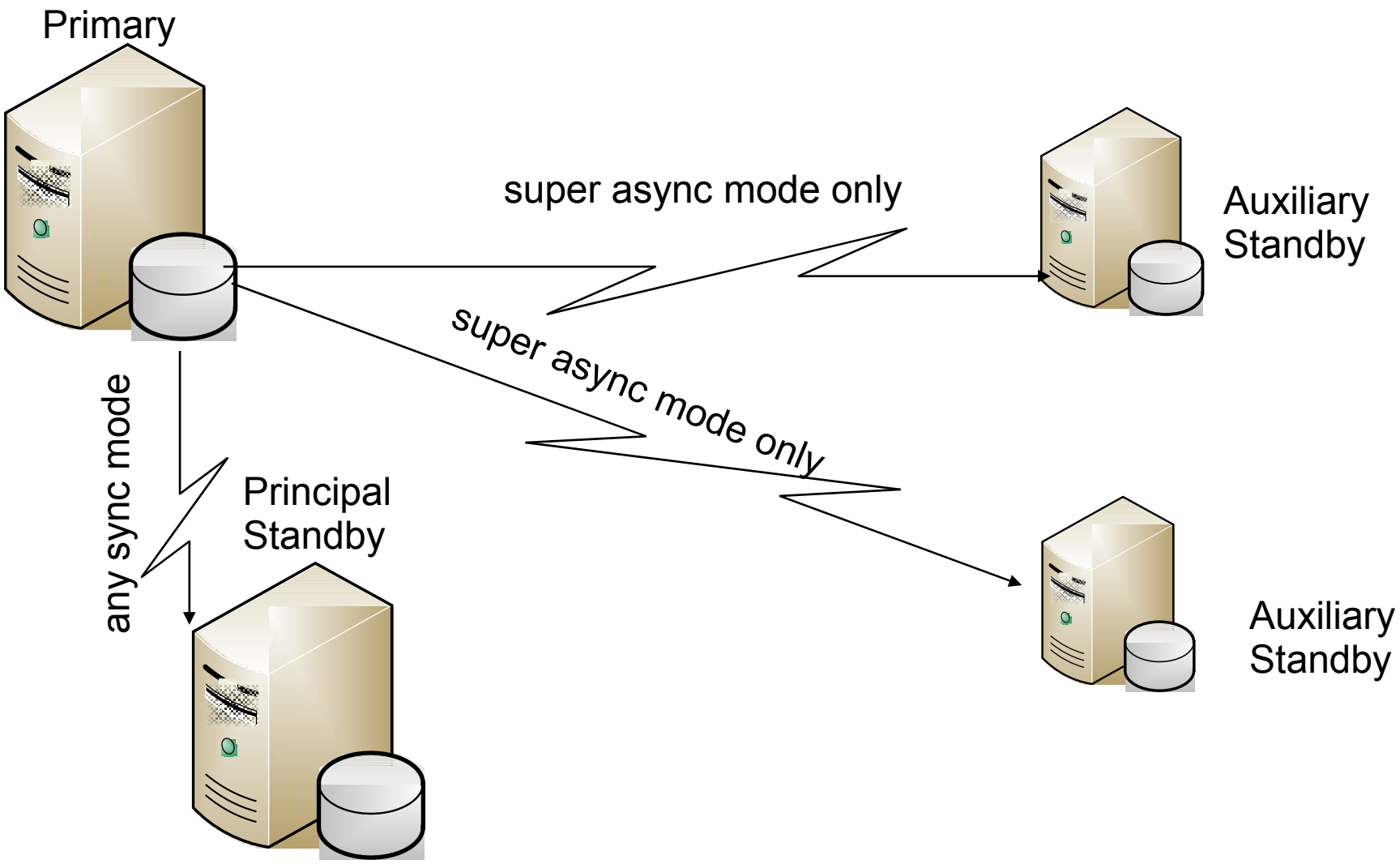
HADR Update: Multiple Standby Support

Dale McInnis
IBM Canada Ltd.

Session Code: H03
Wed. May 16: 3:15 – 4:15 pm| Platform: Linux, UNIX, Windows

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 Enablement

- Principal Standby (PS) is specified via the HADR_REMOTE_HOST, HADR_REMOTE_SVC, HADR_REMOTE_INST
- HADR_TARGET_LIST is used to specify all standbys, both auxiliary as well as the principal standby
- HADR_TARGET_LIST uses a hostname or IP Address and port number format with the "|" character as a delimiter
  - E.g. host1.ibm.com:4000|host2.ibm.com:hadr_service|9.47.73.34:5000
- On each standby the HADR_REMOTE_HOST, HADR_REMOTE_INST, HADR_REMOTE_SVC must point to the current primary
- Primary will validate hostname and port number upon handshake from AS
- Existing single standby installations need no configuration change

Multiple Standby Configuration Setup

On each node (primary and all standbys) set the local configuration information

"UPDATE DB CFG FOR *dbname* USING
HADR_LOCAL_HOST *hostname*
HADR_LOCAL_SVC *servicename*
HADR_SYNCMODE *syncmode*“

Set the hadr_target_list configuration parameter on all of the standbys and the primary.

