **Deactivate the new standby**
7. **Upgrade the new standby**
8. **Reactivate the new standby**
   - Let it catch-up with primary
9. **Optionally, TAKEOVER again**
   - The primary and standby play their original roles



Rehosting – upgrading hardware, moving to a new data center, …

- **If the plan is to rehost the DB to a new set of H/W – be it either in the same DC or another DC, HADR can make the transition easier.**
- **If you are using a standalone ESE system then**
  - Create a instance on the new H/W (HostB)
  - Take an online backup of the existing DB (HostA)
  - Restore on the new H/W (HostB)
  - Configure the new DB as an principal standby for the original DB on HostA
  - Configure the original DB to be the HADR primary
  - Once the HADR systems are in peer state, issue a “normal” takeover
    - This is a zero-data loss roll reversal
  - Stop HADR on the new primary to convert it into a standalone DB
  - The original (HostA) database can be dropped

Rehosting – using HADR on an ESE system



Create instance
Restore DB2

Configure HADR
Start HADR on standby

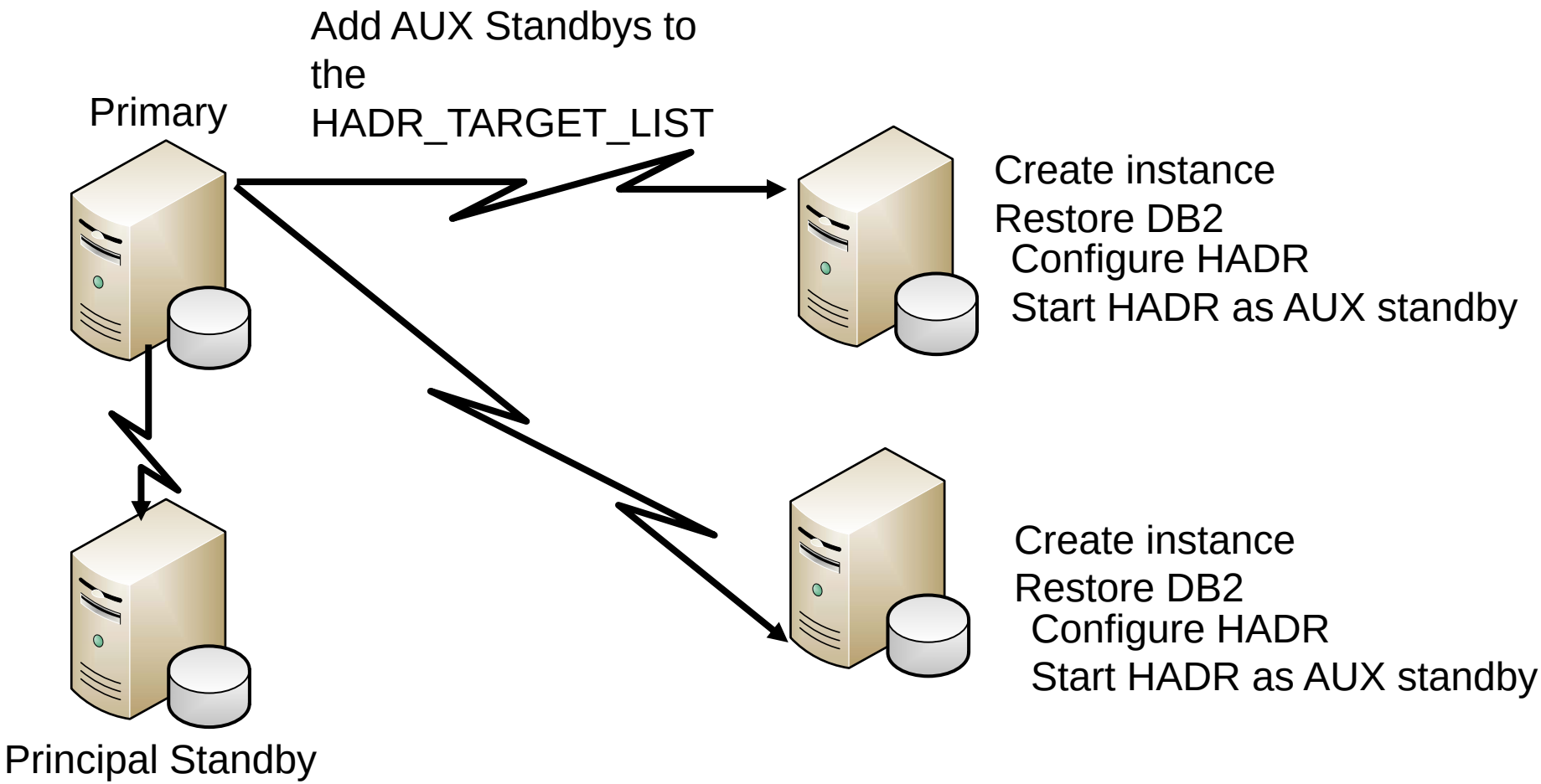
Wait until you reach PEER state
Issue takeover (zero data loss)
Stop HADR (changes DB to standalone)

