IDUG VIRTUAL

2021 EMEA **Db2** Tech Conference

Migrating DB2 from AIX to X86 Linux

Dale McInnis – dmcinnis@ca.ibm.com

IBM Canada Ltd

#IDUGDb2

Platform: LUW

Agenda

- Database Migration Planning
- Resource Sizing
- Cluster Manager Impact
- Database Migration tools
  - Replication Challenges
- Db2 P-Series optimization
- Db2 RHEL optimization
- Guidelines

Migration of Db2 database from P-Series to X86

- Functionality of the database in AIX is identical to that running on RHEL X86.
- It is the same code, just recompiled with a different compiler.
- All features and functionality are ***guaranteed*** to be the same.
- Due to differences in endianness a traditional Db2 Backup and Restore will not work
  - The data must be “byte reversed” and reloaded
- This is a perfect time to re-evaluate your physical database layout
  - E.g. move to automatic storage, separate LOBs into separate tablespaces, separate indexes to separate tablespaces with dedicated bufferpools, …
  - Now is the time to modernize your physical database layout
    - E.g. eliminate SMS and DMS tablespaces
- Planning is the key to success

Recommended steps to follow

Thorough planning is the cornerstone to a successful migration.

The project plan must include the business objective as well as any constraint, e.g. length of outage window, tools available, skill available, fall back plan, …

The following steps should be considered when planning the migration:

- Business Objective & Technical Objectives
- Technical Constraints
- Identify Potential Techniques
- Obtain Appropriate Assistance and Advice
- Analyze Each Potential Technique
- Plan for the Network & Client SW changes required
- Develop Cost Budget & Planning estimates
- Evaluate Alternatives
- Test Solutions (if needed)
- Select Solution
- Implement – Strong Project Management required
- Evaluate Implementation

Minimizing downtime

- Minimizing the downtime of a migration is typically the #1 requirement.
  - There are several techniques that we will discuss, each with a differing amount of outage time required.
  - Do not forget to account for testing/validation on the new platform before turning the system over for production
- Using a logical replication product, such as Q Repl / CDC will provide the least amount of downtime. However certain skills and product licenses will be required
- An Unload / Reload of all data will take the longest but does not require any additional tooling
  - High Performance Unload is a licensed product that would greatly reduce the unload time

Migration Strategies Comparison

	db2move	Load from cursor	Export / Load	HPU	Logical Replication
Outage	Medium	Varies	Long	Medium	Short
Complexity	Low	Medium	Medium	Low	High
Additional Cost	No	No	No	HPU license	Depends on License / Release
Failback capability	Extract new data	Extract new data	Extract new data	Extract new data	Reverse replication
Database size	Good for small databases	Good for small databases	Good for small databases	Good for medium size databases	Good for all sizes of databases

Agenda

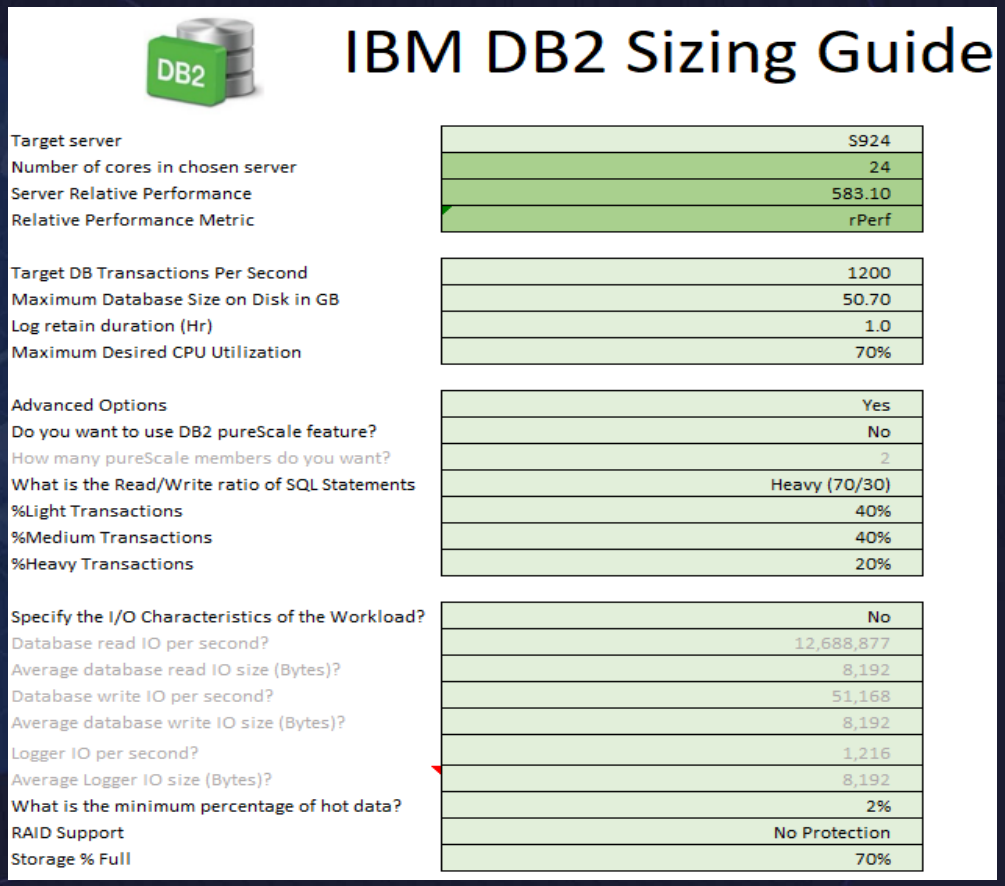
- Database Migration Planning
- Resource Sizing
- Cluster Manager Impact
- Database Migration tools
  - Replication Challenges
- Db2 P-Series optimization
- Db2 RHEL optimization
- Guidelines