“DB2 UPDATE DB CFG FOR *dbname* USING
HADR_TARGET_LIST *principalhostname*:*principalservicename*|*auxhostname1*:*auxservicename1*|*auxhostname2*:*auxservicename2*”

These values are to be set with respect to how the cluster should appear if this node became the primary

Multiple Standby Configuration Setup

Optional step:

- On the primary, set the parameters to the corresponding values on the principal standby by issuing the following command:

  DB2 "UPDATE DB CFG FOR *dbname* USING
  HADR_REMOTE_HOST *principalhostname*
  HADR_REMOTE_SVC *principalservicename*
  HADR_REMOTE_INST *principalinstname*"

- On each standby, set the parameters to the corresponding values on the primary by issuing the following command:

  DB2 "UPDATE DB CFG FOR *dbname* USING
  HADR_REMOTE_HOST *primaryhostname*
  HADR_REMOTE_SVC *primaryservicename*
  HADR_REMOTE_INST *primaryinstname*"

Automatic reconfiguration of HADR Parameters

Reconfiguration after HADR starts

Configuration parameters that identify the primary database for the standbys and identify the principal standby for the primary are automatically reset when HADR starts if you did not correctly set them. This behavior applies to the following configuration parameters:

- **hadr_remote_host**
- **hadr_remote_inst**
- **hadr_remote_svc**

Reconfiguration during and after a takeover

After a forced or non-forced takeover, the values for the **hadr_remote_host, hadr_remote_inst, and hadr_remote_svc** configuration parameters are updated automatically on all the databases that are potentially a part of the new setup. Any database that is not a valid standby for the new primary, because they are not included in each other's target list, is not updated. If you want to include a database as a standby, you must ensure that it is in the target list of the primary and that the primary is in the target list of the new standby database. Otherwise, the standby database waits for the old primary to restart as a primary.

Takeover Behavior in Multiple Standby Environment

Two types of takeovers as in traditional HADR

- Role Switch
  - sometimes called graceful takeover or non-forced takeover, can be performed only when the primary is available and it switches the role of primary and standby Provides zero data loss guarantee
- Failover
  - can be performed when the primary is not available. It is commonly used in primary failure cases to make the standby the new primary. The old primary remains in primary role in a forced takeover.
- Both types of takeover are supported in multiple standby mode, and any of the standby databases can take over as the primary. A crucial thing to remember, though, is that if a standby is not included in the new primary's target list, it is considered to be orphaned and cannot connect to the new primary.

Takeover Behavior in Multiple Standby Environment

- Takeover (forced and non-forced) is supported on all standbys
- After a takeover, DB2 auto-redirects and makes necessary configuration changes (host/service/instance name of new primary) on the standbys that are in the new primary's target list (and vice versa)
- Standbys not in new primary's target list (and vice versa) are “orphaned” standbys
- Data loss (usually from a failover) complicates the picture:
  - If old primary has more data than new primary, it cannot be reintegrated without being reinitialized
  - If a standby has more data than new primary, it will not pass pair validation check and cannot become a standby for that primary
  - To avoid the latter, check the log positions of all standbys and fail over to the one with the most data

Takeover Behavior in Multiple Standby Environment

DB2 automatically makes a number of configuration changes for you so that the standbys listed in new primary's target list can connect to the new primary. The hadr_remote_host, hadr_remote_svc, and hadr_remote_inst configuration parameters are updated on the new primary and listed standbys in the following way

- **On the new primary:** They refer to the principal standby (the first database listed in the new primary's target list).

- **On the standbys:** They refer to the new primary. When an old primary is reintegrated to become standby, the START HADR AS STANDBY command first converts it to a standby. Thus it can also be automatically redirected to the new primary if it is listed in the target list of the new primary.

- Orphaned standbys are not automatically updated in this way. If you want them to join as standbys, you need to ensure they are in the new primary's target list and that they include the new primary in their target lists.

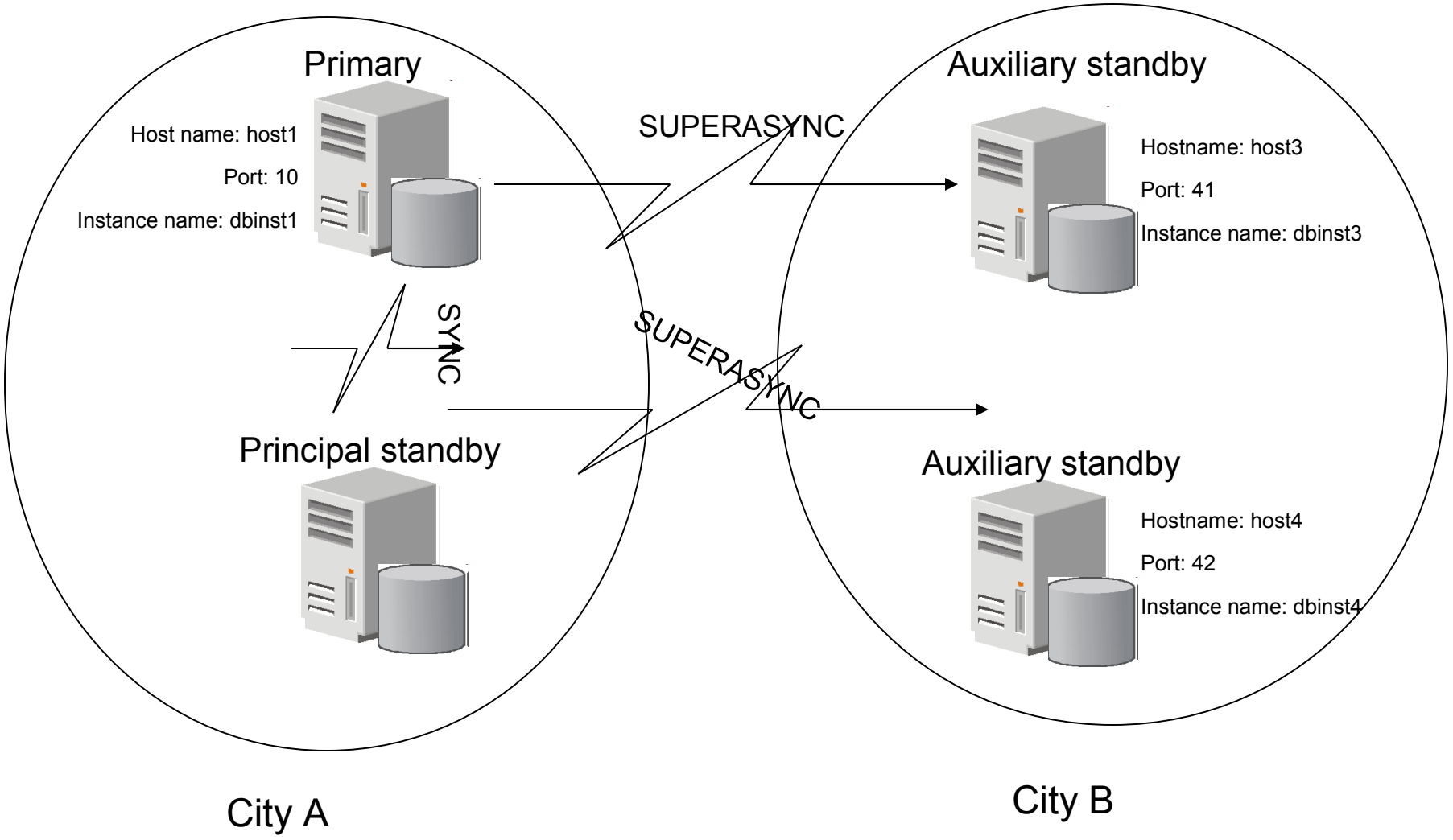
Takeover Behavior in Multiple Standby Environment