Drop database

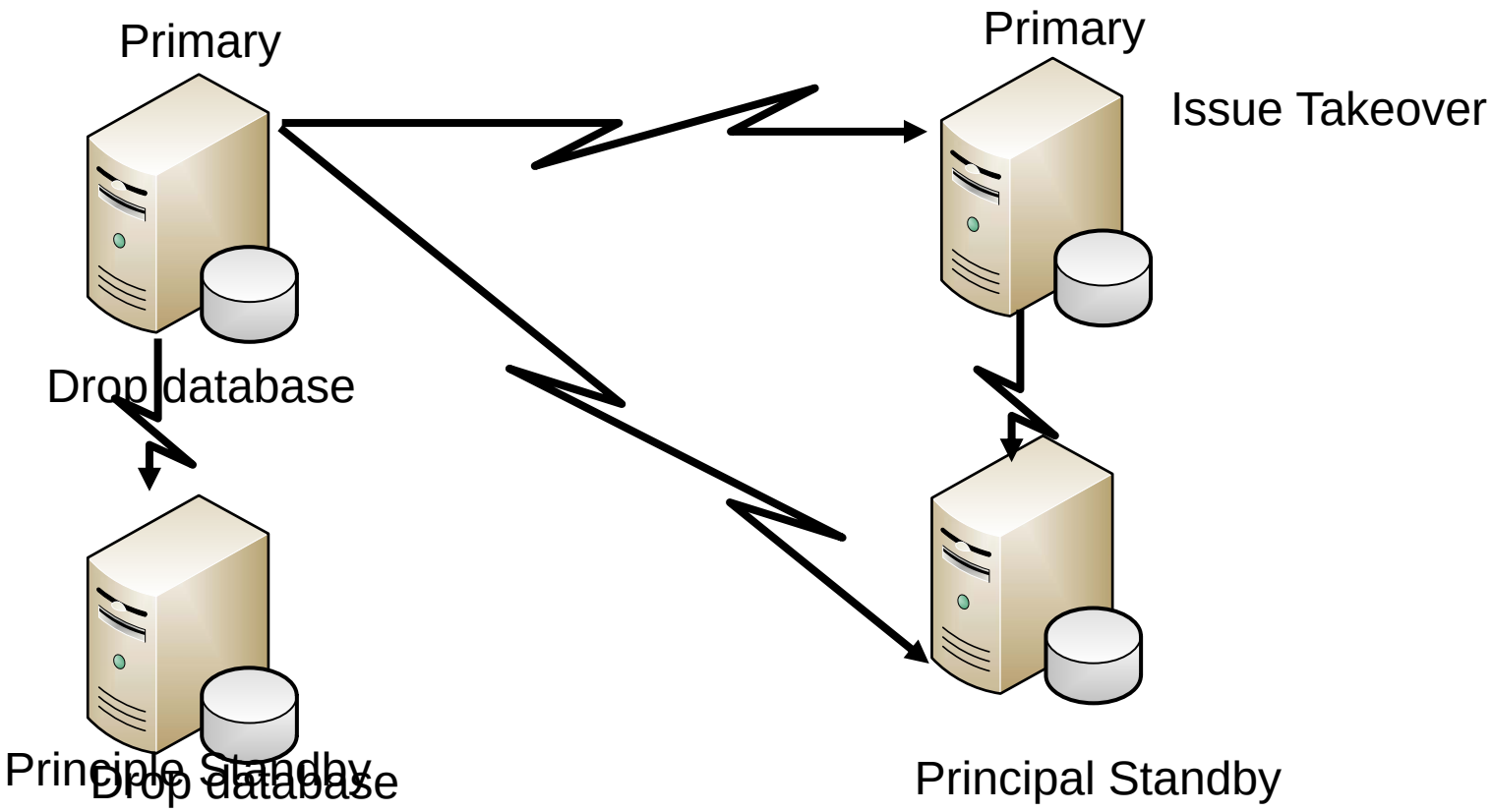
Rehosting – upgrading hardware, moving to a new data center, …

- **If you are using an HADR ESE system then**
  - Create a instance on the new H/W (HostC+HostD)
  - Take an online backup of the existing DB (HostA)
  - Restore on the new H/W (HostC+HostD)
  - Configure the new DBs(HostC+HostD) as an auxiliary standby for the original DB on HostA
    - Set the HADR_TARGET_LIST to reflect the new topology (Only HostC+HostD)
  - Configure the original DB add HostC+HostD as Auxiliary servers
  - Once the HADR systems are in peer state, issue a “normal” takeover
    - This is a zero-data loss roll reversal
    - Since HostA is not in the target list it will be orphaned
  - The original (HostA+HostB) database can be dropped

Rehosting – using HADR on an existing HADR environment



Rehosting – using HADR on an existing HADR environment



Conversion from HADR to pureScale

- **Existing DB2 ESE HADR customer wanted to convert from ESE nodes to pureScale to increase the resiliency at each site.**
- **Business requirement was to minimize the service interruption during the migration/deployment**
- **Current environment: DB2 V 10.5 ESE on RHEL 6.7 with a multi-TB SAP R3 system using HADR for resiliency using EXT4 filesystem**
- **Target environment: DB2 V 11.1.1.1 Purescale on RHEL 7.2 using 3 member and 2 CFs as well as HADR**
- **SAP procedure required substantial service interruption**
  - Setup GPFS cluster, creating the required filesystems
  - Use DB2 rebalance to move data from existing EXT4 filesystems to GPFS (this would be an online operation but significant impact on performance experienced)