Sizing – it is not as simple as 1 to 1 ☹

- Ensure that you have sufficient # of cores and memory in the new environment
- 1 P-Series core != 1 X86 core
- Hypervisors add overhead (VMWare)
- Recommend to resize each environment

TPS 1000										
					# cores	Base2017	Base 2017 per core	Ratio		
Intel X86 chipset	Intel(R) Xeon(R) Gold 6248 CPU 2,5 GHz				20	120	6	1.479167		
Power Chipset	POWER9 IBM Power S924 (3.4 - 3.9 GHz)				24	213	8.875			
	# of members	# of CFs	# cores / member - Pseries	# cores / CF - Pseries	Total # of cores - Pseries	# cores / member - X86	# cores / CF - X86	Total # of cores - X86	Memory per member	Memory per CF
PureScale	2	2	3	1	8	5	2	14	250 GB	200 GB
nonPureScale	1	0	5	0	5	7	0	7	500 GB	

How to size an environment



IBM DB2 Sizing Guide

Target server	S924
Number of cores in chosen server	24
Server Relative Performance	583.10
Relative Performance Metric	rPerf
Target DB Transactions Per Second	1200
Maximum Database Size on Disk in GB	50.70
Log retain duration (Hr)	1.0
Maximum Desired CPU Utilization	70%
Advanced Options	Yes
Do you want to use DB2 pureScale feature?	No
How many pureScale members do you want?	2
What is the Read/Write ratio of SQL Statements	Heavy (70/30)
%Light Transactions	40%
%Medium Transactions	40%
%Heavy Transactions	20%
Specify the I/O Characteristics of the Workload?	No
Database read IO per second?	12,688,877
Average database read IO size (Bytes)?	8,192
Database write IO per second?	51,168
Average database write IO size (Bytes)?	8,192
Logger IO per second?	1,216
Average Logger IO size (Bytes)?	8,192
What is the minimum percentage of hot data?	2%
RAID Support	No Protection
Storage % Full	70%

	DB2 Power	DB2 x86	pS Power	pS x86	CF Power	CF x86
rperf or SPEC units required	141.42		101.12		23.57	
Utilization for Bare Metal	24.25%		17.34%		4.04%	
Cores Needed to Run Workload	5.9		4.2		1.0	
Cores Needed to Run Workload wth Utilization Target	8.4		6.0		1.4	
Memory (GB)	144	108	211	158	190	143
Size Default Partition (GB)	14	14	14	14	14	14
Size DB (GB)	50.70	50.70	50.70	50.70		
Size Log (GB)	75.21	75.21	75.21	75.21		
Size DB Total (GB)	180.05	125.91	125.91	125.91		
Disk Size with RAID (GB)	257.22	179.87	179.87	179.87		
Data IOPS	10,953.47	10,953.47	10,953.47	10,953.47		
Log IOPS	2,738.37	2,738.37	2,738.37	2,738.37		
Total IOPS	13,691.83	13,691.83	13,691.83	13,691.83		
Primary CF HCA					4	4
Secondary CF HCA					4	4
HCA Memory						
PureScale Members			2	2		

Agenda

- Database Migration Planning
- Resource Sizing
- Cluster Manager Impact
- Database Migration tools
  - Replication Challenges
- Db2 P-Series optimization
- Db2 RHEL optimization
- Guidelines

Cluster Managers

- Most popular cluster manager on AIX is PowerHA, aka HACMP
- This is an AIX only solution which is tightly tied to the operating system
- Db2 ships with a cluster manager for free
  - For both AIX and Linux Db2 includes Tivoli System Automation for Multi-Platforms (TSA/MP)
  - Supports 3 topologies
    - Shared Disk Cluster
    - HADR Automation
    - DPF automation
- Db2 can be configured with any external cluster manager
  - E.g. Veritas Cluster Server (VCS), RHEL HA, PowerHA(HACMP)
  - This will be done outside of the database, no sample scripts will be provided.