- Role switch
  - Just as in single standby mode, role switch in multiple standby mode guarantees no data is lost between the old primary and new primary. Other standbys configured in the new primary's hadr_target_list configuration parameter are automatically redirected to the new primary and continue receiving logs.
- Failover
  - Just as in single standby mode, if a failover results in any data loss in multiple standby mode (meaning that the new primary does not have all of the data of the old primary), the old and new primary's log streams diverge and the old primary has to be reinitialized. For the other standbys, if a standby received logs from the old primary beyond the diverge point, it has to be reinitialized. Otherwise, it can connect to the new primary and continue log shipping and replay. As a result, it is very important that you check the log positions of all of the standbys and choose the standby with the most data as the failover target. You can query this information using the db2pd command or the MON_GET_HADR table function.

Multiple Standby Restrictions

- You can have a maximum of three standby databases: one principal standby and one or two auxiliary standbys
- Only the principal standby supports all the HADR synchronization modes; all auxiliary standbys will be in SUPERASYNC mode
- IBM Tivoli System Automation for Multiplatforms (SA MP) support applies only between the primary HADR database and its principal standby
- You must set the **hadr_target_list database configuration parameter on all the** databases in the multiple standby setup. In addition, for each combination of primary and standby, role switch between those databases must be allowed. That is, each database in the target list of a particular database must also have that particular database in its target list.

HADR Multiple Standby Example



Multiple Standby Example

Intended role	Host name	Port #	Instance name
Primary	host1	10	dbinst1
Principal standby	host2	40	dbinst1
Auxiliary standby	host3	41	dbinst3
Auxiliary standby	host4	42	dbinst4

Initial Setup

Step 1: Create a backup image:

On host1 (primary)

DB2 BACKUP DB HADR_DB to /nfs/db_backup

Step 2: Initialize the standbys:

on each of host2, host3, and host 4

DB2 RESTORE DB HADR_DB from /nfs/db_backup

Step 3: Update the configuration

The HADR_LOCAL_HOST, HADR_LOCAL_SVC and HADR_SYNCMODE must be set on all nodes

the HADR_TARGET_LIST should be set to the following on all nodes:

PrincipalHost:Principalsvc | auxhost1:auxsvc1 | auxhost2:auxsvc2

Initial Setup

Step 4: Update the configuration optional step

The following are not required to be set in a multiple standby environment as they will be automatically set:

- hadr_remote_host
- hadr_remote_svc
- hadr_remote_inst

On the primary set the following:

HADR_REMOTE_HOST = principalhostname
HADR_REMOTE_SVC = principalservicename
HADR_REMOTE_INST= principalinstancename

On each standby set the following:

HADR_REMOTE_HOST = primaryhostname
HADR_REMOTE_SVC = primaryservicename
HADR_REMOTE_INST= primaryinstancename

Initial Setup

On host1 (the primary):

```
DB2 "UPDATE DB CFG FOR hadr_db USING
HADR_TARGET_LIST host2:40|host3:41|host4:42
HADR_REMOTE_HOST host2
HADR_REMOTE_SVC 40
HADR_LOCAL_HOST host1
HADR_LOCAL_SVC 10
HADR_SYNCMODE sync
HADR_REMOTE_INST db2inst2"
```

Initial Setup con't

On host2 (the principal standby):

```
DB2 "UPDATE DB CFG FOR hadr_db USING
HADR_TARGET_LIST host1:10|host3:41|host4:42
HADR_REMOTE_HOST host1
HADR_REMOTE_SVC 10
HADR_LOCAL_HOST host2
HADR_LOCAL_SVC 40
HADR_SYNCMODE sync
HADR_REMOTE_INST db2inst1“
```

On host3 (an auxiliary standby):

```
DB2 "UPDATE DB CFG FOR hadr_db USING
HADR_TARGET_LIST host2:40|host1:10|host4:42
HADR_REMOTE_HOST host1
HADR_REMOTE_SVC 10
HADR_LOCAL_HOST host3
HADR_LOCAL_SVC 41
HADR_SYNCMODE superasync
HADR_REMOTE_INST db2inst1“
```

Initial Setup con't

On host4 (an auxiliary standby):

```
DB2 "UPDATE DB CFG FOR hadr_db USING
HADR_TARGET_LIST host2:40|host1:10|host3:41
HADR_REMOTE_HOST host2
HADR_REMOTE_SVC 10
HADR_LOCAL_HOST host4
HADR_LOCAL_SVC 42
HADR_SYNCMODE superasync
HADR_REMOTE_INST db2inst1
HADR_REPLAY_DELAY 86400"
```