SAP Documented upgrade procedure

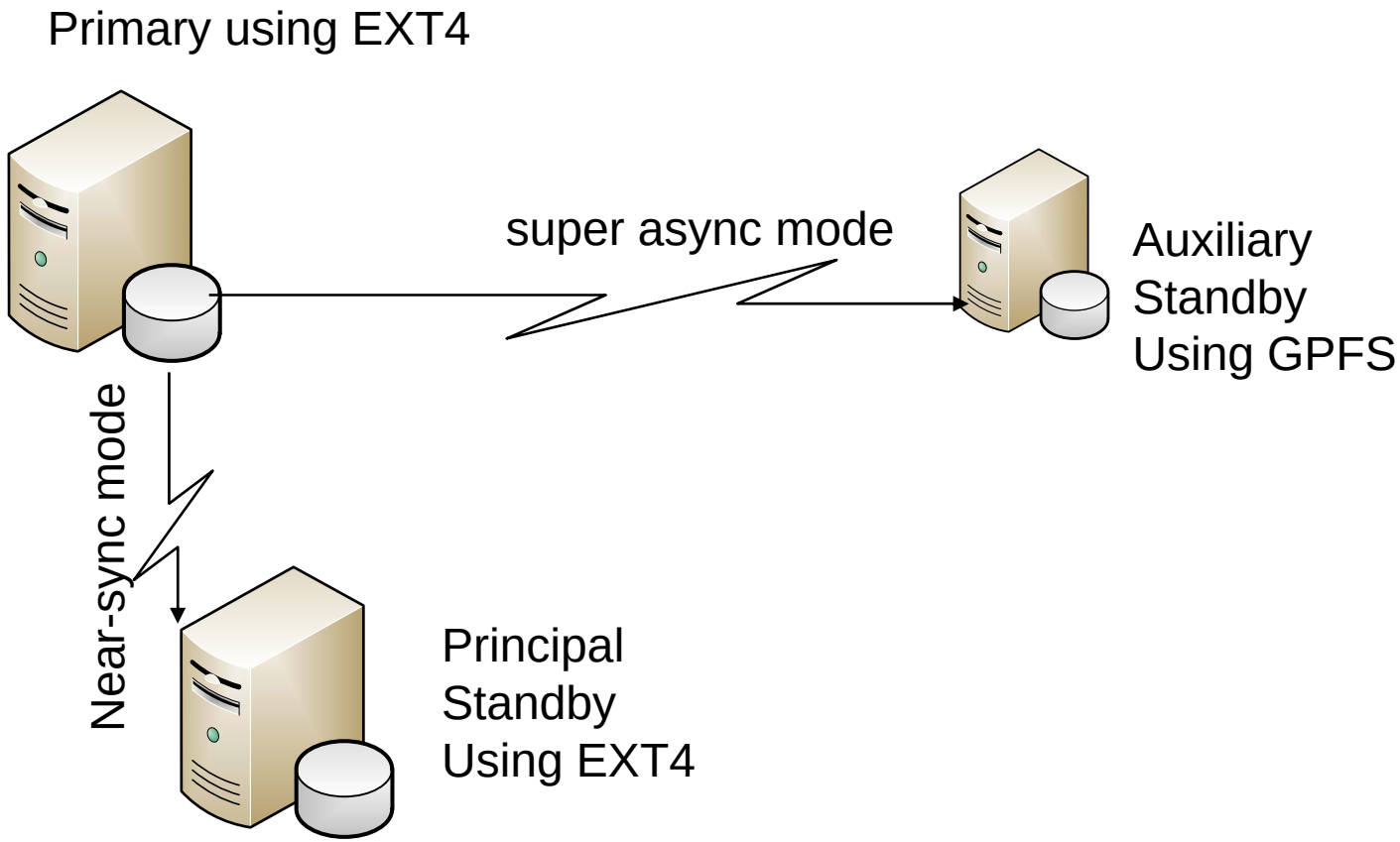
- Upgrade your Database to DB2 11.1
- Apply the Latest Kernel Patch
- Perform an offline backup
- Set the required configuration parameters
- Add the DB2 pureScale Feature to the existing DB2 11.1 software
- Prepare the cluster for the upgrade
- Manually set up passwordless access for user root
- Run the db2checkSD utility
- Prepare the GPFS cluster
- Create the necessary GPFS file systems
- Rebalance tablespaces to the new file system
- Move remaining data to the GFPSs
- Mount GFPSs under new mount points
- Convert the DB2 11.1 AESE instance to a DB2 pureScale instance
- Adapt the JDBC URL (AS Java only)
- Test the DB2 pureScale installation
- Add members and CFs
- Run the db6_update_db script
- Check configuration settings
- Install the DB2 pureScale license

Determined rebalance would require 20 days to complete, with significant performance impact on applications

Modified Conversion Procedure

- Upgrade existing production system to DB2 V 11.1 ESE – both primary and standby
  - Upgrade primary in place
  - Take an online backup of primary
  - Reinitialize the standby
- Add an AUX standby - but this AUX standby would only use GPFS (with the same mount points as the primary node)
- Stop production once the AUX is caught up to the production system
- Convert instance on the AUX standby to pureScale
- Add all CFs and pureScale members
- Restart production on the pureScale node
- Backup DB and use it to initialize the pureScale HADR standby