Cluster Managers Con't

- Choice of a tiebreaker device is critical to success
- TSA supports 3 tie breakers (one per TSA cluster)
  - Network (TCP/IP)
  - Shared Disk
  - 3rd Node
- Recommendation is to always use 3rd node tie breaker
  - Prevents possibility of a split brain
- PaceMaker only supports 3rd node tie breaker
  - Tiebreaker can be shared across clusters and does not have to be the same endianness

Db2 Cluster Manager Direction

- Db2 V 11.5.5.0 includes a replacement for TSA, specifically Pacemaker and Corosync on Linux only.
- All future enhancements will be made to the Pacemaker automation tool
- This technology will also be used in both DPF and PureScale in a future release

Agenda

- Database Migration Planning
- Resource Sizing
- Cluster Manager Impact
- Database Migration tools
  - Replication Challenges
- Db2 P-Series optimization
- Db2 RHEL optimization
- Guidelines

Migration of Db2 from P-Series AIX to X86 Linux

- Due to differences in endianness on these chips, a Db2 Backup and Restore will not work
  - AIX = Big “E” whereas RHEL on X86 = Little “E”
- The data must be unloaded, converted to the new format then reloaded on the target database
- Step 1: Extract the DDL from the source DB
  - See db2look section
- Step 2: Create an empty database on the target server
  - This is a great time to re-evaluate your table/index to tablespace assignments
  - Ensure that the new target database is created with AUTOMATIC STORAGE
    - Both DMS and SMS tablespaces have been deprecated
- Step 3: Create the tablespaces and bufferpools, recreate all of the objects contained in the db2look.out file
- Step 4: Copy the data into the new DB
  - Several approaches are possible

Db2look – DDL extraction utility

Generate a Data Definition Language (DDL) file using the **db2look** command. The **db2look** command generates the DDL statements by object type. Note that this command ignores all objects under the SYSTOOLS schema except user-defined functions and stored procedures. Using the **db2look** utility, we were able to generate DDL statements for schemas, statements for table, statements for user-defined functions and procedures, statements for foreign keys, and statements for trigger queries in SQL files.

```
#db2look -d <DBname>-a -e -m -l -x -f -o DBname.sql
```

Command parameters:

-d <DBname>: Is the alias name of the database that is to be moved.
-a: Generates DDL statements for objects that were created by any user, including inoperative objects.
-e: Extracts DDL statements.
-m: Generates the UPDATE statements that are required to replicate the statistics on tables, statistical views, columns, and indexes.
-l: Generates DDL statements for user-defined table spaces, user-defined database partition groups, and user-defined buffer pools.
-x: Generates authorization DDL statements such as GRANT statements.
-f: Extracts the configuration parameters and registry variables that affect the query optimizer.
-o: Writes the output to the DBname.sql file

Execution of db2look

```
db2hadev:/home/dmcinnis/CrossPlatformMigration/db2look>  db2look -d sample -a -e -m -l -x -f -o sample.sql
-- Generate statistics for all creators
-- Creating DDL for table(s)
-- Running db2look in mimic mode
-- Output is sent to file: sample.sql
db2hadev:/home/dmcinnis/CrossPlatformMigration/db2look> ls
sample.sql
db2hadev:/home/dmcinnis/CrossPlatformMigration/db2look>
```

db2look con’t

- Separate the foreign key and the triggers from the DBname.sql file.
- If you execute the *DBname.sql* without separating the foreign keys, we may get errors due to missing primary key during import. Imported rows will be rejected due to dependencies between tables. You can see the error messages under **tabXX.msg**.
- From the above generated SQL *DBname.sql* file, search for triggers, and separate the triggers into a new file named *DBname_Trigger.sql*. You will get error (*“SQL3550W The field value in row “noX.” and column “no.X” is not NULL, but the target column has been defined as “GENERATED ALWAYS”*) if you run the DDL file (for example, *dbname.sql*) without separating triggers.

```
db2hadev:/home/dmcinnis/CrossPlatformMigration/db2look> ls
sample.sql          sample_Foreign.sql  sample_Trigger.sql
```

Create tablespaces + bufferpools in target DB

- This is the perfect time to re-evaluate your table to tablespace and bufferpool assignments.
- If you are using LOBs this is the time to separate them into a separate tablespace.
  - LOB tablespaces should be configured with FILE SYSTEM CACHING ENABLED
  - Particularly important in cloud environments
    - One customer experienced 4X slowdown when then moved to Azure on all LOB inserts
    - Selection of storage devices should be driven by IOPS requirement
- Consider putting indexes into a separate bufferpool to ensure the pages stay in memory longer

Create tablespaces + bufferpools in target DB

- If you have very active tables you may want to consider putting them into separate tablespaces with separate bufferpools.
  - Do not go too far – one customer has 40+ bufferpools with an avg hit ratio of 65%
- Db2 native backup and recovery granularity is at the tablespace level, so consider placing tables that are dependent upon each other in the same tablespace so they can be recovered as a group
- Finally get eliminating the use of SMS/DMS containers