Starting HADR

Starting the HADR databases

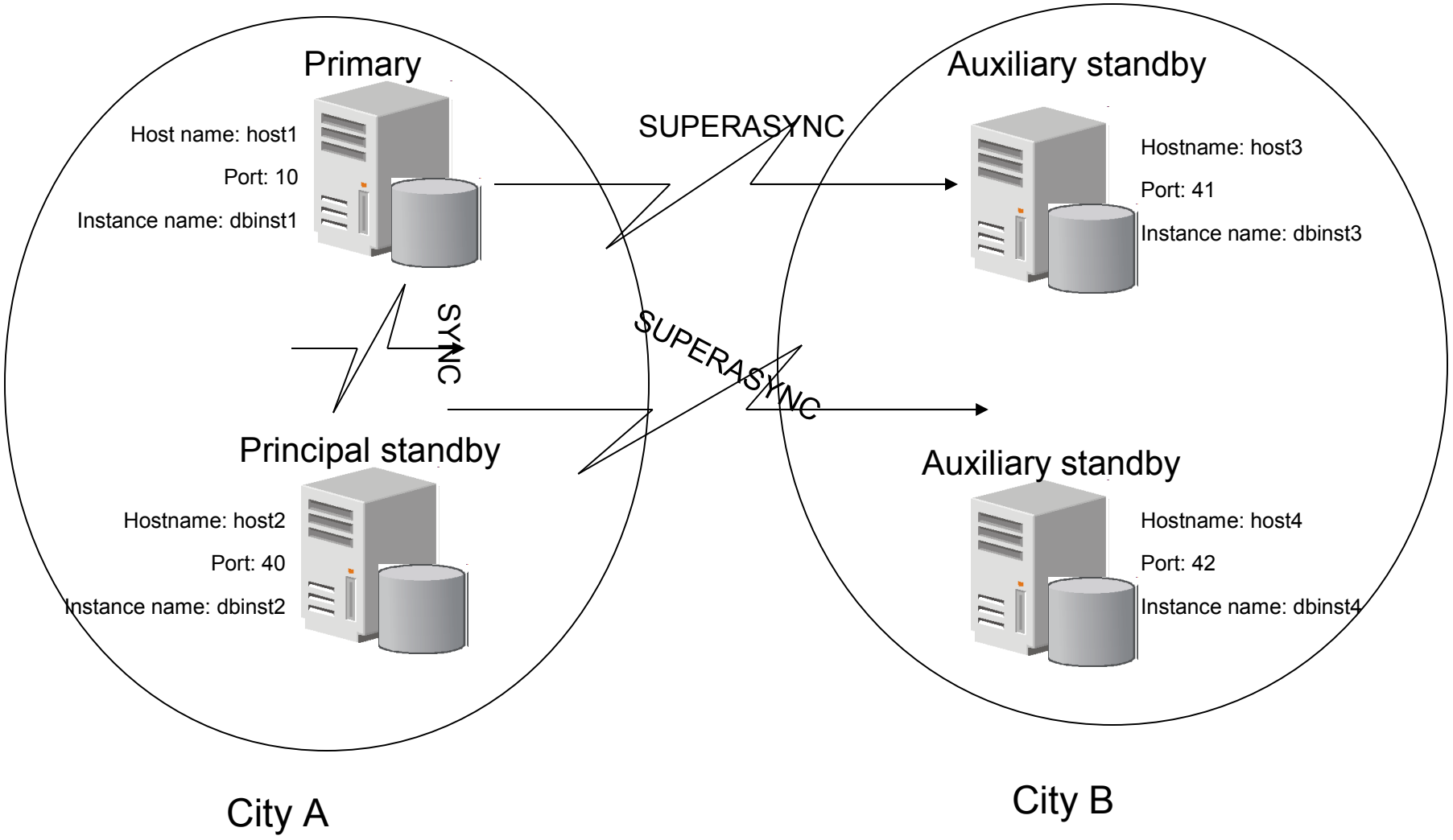
The DBA starts the standby databases first, by issuing the following command on each of host2, host3, and host 4:

```
DB2 START HADR ON DB hadr_db AS STANDBY
```

Next, the DBA starts HADR on the primary database, on host1:

```
DB2 START HADR ON DB hadr_db AS PRIMARY
```