Add Aux Standby

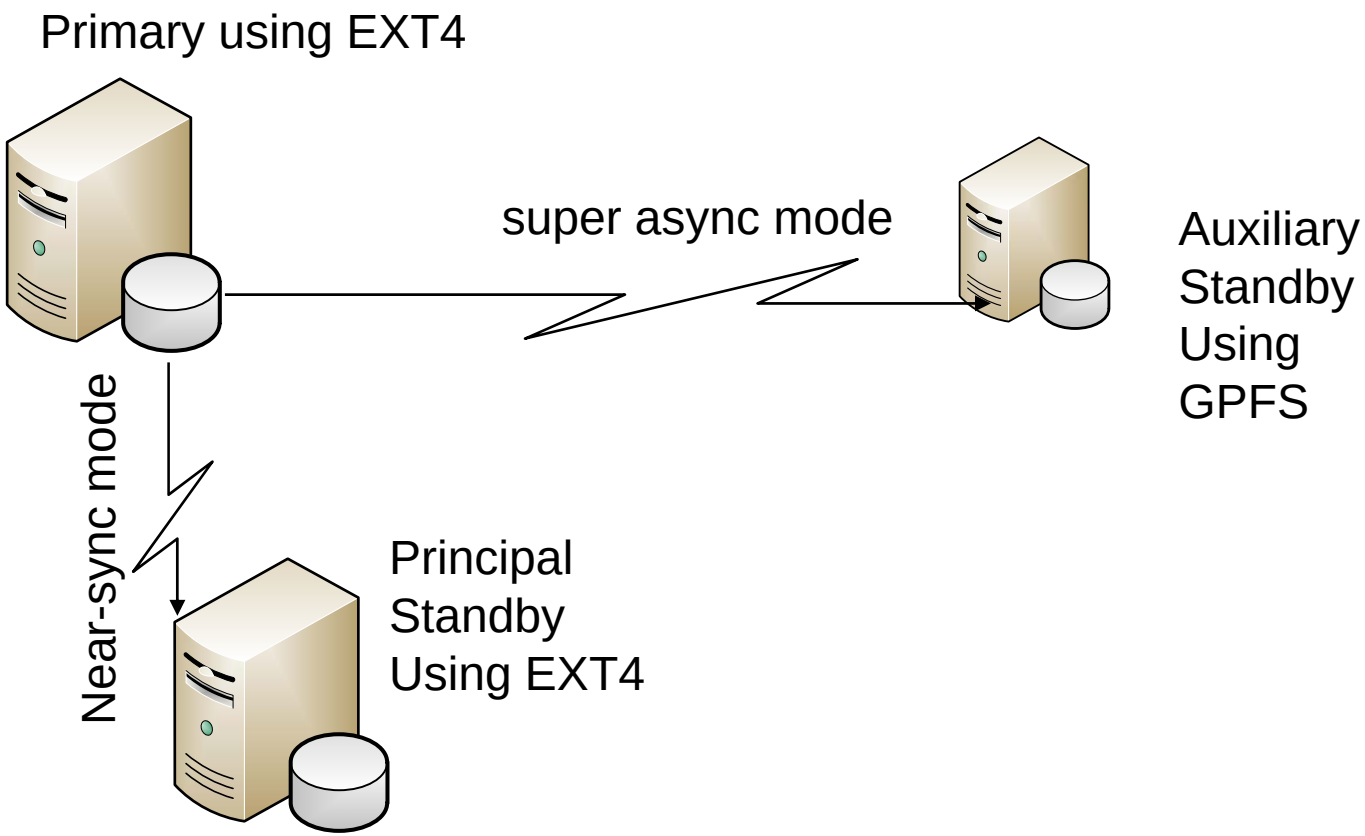


Using AUX standby to convert from HADR ESE to pureScale

Prereqs: On existing HADR Primary run db2checksd

- Will ensure that the DB is compatible with pureScale
  - No MCD, ITC or Range partitioned tables
  - Use of automatic storage exclusively
  - …
- Plan disk layout as per SAP documentation

Once AUX standby is caught up bring down primary and principal standby



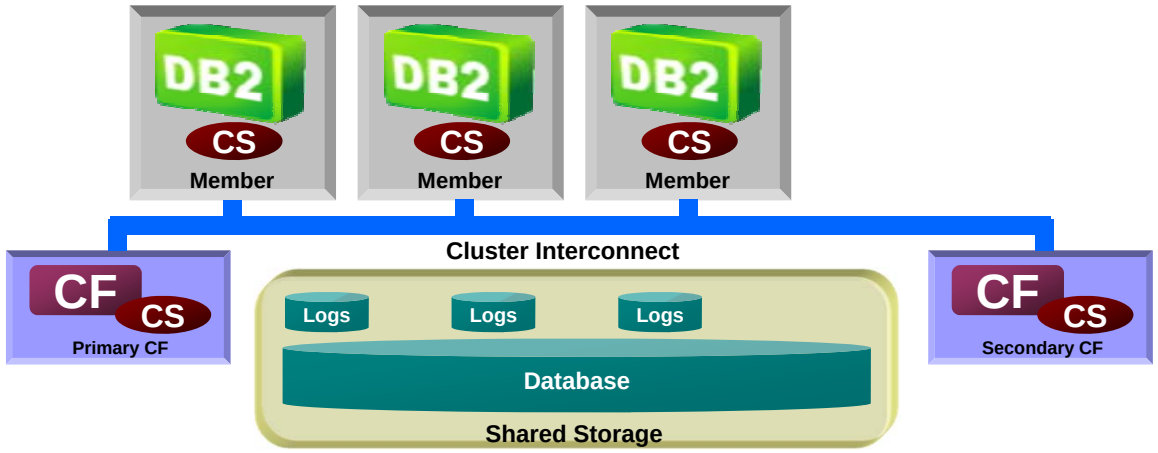
Convert AUX to standalone node – to become first member in the pureScale cluster

Standalone node

Auxiliary Standby Using GPFS

This node will become the first member in the pureScale cluster

Convert old AUX to pureScale instance adding 2 CF and 2 additional members



DB2 pureScale Cluster (Instance)