Migration Strategies Comparison

	db2move	Load from cursor	HPU	Logical Replication
Outage	Medium	varies	Medium	Short
Complexity	Low	Medium	Low	High
Additional Cost	No	No	HPU license	Depends on License
Failback capability		Extract new data	Extract new data	Reverse replication
Database size	Good for small databases	Good for small databases, few tables	Good for medium size databases	Good for all sizes of databases

Migration Strategies

- db2move
- Load from cursor
  - Does not require any additional storage
- Unload (HPU) / Load
  - Chargeable feature (HPU)
  - Extremely fast
- Logical Replication (Q Repl)
  - Requires source and target to be V 11.5.4.0 or above for Columnar support

Exporting the data: using db2move

Export all data from the source server (AIX) using the db2move utility to retrieve a list of all user tables in a database from the system catalog and export these tables to the PC/IXF format. Export all the tables that meet the filtering criteria according to the option specified. If you do not specify an option, then all tables are exported.

Internal staging information is stored in the *db2move.lst* file.

db2move <DBname> Export

- **EXPORT.out:** Shows the summarized result of the EXPORT action.
- **db2move.lst:** Holds the list of original table names, their corresponding PC/IXF file names (tabnnn.ixf), and message file names (tabnnn.msg). This list, the exported PC/IXF files, and large object (LOB) files (tabnnnc.yyy) are used as input to the db2move IMPORT or LOAD action.
- **tabnnn.ixf:** Is the exported PC/IXF file of a specific table.
- **tabnnn.msg:** Is the export message file of the corresponding table.
- **tabnnnc.yyy:** The exported LOB files of a specific table.nnn is the table number. 'c' is a letter of the alphabet. 'yyy' is a number ranging from 001 to 999. These files are created only if the table being exported contains LOB data. If created, these LOB files are placed in the path of the LOB directories. There is a total of 26,000 possible names for the LOB files.
- **System.msg:** Is the message file that contains system messages for creating or deleting a file or directory commands. This is only used if the action is EXPORT, and a LOB path is specified.

Executing db2move

- db2move sample export

```
db2hadev:/home/dmcinnis/CrossPlatformMigration> ls
EXPORT.out      tab13.msg       tab18.msg       tab23.ixf       tab5.ixf
db2move.lst     tab14.ixf       tab19.ixf       tab23.msg       tab5.msg
tab1.ixf        tab14.msg       tab19.msg       tab24.ixf       tab6.ixf
tab1.msg        tab15.ixf       tab2.ixf        tab24.msg       tab6.msg
tab10.ixf       tab15.msg       tab2.msg        tab25.ixf       tab7.ixf
tab10.msg       tab16.ixf       tab20.ixf       tab25.msg       tab7.msg
tab10a.001.lob  tab16.msg       tab20.msg       tab25a.001.xml  tab8.ixf
tab11.ixf       tab16a.001.lob  tab21.ixf       tab3.ixf        tab8.msg
tab11.msg       tab17.ixf       tab21.msg       tab3.msg        tab9.ixf
tab12.ixf       tab17.msg       tab21a.001.xml  tab4.ixf        tab9.msg
tab12.msg       tab17a.001.xml  tab22.ixf       tab4.msg        tab9a.001.lob
tab13.ixf       tab18.ixf       tab22.msg       tab4a.001.xml
db2hadev:/home/dmcinnis/CrossPlatformMigration>
```

Exporting the data: using db2move con’t

Copy all the files, that is, the SQL files (*DBname.sql, DBname_Foreignkey.sql, “DBname_Trigger.sql”*) and the exported data, to any directory on the RHEL destination system

Connect to the database and capture the output of the following command to grant a similar level permission to the connection ID.

```
“db2 "select char(grantee,8) as grantee, char(granteetype,1) as type, \ char(dbadmauth,1) as
dbadm, char(createtabauth,1) as createtab, \ char(bindaddauth,1) as bindadd, char(connectauth,1)
as connect, \ char(nofenceauth,1) as nofence, char(implschemaauth,1) as implschema, \
char(loadauth,1) as load, char(externalroutineauth,1) as extroutine, \ char(quiesceconnectauth,1)
as quiesceconn, \ char(libraryadmauth,1) as libadm, char(securityadmauth,1) \ as securityadm from
syscat.dbauth order by grantee"
```

Importing the data – using db2move

1.Create a database using automatic storage stripped across multiple filesystems.

```
db2 'CREATE DATABASE <DBname> AUTOMATIC STORAGE YES on PATH /db2fs1, /db2fs2, /db2fs3 DBPATH /db2fs4';
Db2 'CONNECT TO <DBname>';
```

2.Run a SQL query that was generated on the source server that is, the *DBname.sql* script to create a tables and indexes. Note: Do not run *DBname_Foreignkey.sql* and *Dbname_Trigger.sql* before the import to avoid errors.

```
db2 -tvf DBname.sql
```

3.Grant similar level permissions to the connection ID and instance captured earlier.