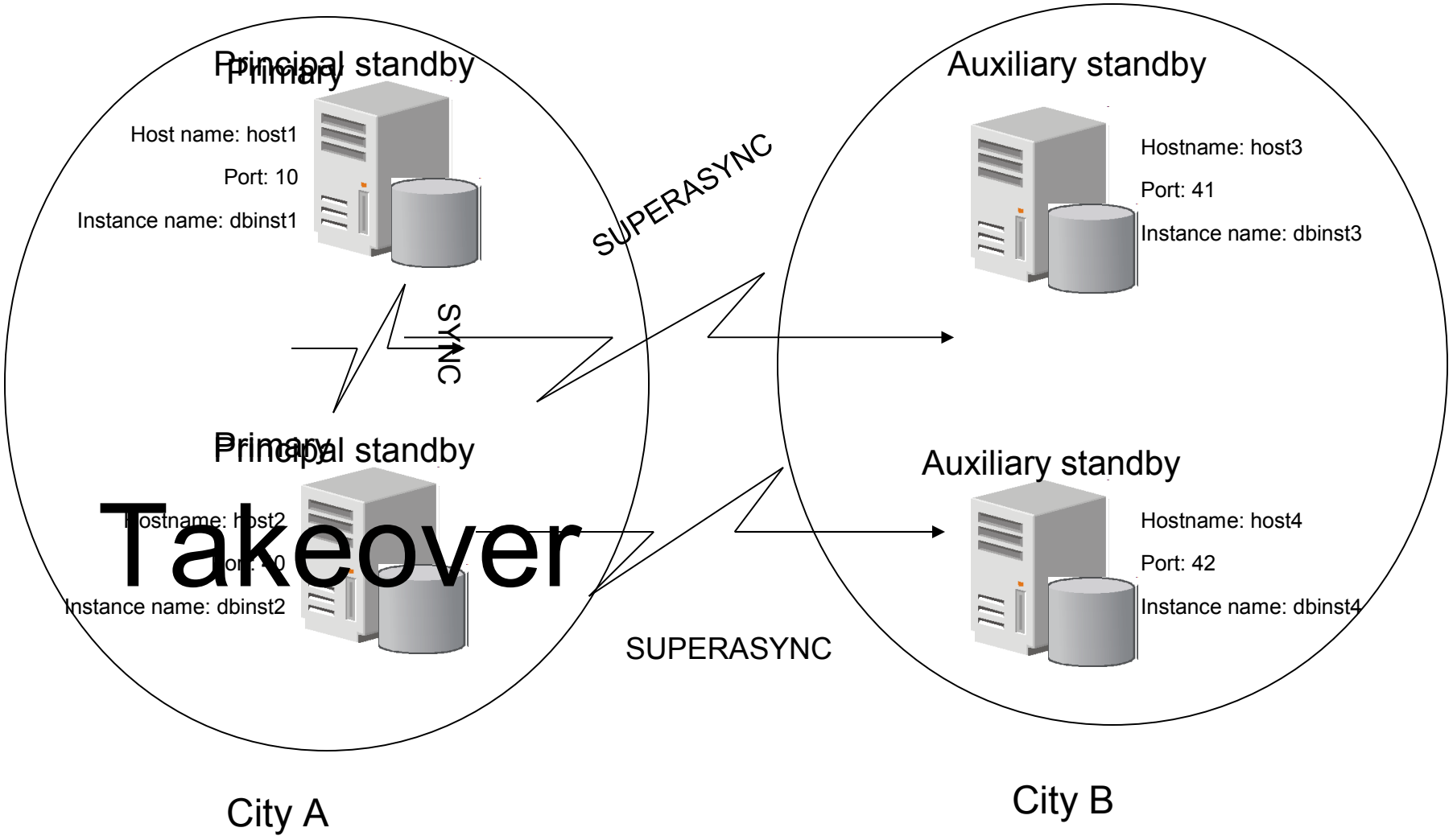
HADR Multiple Standby Example



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	host2:20 | host1:10 | host4:42	host2:40 | host1:10 | host3:41
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	superasync	superasync
Effective Hadr_syncmode	N/A	sync	superasync	superasync

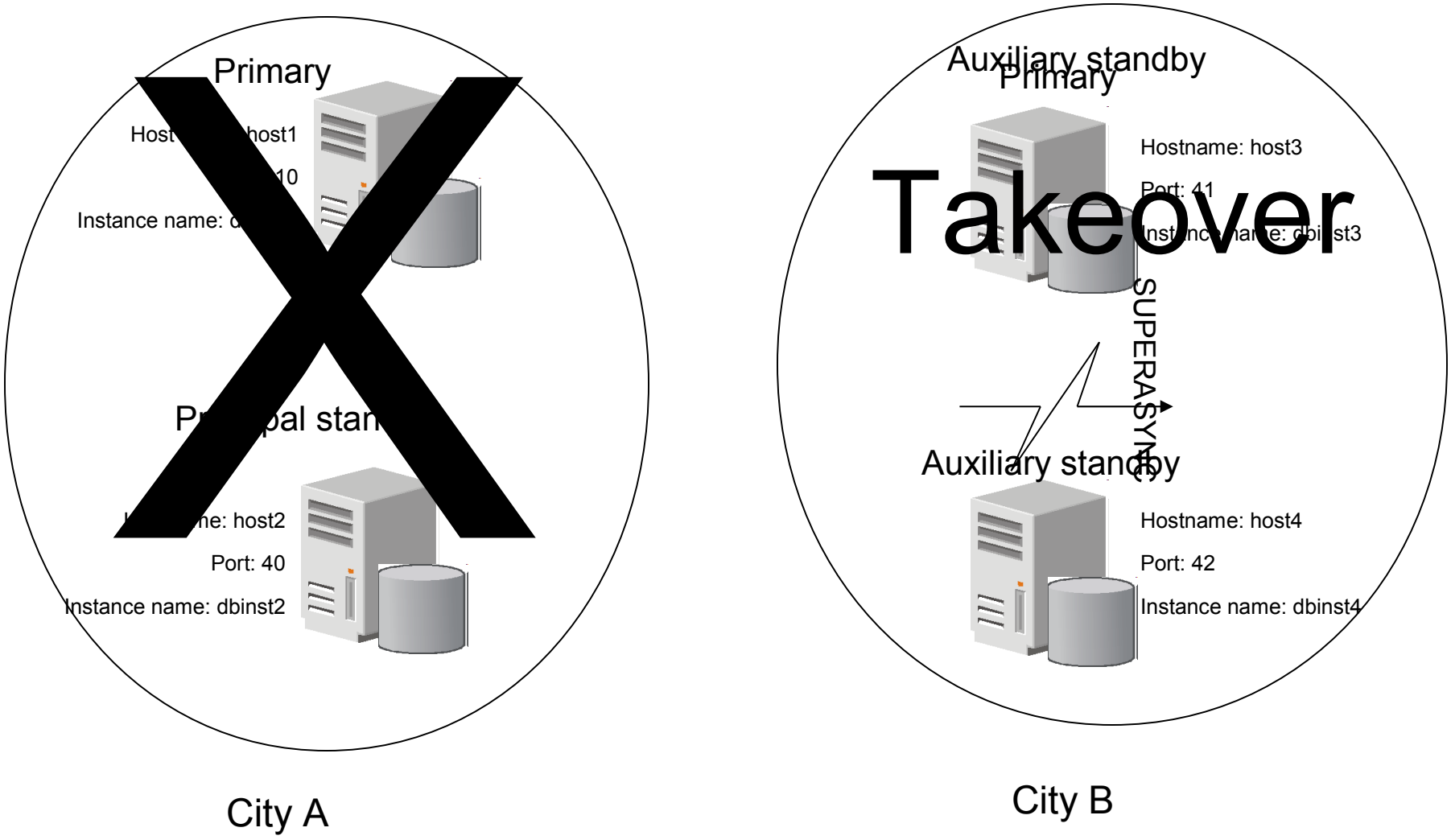
HADR Multiple Standby Role Reversal Example