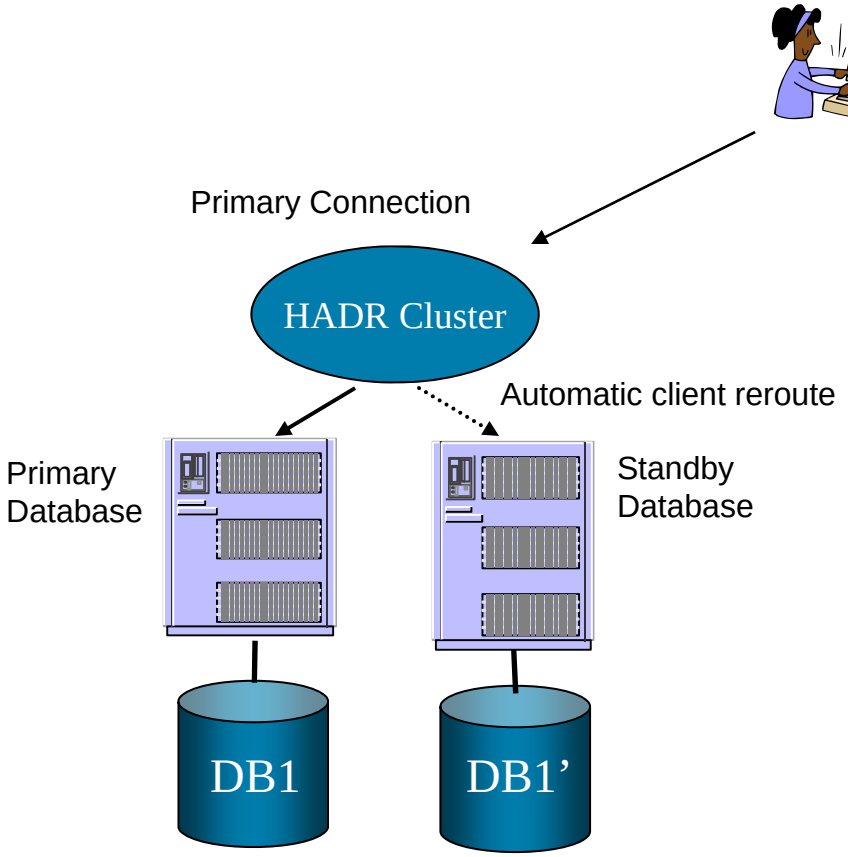
Agenda

- WHAT is HADR
- WHY should I use HADR
- WHERE should I deploy it
- WHEN should I use HADR
- **WHO is using HADR**

Who is using HADR?

- **Most of the Db2 OLTP customers who require HA have HADR deployed somewhere in the environment**
  - Financial sector
  - Manufacturing sector
  - Communications
  - Energy / Utilities
  - Transportation
- **HADR is often combined with other technologies such as**
  - Traditional Shared Disk Cluster
  - Storage Based Replication
  - Logical Replication, e.g. Q Repl, CDC

HADR Local or Remote



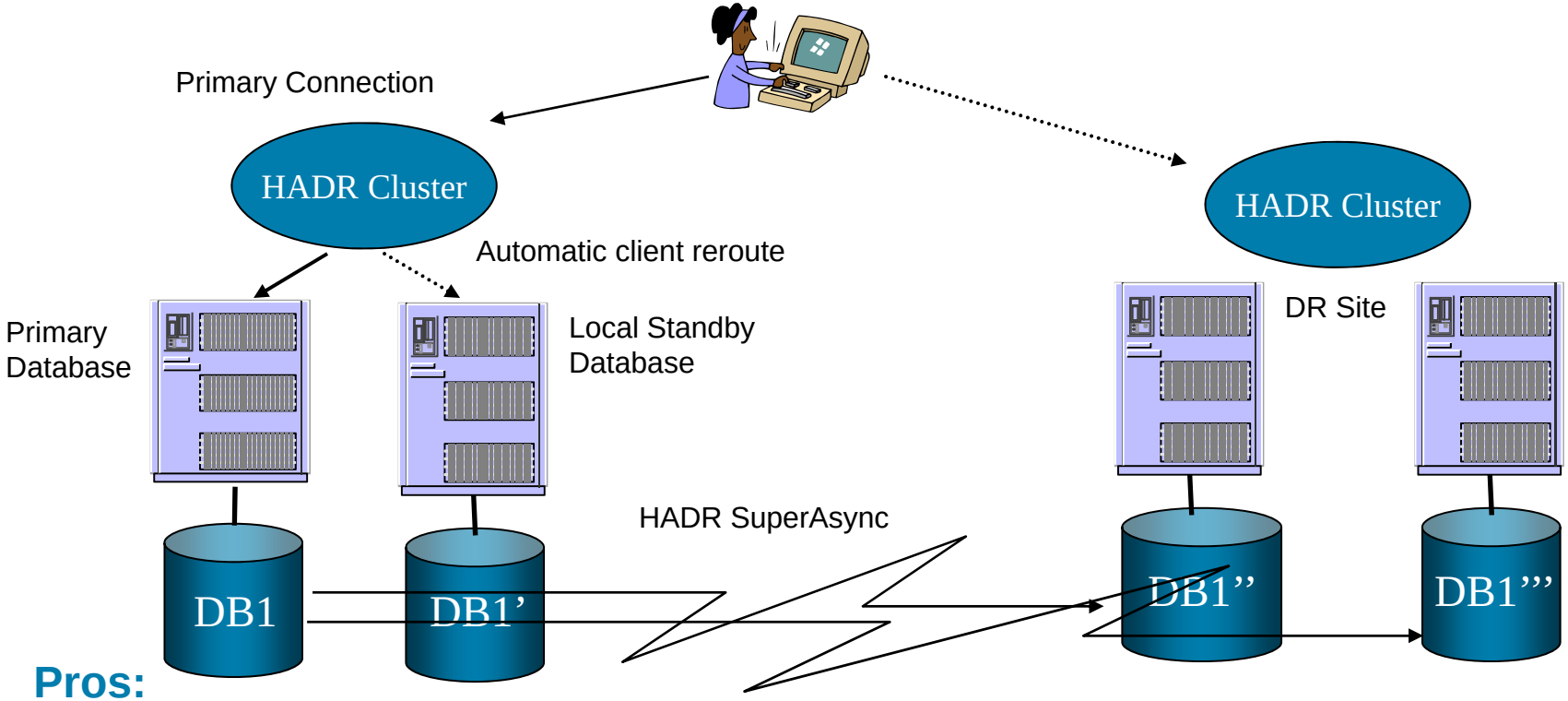
Pros:

- **Inexpensive local failover or DR solution**
- **Protection from software, server, storage or site failure**
- **Simple to setup and monitor**
- **Failover time in the range of 30 sec or less**

Cons:

- **Two full copies of the database (also a plus from a redundancy perspective)**
- **Standby database is not accessible (but standby server can be used for other purposes)**

HADR for both HA and DR



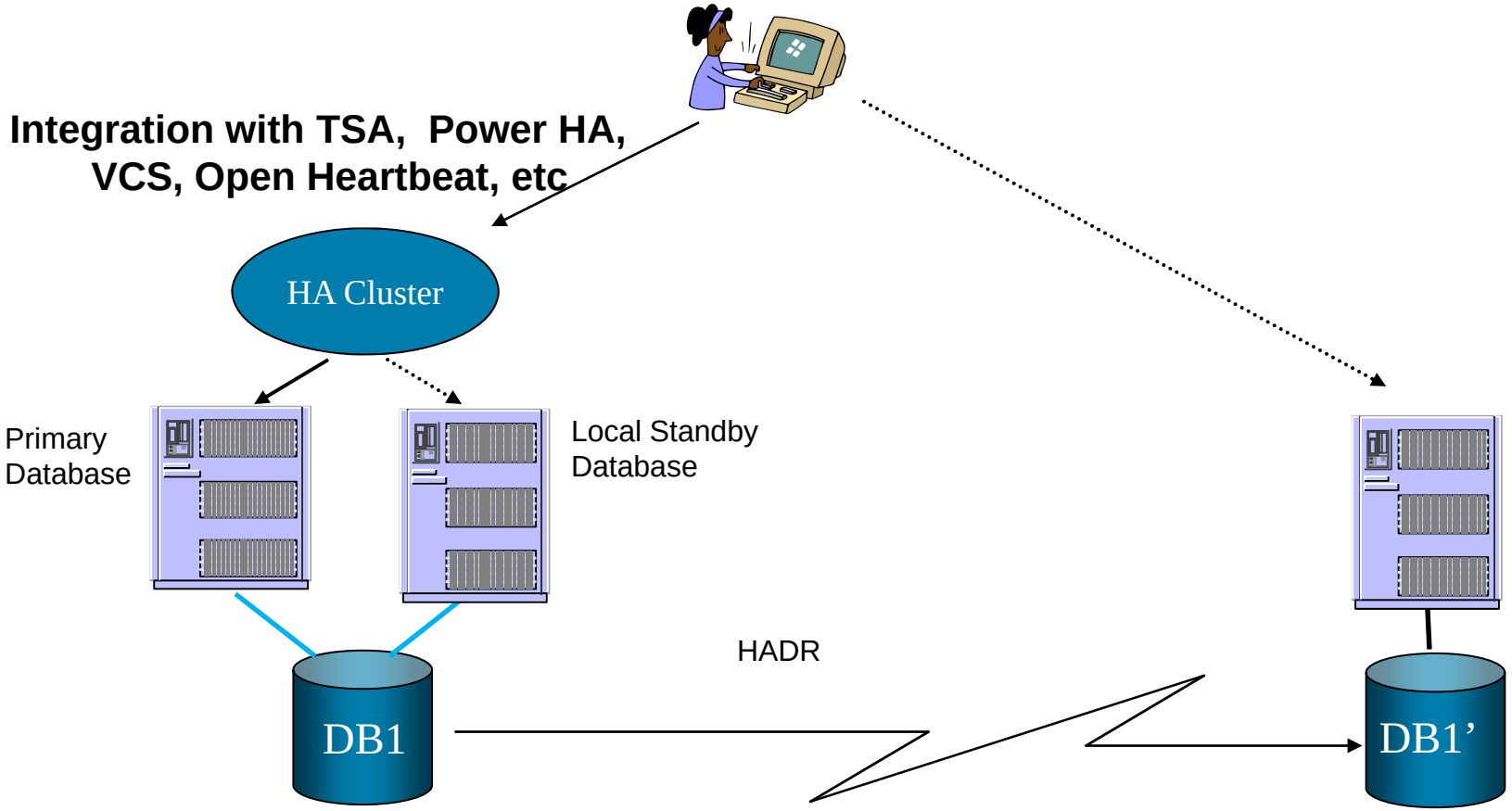
Pros:

- **Very fast local failover with DR ability as well**
- **Protection from software, server, storage or site failure**
- **Failover time in the range of 30 sec or less local**
- **Protection even in a DR scenario**

Cons:

- **Four full copies of the database**

Local cluster failover + HADR for DR



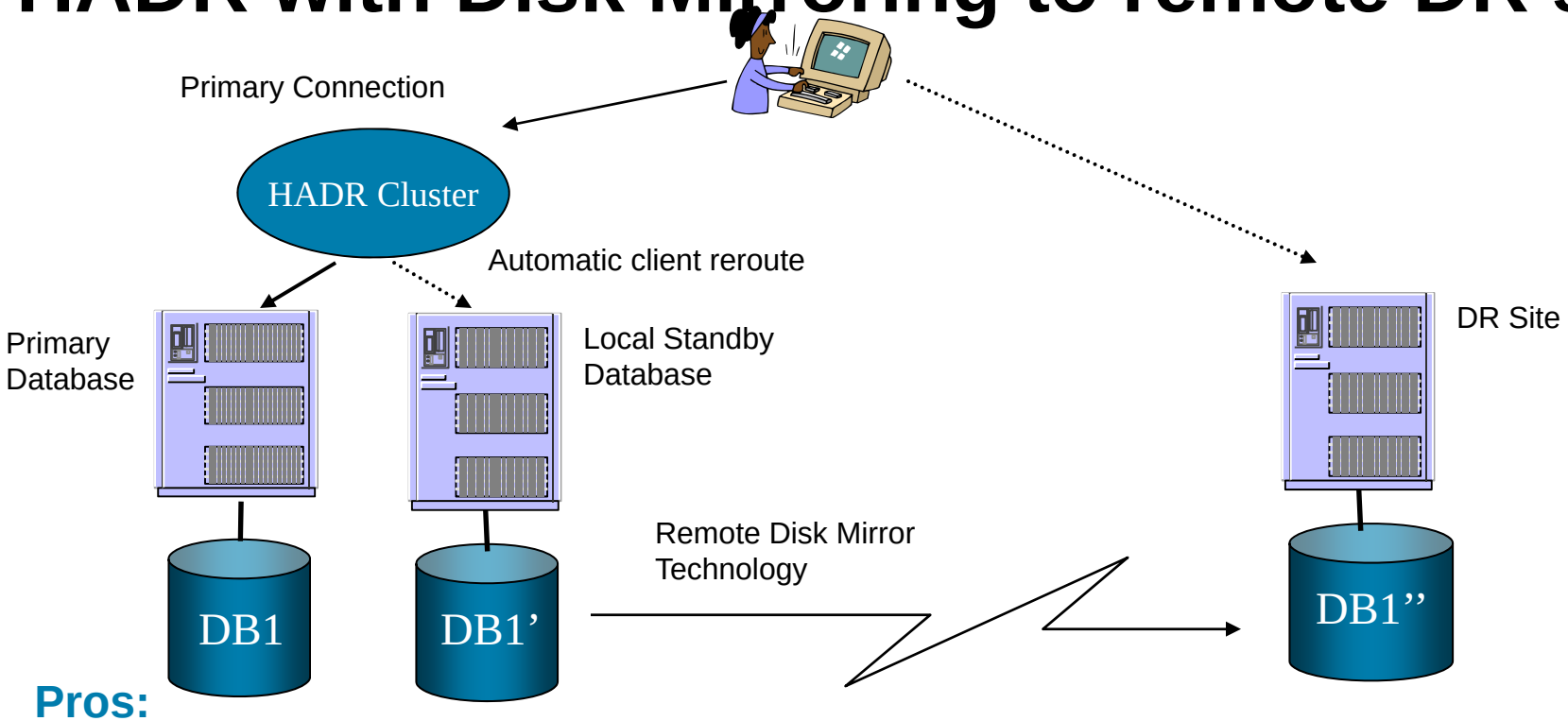
Pros:

- **Protection from software or server failure or site failure**

Cons:

- **Failover times vary from 1 to 5+ minutes**
- **Requires storage capable of remote disk mirroring**

HADR with Disk Mirroring to remote DR site



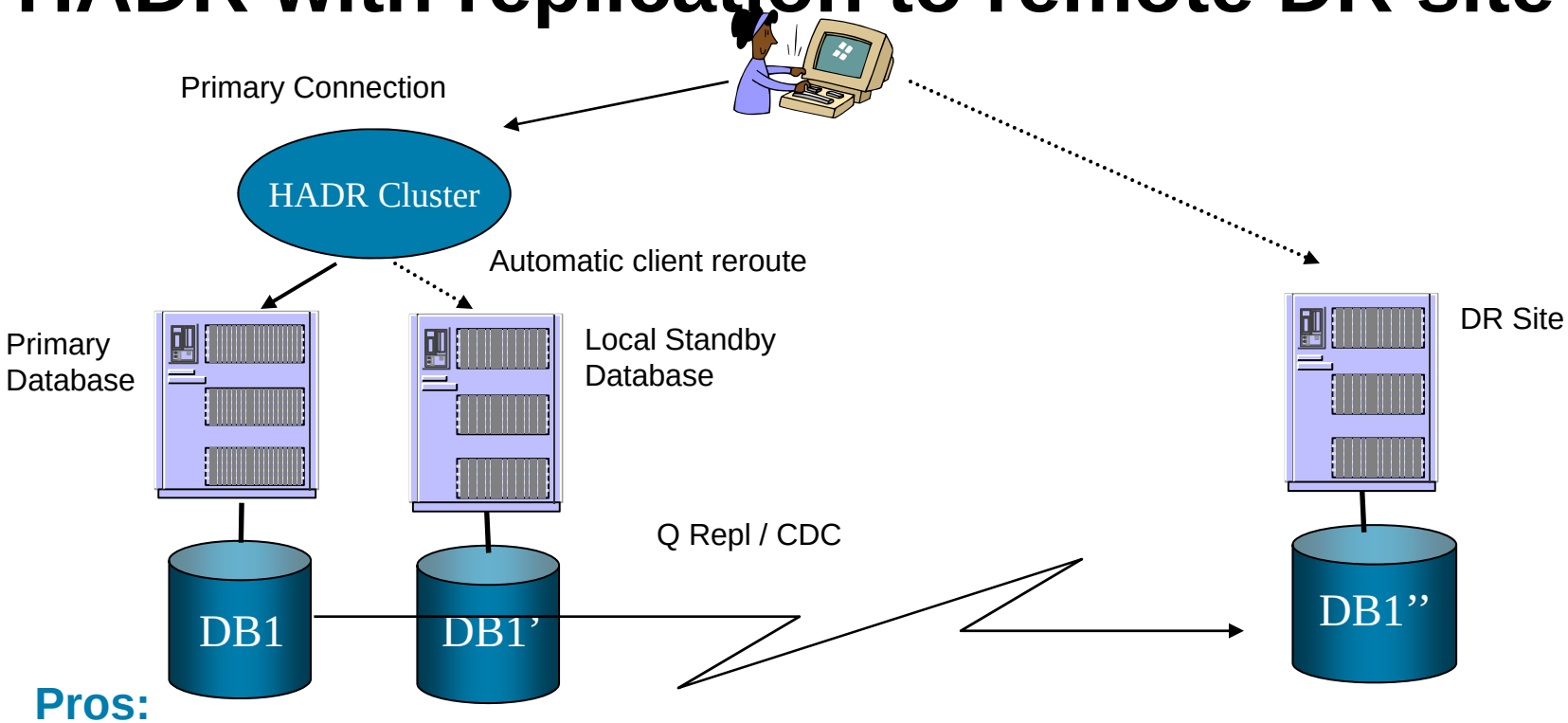
Pros:

- **Very fast local failover with DR ability as well**
- **Protection from software, server, storage or site failure**
- **Failover time in the range of 30 sec or less local**

Cons:

- **Three full copies of the database (also a plus from a redundancy perspective)**
- **More costly than HADR for just DR**

HADR with replication to remote DR site



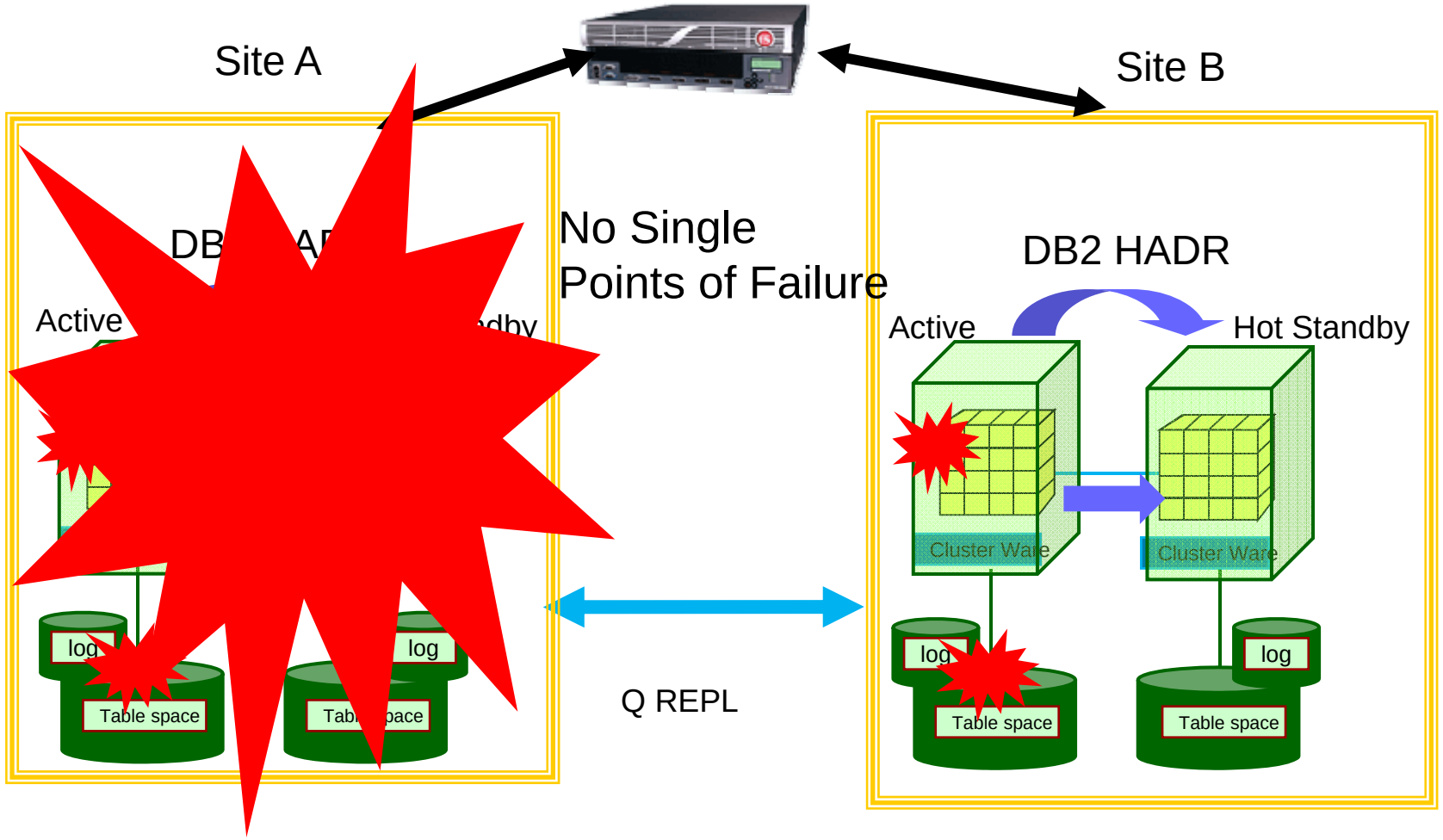
Pros:

- **Very fast local failover with DR ability as well**
- **Protection from software, server, storage or site failure**
- **Failover time in the range of 30 sec or less local**
- **Full access to the DR site**

Cons:

- **Three full copies of the database (also a plus from a redundancy perspective)**
- **More costly than HADR for just DR**

Implement HADR in both DCs with Q Repl between DCs - Active / Active



Dale McInnis

IBM Canada Ltd.

dmcinnis@ca.ibm.com

The “5 Ws” of HADR