4.Import all the tables listed in the db2move.lst internal staging file. Use the -io option for import-specific actions. It loads the actual data into the blank tables which are created using the db2 -tvf DBname.sql command (in step 2). While importing, the data utility shows the progress (table wise as shown below):

```
db2move <DBname> import
```

The following files act as the input while importing data:

db2move.lst: An output file from the EXPORT action.
tabnnn.ixf: An output file from the EXPORT action.
tabnnnc.yyy: An output file from the EXPORT action.

The following files are generated when using the Import command:

IMPORT.out: The summarized result of the IMPORT action.
tabnnn.msg: The import message file of the corresponding table.

Importing the data – using db2move

5.Run the *DBname_Foreignkey.sql* and *DBname_Trigger.sql* files separately to re-establish the foreign key relationships and triggers

db2 -tvf DBname_Foreignkey.sql db2 -tvf DBname_Trigger.sql

6. Issue a runstats on all tables to ensure statistics are up to date

7.Before enabling a connection between the database and the application, take a backup of the database

Db2move

- See article https://developer.ibm.com/tutorials/migrating-db2-from-little-endian-linux-to-big-endian-aix/

High Performance Unload (HPU)

•Create an empty database, NEWDB, with the new instance, newinst, on node2. NEWDB must be identical to the OLDDB database on node1. All of the tables must have the same names and the same data types and sizes. You can use any tool to create the database (for example, the db2look tool).

•Install Optim High Performance Unload on both node1 and node2.

•Create a Optim High Performance Unload control file and specify the migration options. For example (scenario2.ctr):

```
GLOBAL CONNECT TO OLDDB
UMASK "022"
;
MIGRATE DATABASE
TARGET ENVIRONMENT (INSTANCE "newinst" ON “node2" IN NEWDB)
WORKING IN ("user_dir_on_new_system")
TARGET KEYS (CURRENT PARTS(ALL))
FORMAT MIGRATION
;
```

See: https://www.ibm.com/docs/en/iohpufdflu-and-w/6.1?topic=SSC6UE_6.1.0/inmutsk_scenmigration.html

•Save the control file in a directory on the old system. For example, */home/db2inst1/*HPU_ControlFiles/scenario2.ctr.

•Create credentials of local type for the user concerned on the target environment.

•Open the command line and run the Optim High Performance Unload command with the control file:

db2hpu -f /home/db2inst1/HPU_ControlFiles/scenario2.ctr

Load from cursor - Federation

Approach 1 uses a federated access to the source database to copy table contents.
This approach requires that the target database is configured as federated database. Therefore the parameter FEDERATED of the corresponding DB2 instance has to be set to YES (UPDATE DBM CFG).
After changing the DBM CFG parameter FEDERATED, the DB2 instance has to be restarted (db2stop/db2start).

Next extract the DDL from the source database using the db2look utility
or you can just generate a list of tables

Execute the DDL generate by the db2look utility

Create entries in the node and database directory to access the remote database

Load from cursor - Federation

There are essentially three steps once the source database is cataloged on the server of the target database:

1.On target server, connect to target database

2.Declare cursor against the source database: `db2 “DECLARE C1 CURSOR select * from schema.table with ur”`

3.Load from cursor: `db2 “LOAD FROM C1 of CURSOR MESSAGES test_load.msg replace into schema.table nonrecoverable”`

Setting up federation – on RHEL X86

- db2 update dbm cfg using federated yes
- db2stop
- db2start
- db2 catalog TCPIP NODE db2hadev REMOTE db2hadev SERVER 50000
- db2 catalog database sample as aixsampl at node db2hadev authentication server
- db2 create db sample
- db2 connect to sample
- db2 create wrapper DRDA
- db2 " create server db2hadev type DB2/UDB Version 11 WRAPPER DRDA authorization \"dmcinnis\" PASSWORD \“mypassword\" options (ADD DBNAME 'aixsampl')"
- db2 create user mapping for dmcinnis server db2hadev options (REMOTE_AUTHID 'dmcinnis', REMOTE_PASSWORD ‘mypassword')
- create nickname sourceschema.tablename for db2hadev.sourceschema.sourcetablename #REPEAT FOR EACH TABLE you want to load