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	host2:40 | host1:10 | host3:41
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	superasync	superasync
Effective Hadr_syncmode	sync	N/A	superasync	superasync

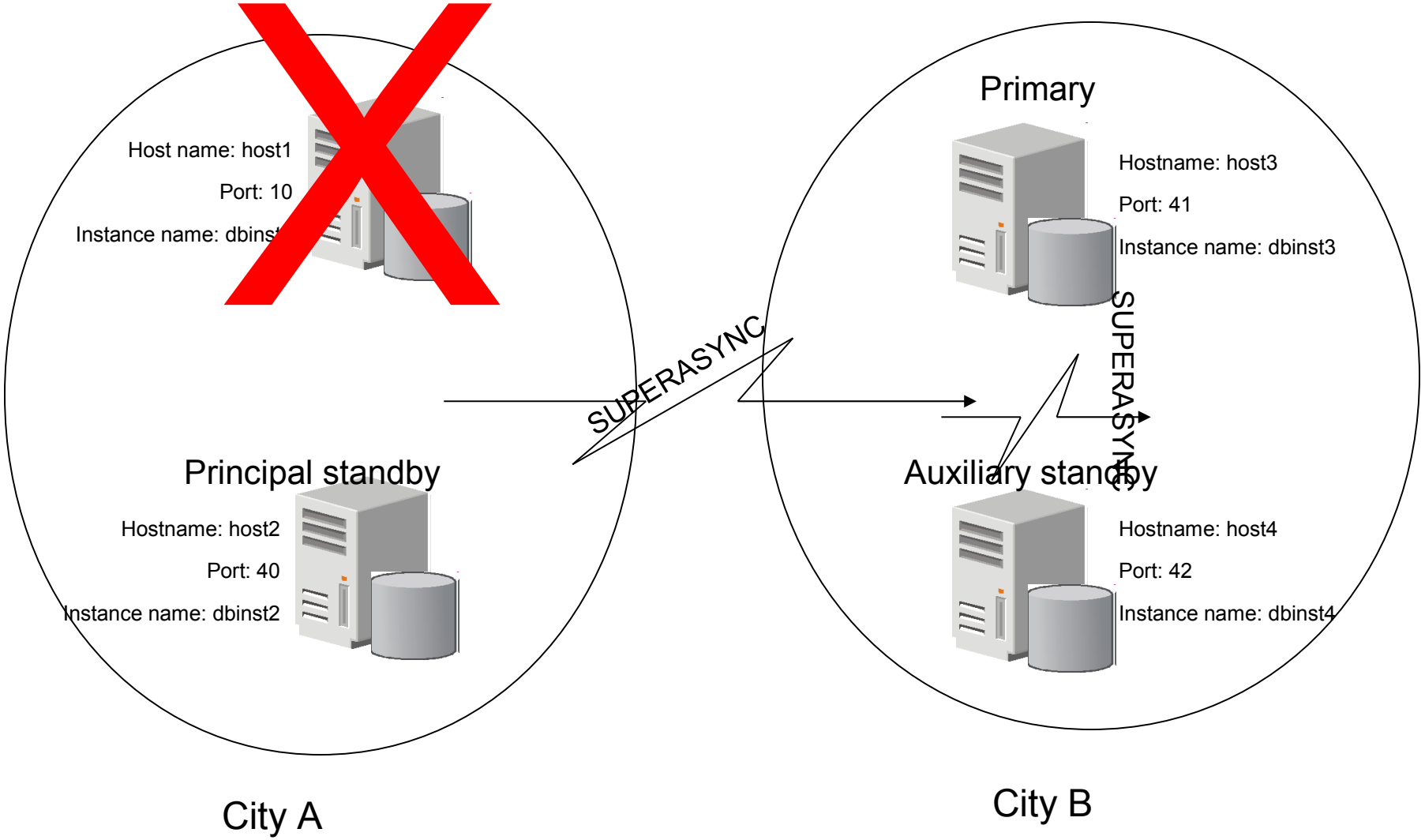
HADR Multiple Standby Forced Takeover Example



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	host2:40 | host1:10 | host3:41
Hadr_remote_host	host2	host1	host2	host3
Hadr_remote_svc	40	10	40	41
Hadr_remote_inst	dbinst	dbins	dbinst2	dbinst3
Hadr_local_host	host1	host2	host3	host4
Hadr_local_svc	10	40	41	42
Operational Hadr_syncmode	sync	sync	superasync	superasync
Effective Hadr_syncmode	N/A	sync	n/a	superasync