Setting up federation – on RHEL X86

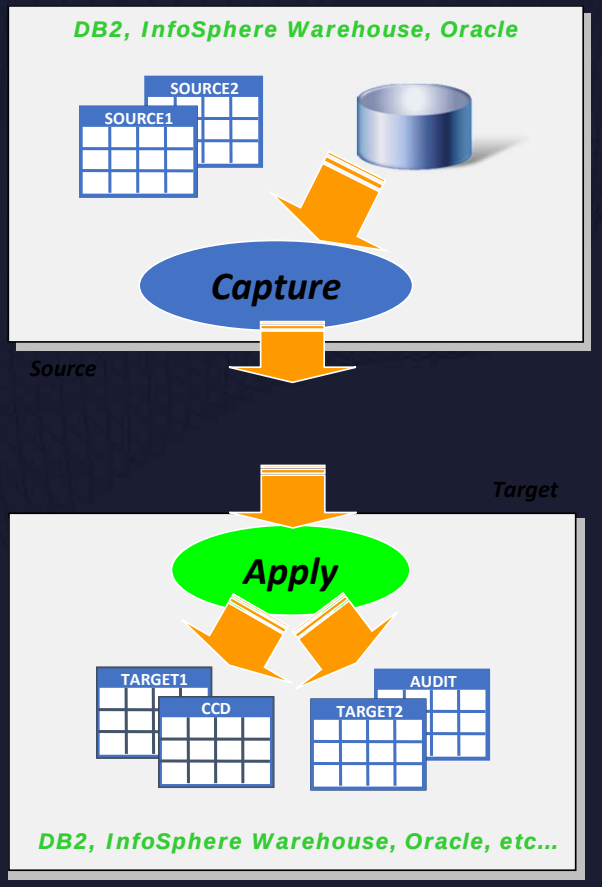
- drop table targetschema.tablename;
- create table targetschema.tablename like sourceschema.tablename ;
- declare C1 cursor for select * from sourceschema.tablename with ur;
- load from C1 of cursor messages test_load.msg insert into targetschema.tablename nonrecoverable;
- select count(*) from targetschema.tablename ;
- select count(*) from sourceschema.tablename ;
- terminate;

Logical replication

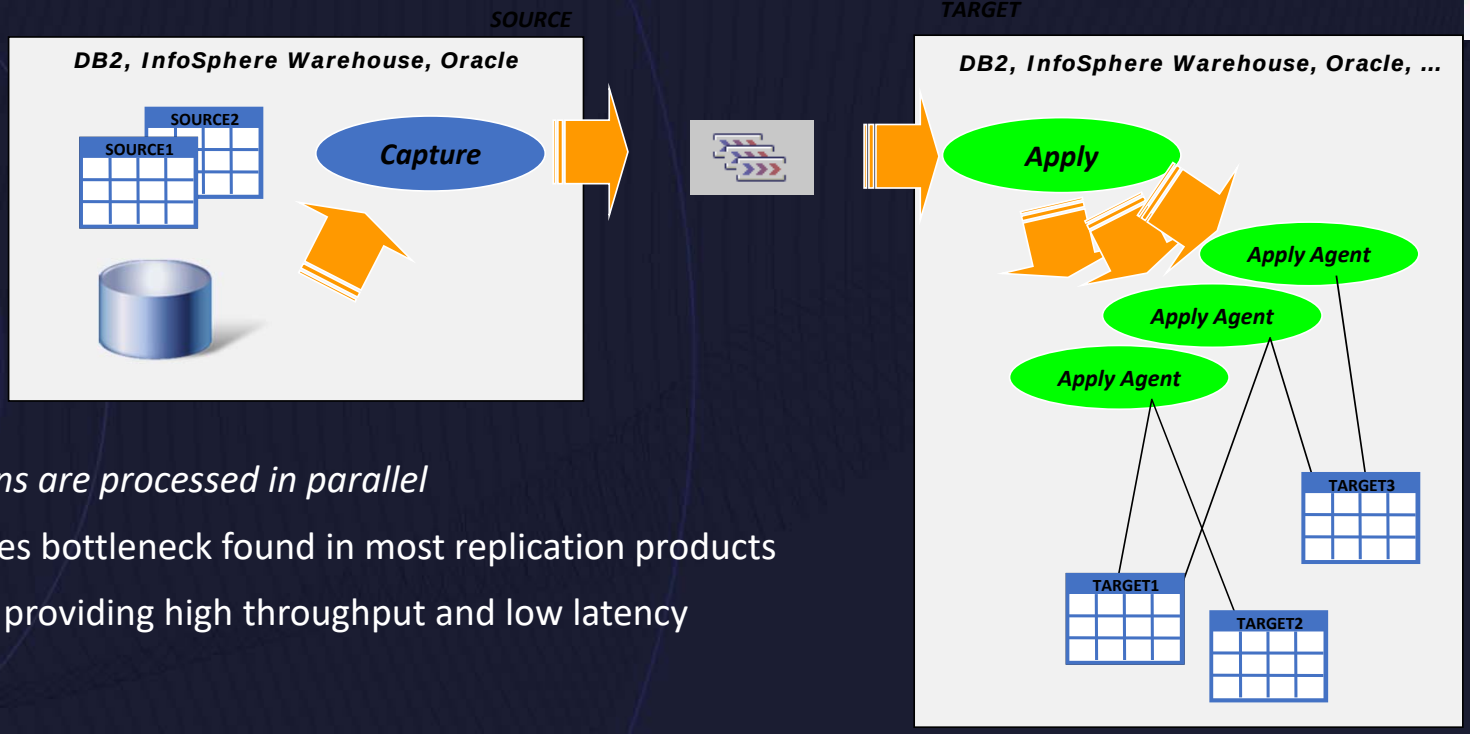
- The IIDR product includes 3 technologies that can be used
  - Q Repl – Best suited for DB2 to DB2 homogenous replication
  - CDC – Best suited for heterogenous replication
  - SQL Repl – Best suited for publish/subscription topologies
- Both Q Repl and CDC can be used for migrating from AIX to RHEL X86
  - SQL Repl will require an additional table (CD_TABLE) for every table that will be replicated
- If your source DB makes use of Referential Integrity(RI) Constraints then Q Repl is the preferred technology to use
  - CDC Best Practices state that the RI constraints on the target be dropped OR fast replay will be disabled
  - Q Repl allows for RI constraints to be on the target with fast replay enabled