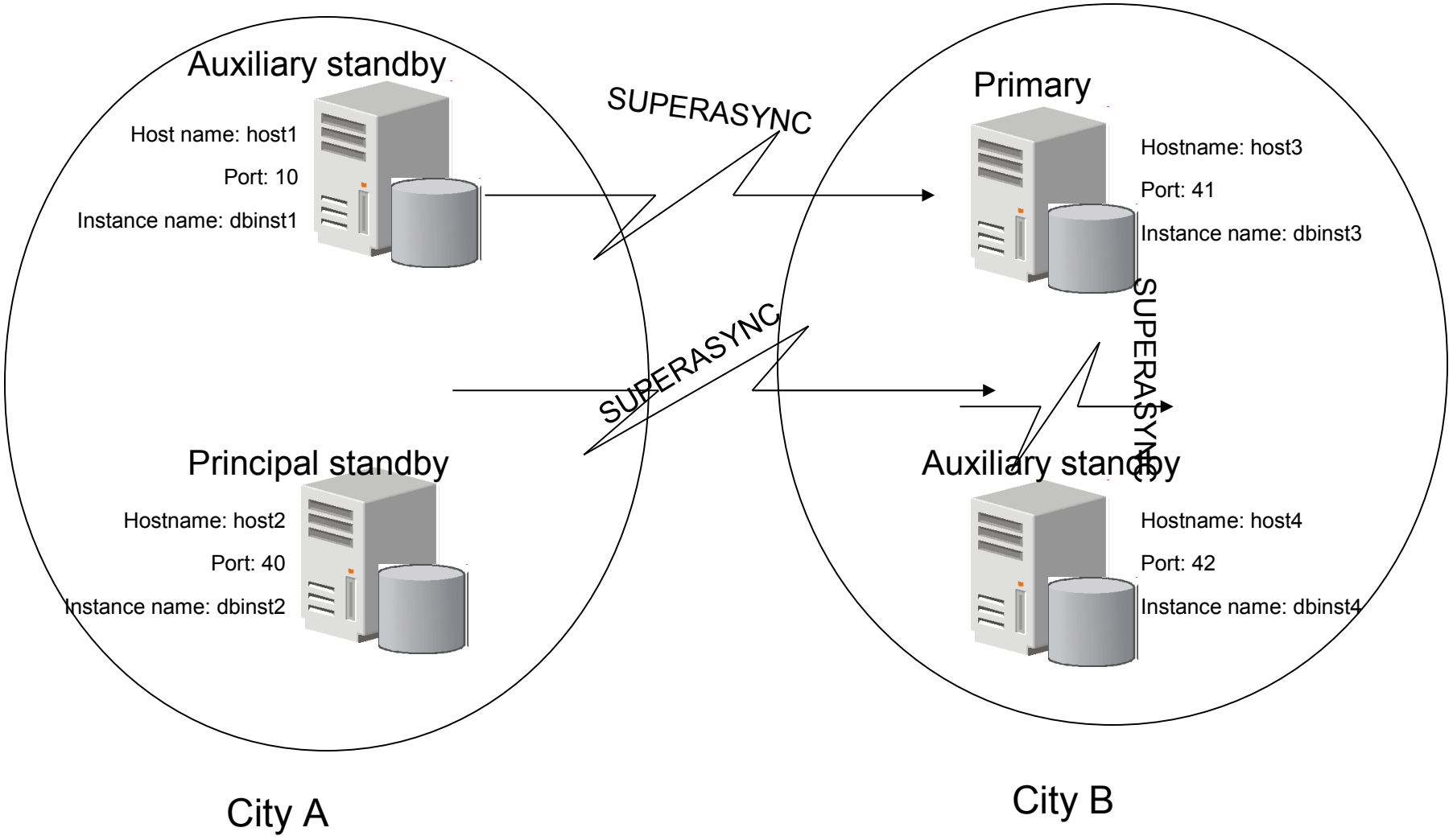
HADR Multiple Standby Example



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	host2:40 | host1:10 | host3:41
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	superasync	superasync
Effective Hadr_syncmode	N/A	superasync	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	host2:40 | host1:10 | host3:41
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	superasync	superasync
Effective Hadr_syncmode	superasync	superasync	n/a	superasync

Software upgrades in a Multi Standby Environment

The procedure is essentially the same as with single standby mode, except you should perform the upgrade on one database at a time and starting with an auxiliary standby. For example, consider the following HADR setup:\

- host1 is the primary
- host2 is the principal standby
- host 3 is the auxiliary standby

For this setup, perform the rolling upgrade or update according to the following sequence:

1. Deactivate host3, make the required changes, activate host3, and start HADR on host3 (as a standby).
2. After host3 is caught up in log replay, deactivate host2, make the required changes, activate host2, and start HADR on host2 (as a standby).
3. After host2 is caught up in log replay and in peer state with host1, issue a takeover on host2.
4. Deactivate host1, make the required changes, activate host1, and start HADR on host1 (as a standby).
5. After host1 is in peer state with host 2, issue a takeover on host1 so that it becomes the primary again and host2 becomes the principal standby again.

HADR Configuration Parameters

hadr_db_role
This parameter indicates the current role of a database; Valid values are: STANDARD, PRIMARY, or STANDBY.

hadr_local_host
specifies the local host for high availability disaster recovery (HADR) TCP communication.

hadr_local_svc
specifies the TCP service name or port number for which the local high availability disaster recovery (HADR) process accepts connections.

hadr_peer_window
When you set hadr_peer_window to a non-zero time value, then a HADR primary-standby database pair continues to behave as though still in peer state, for the configured amount of time, if the primary database loses connection with the standby database. This helps ensure data consistency

hadr_remote_host
specifies the TCP/IP host name or IP address of the remote high availability disaster recovery (HADR) database server

HADR Configuration Parameters

hadr_remote_inst
specifies the instance name of the remote server. High availability disaster recovery (HADR) also checks whether a remote database requesting a connection belongs to the declared remote instance.

hadr_remote_svc
specifies the TCP service name or port number that will be used by the remote high availability disaster recovery (HADR) database server.

hadr_replay_delay
specifies the number of seconds that must pass from the time that a transaction is committed on the primary database to the time that the transaction is committed on the standby database

hadr_spool_limit
determines the maximum amount of log data that is allowed to be spooled to disk on HADR standby

hadr_syncmode
specifies the synchronization mode, which determines how primary log writes are synchronized with the standby when the systems are in peer state

HADR Configuration Parameters

hadr_target_list

This parameter, which enables HADR to run in multiple standby mode, specifies a list of up to three target *host*:*port* pairs that act as HADR standby databases.

hadr_timeout

specifies the time (in seconds) that the high availability disaster recovery (HADR) process waits before considering a communication attempt to have failed

Blocknonlogged

specifies whether the database manager will allow tables to have the NOT LOGGED or NOT LOGGED INITIALLY attributes activated.

Logindexbuild

specifies whether index creation, recreation, or reorganization operations are to be logged so that indexes can be reconstructed during DB2 rollforward operations or high availability disaster recovery (HADR) log replay procedures.

HADR Configuration Parameters Updates

- you need only stop and start HADR for updates to some HADR configuration parameters for the primary database to take effect. You do not have to deactivate and reactivate the database. This dynamic capability affects only the primary database because stopping HADR deactivates any standby database.
- The affected configuration parameters are as follows:
  - **hadr_local_host**
  - **hadr_local_svc**
  - **hadr_peer_window**
  - **hadr_remote_host**
  - **hadr_remote_inst**
  - **hadr_remote_svc**
  - **hadr_replay_delay**
  - **hadr_spool_limit**
  - **hadr_syncmode**
  - **hadr_target_list**
  - **hadr_timeout**

HADR Monitoring Changes

There are two preferred ways of monitoring HADR:

- The db2pd command
- The MON_GET_HADR table function
- From the primary = information about the primary and all standbys
- From a standby = information about that standby and the primary

You can also use the following methods, but have been deprecated and may be removed in a future release:

The **GET SNAPSHOT FOR DATABASE command**

- The db2GetSnapshot API
- The SNAPHADR administrative view
- The SNAP_GET_HADR table function
- Other snapshot administrative views and table functions

DB2 PD Changes – one entry for each primary-standby pair

db2pd -db HADRDB -hadr

Database Member 0 -- Database HADRDB -- Active -- Up 0 days 00:23:17 -- Date 06/08/2011
HADR_ROLE = PRIMARY
REPLAY_TYPE = PHYSICAL
HADR_SYNCMODE = SYNC
STANDBY_ID = 1
LOG_STREAM_ID = 0
HADR_STATE = PEER
PRIMARY_MEMBER_HOST = hostP.ibm.com
PRIMARY_INSTANCE = db2inst
PRIMARY_MEMBER = 0
STANDBY_MEMBER_HOST = hostS1.ibm.com
STANDBY_INSTANCE = db2inst
STANDBY_MEMBER = 0
HADR_CONNECT_STATUS = CONNECTED
HADR_CONNECT_STATUS_TIME = 06/08/2011 13:38:10.199479 (1307565490)
HEARTBEAT_INTERVAL(seconds) = 25
HADR_TIMEOUT(seconds) = 100
TIME_SINCE_LAST_RECV(seconds) = 3

DB2 PD Changes – one entry for each primary-standby pair con't

LOG_HADR_WAIT_CUR(seconds) = 0.000
LOG_HADR_WAIT_RECENT_AVG(seconds) = 0.006298
LOG_HADR_WAIT_ACCUMULATED(seconds) = 0.516
LOG_HADR_WAIT_COUNT = 82
SOCK_SEND_BUF_REQUESTED,ACTUAL(bytes) = 0, 50772
SOCK_RECV_BUF_REQUESTED,ACTUAL(bytes) = 0, 87616
PRIMARY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
STANDBY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
HADR_LOG_GAP(bytes) = 0
STANDBY_REPLAY_LOG_FILE,PAGE,POS = S0000009.LOG, 1, 49262315
STANDBY_RECV_REPLAY_GAP(bytes) = 0
PRIMARY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
STANDBY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
STANDBY_REPLAY_LOG_TIME = 06/08/2011 13:49:19.000000 (1307566159)
STANDBY_RECV_BUF_SIZE(pages) = 16
STANDBY_RECV_BUF_PERCENT = 0
STANDBY_SPOOL_LIMIT(pages) = 0
PEER_WINDOW(seconds) = 0
READS_ON_STANDBY_ENABLED = Y
STANDBY_REPLAY_ONLY_WINDOW_ACTIVE = N

Monitoring Standby Servers

db2pd command

This command retrieves information from the DB2 memory sets. You can issue this command from either a primary database or a standby database. If you are using multiple standby mode and you issue this command from a standby, it does not return any information about the other standbys.

MON_GET_HADR table function

If you want to issue the MON_GET_HADR function against a standby database, be aware of the following points:

- You must enable reads on standby on the standby.
- Even if your HADR setup is in multiple standby mode, the table function does not return any information about any other standbys.

High Availability Positioning

Bronze		Silver		Gold
PureScale	**HADR**	**DB2 Integrated Clustering**		
Transactions on surviving nodes are not impacted	Provides the fastest failover, typically less than 30 seconds	Hot/Warm (optional hot with read on standby option	Online scaleout capability to address workload spikes	Online failover

Disaster Recovery Positioning

	Bronze	Silver	Gold
Ability to bring target DB online	**HADR** Hot/Warm (optional hot with read on standby option Provides very fast failover, typically less than 1 minute Easy to setup with turnkey availability Minimal licensing on the standby server Basic reporting (DB2 v9.7 FP1) requires full standby licensing Zero data loss option	**Q Replication** •Hot / Hot •Unrestricted access to the target DB •Supports different versions of DB2 on source and target •Supports sub setting of data •Multiple target support •Online failover •Can replicate to a different topology •Async only •Complex to setup •No DDL support	**Log Shipping** Target DB always at least 1 full log file behind

Dale McInnis
IBM Canada Ltd.
dmcinnis@ca.ibm.com

Session H03
HADR Update: Multiple Standby Support