Logical Replication - Q Replication

- *Change data read from database log*
- *Changes shipped directly to target*
  - No staging
- *MQ provides transport persistence*
  - A key to recoverability
- *Data then applied to target tables*
  - Highly parallel … more on next slide
- *DBA friendly*
  - All metadata in tables
  - Statistics and messages in tables



Best of Breed Performance – Parallel Apply



- *Transactions are processed in parallel*
  - Eliminates bottleneck found in most replication products
  - A key to providing high throughput and low latency
- *One of many reasons for good performance*
  - Homogeneous or heterogeneous

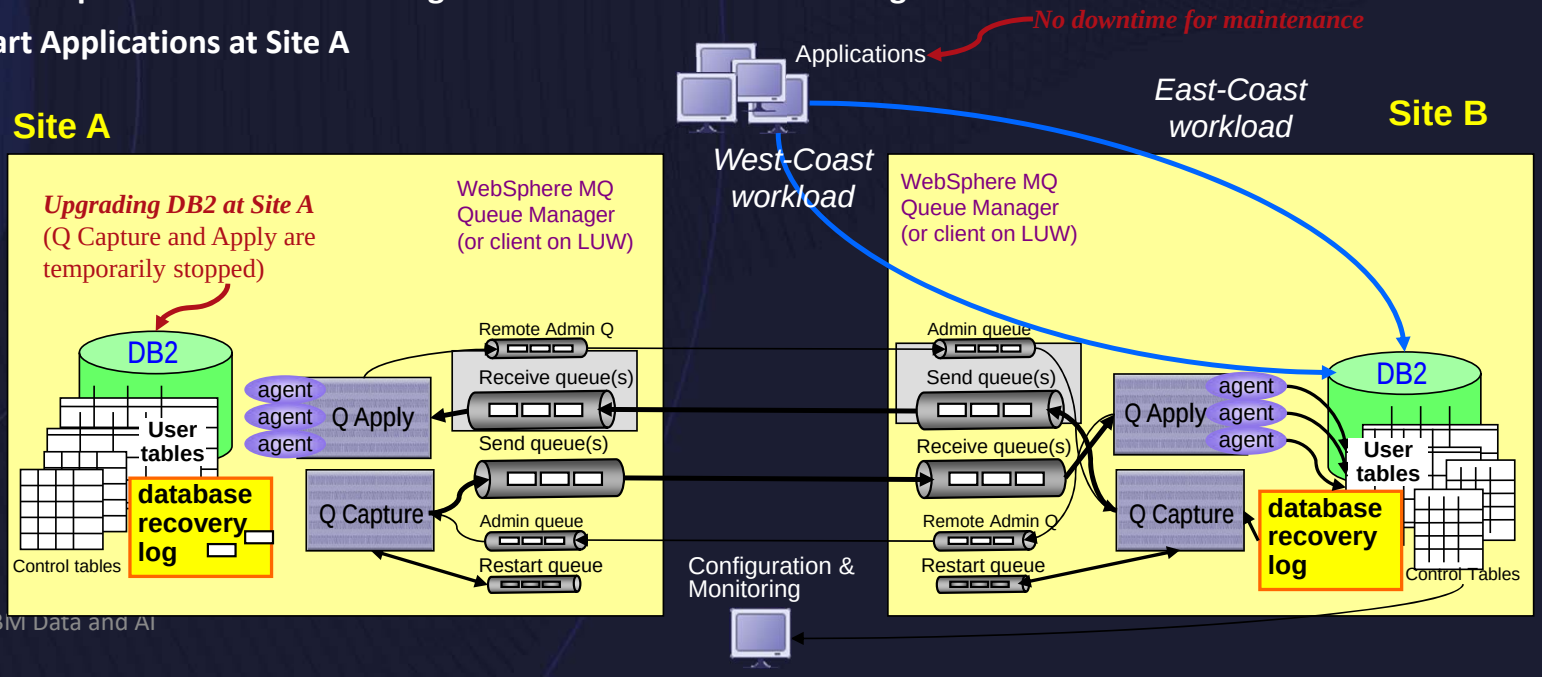
Maintenance or Upgrade of a Site

Failover:

1. **Stop Applications at Site A; Wait for replication to replicate all changes to Site B (usually a few seconds)**
2. **Reroute Applications to Site B**
3. **Perform maintenance at Site A**
   - You can continue to capture changes at site B during maintenance of A and stage them in the xmit or receive queue ready for a faster restart after the maintenance is completed; it allows DB2 logs to be archived at site B

Go-Home:

1. **Restart Replication to deliver changes accumulated at site B during the maintenance**
2. **Restart Applications at Site A**



Agenda

- Database Migration Planning
- Resource Sizing
- Cluster Manager Impact
- Database Migration tools
  - Replication Challenges
- Db2 P-Series optimization
- Db2 RHEL optimization
- Guidelines

Replication Challenges

- Tables with no unique indexes
- Generated Always Columns
- Tables with Identity Columns
- Data Sequences
- Large Objects
- Referential Integrity
- Tables with triggers that update other tables

Tables with no unique index

- Tables with no primary key or unique index can be replicated, but they cannot be loaded automatically by Q Replication unless the source application is temporarily suspended.
- Use one of the following methods if you need to load target tables with no primary key or unique index: Use the no load option for the Q subscriptions to those tables and load them at both sites before starting these Q subscriptions.
- Let replication perform the load, but suspend any changes to the source table until the load has completed at the target and the Q subscription is in active state.
- Create a unique index at the target, if possible, for better replication performance

Generated Always columns

- A Db2 system does not allow the Q Apply program at the target to update columns that are defined with a GENERATED ALWAYS clause by using the value that was generated at the source database.

Tables with Identity columns

For identity columns the range of values that are generated must not overlap between the source and the target databases. For example:

```
CREATE TABLE CUSTOMER (ID SMALLINT
        GENERATED BY DEFAULT AS IDENTITY(START WITH 100 INCREMENT BY 1),
        NAME VARCHAR(30));
```

Follow these recommendations:

•Use an odd-even scheme if you are replicating between two sites, or different ranges at each site. For example, if you use ranges, alter the table at the target:

```
ALTER TABLE CUSTOMER ALTER COLUMN ID RESTART WITH 16000
```

In this example, values 1 to 15999 will be used at site 1, and values 16000 and greater at site 2.

•When you use replication in an active/standby configuration (where read-write operations against the same data are allowed only on one site until failover to the standby), write a script to alter identity columns at the standby site to start from MAX(identity-column-name)+1 before failover of the application to the standby site. For example:

```
SELECT MAX(ID)+1 FROM CUSTOMER ALTER TABLE CUSTOMER ALTER COLUMN ID RESTART WITH
:val
```

Tip: If you have a unique constraint on the identity column, Q Replication can detect overlaps. If these occur, Q Apply follows the conflict action that is defined for the Q subscription.

Data Sequences

Sequences are database objects that you can use to generate values for insert or update into tables. For example, insert into t2(col1) values(next value for myseq). Follow these recommendations:

- Partition sequence generation between sites, using either an odd/even scheme or different value ranges, so that values do not overlap. A sequence that can generate overlapping values cannot be used in an active/active configuration without a high risk of conflicts.
- **Active/standby configurations:** In situations where the application will failover in case of a disaster or for maintenance, alter the sequence to max+1 before rerouting the application, for example: ALTER SEQUENCE ORG_SEQ RESTART SELECT NEXT VALUE FOR ORG_SEQ, ORG.* FROM ORG.

Large Objects (LOBS)

- LOB data can be replicated with no size limit: Replicating LOB data reduces the maximum replication throughput because large LOBs are fetched from the source database by the Q Capture program when it finds a change to a LOB column in the log record.
- LOB data can require a considerable amount of additional disk space at the target if the Q subscription is defined with automatic load and the Q Apply program selects the LOAD from CURSOR option that uses a nickname. In this case, Db2 at the target creates a temporary table space and materializes the entire contents of the LOB data. If LOB data is larger, use LOAD_TYPE 3 (EXPORT and LOAD utilities) for the Q subscription, or use a manual load in which you load the table outside of replication, rather than LOAD from CURSOR.

Referential-integrity constraints

Very complex RI dependencies affect the database performance and the maximum level of parallelism that the Q Apply program can achieve. Replication supports RI constraints on replicated objects, but the constraints affect Db2 and replication performance. For improved performance, avoid overly complex constraints if and when possible.

Tips: Follow these tips to improve replication performance with referential integrity constraints:

- For fallback after a prolonged outage, drop RI constraints at the site that is being brought back online for faster resynchronization. Restore the RI constraints before rerouting applications to that site.
- Use the igncasdel parameter to instruct the Q Capture program not to replicate delete operations that result from the delete of parent rows (cascading deletes). This method gives better performance because fewer changes are propagated.

RI Constraints Enhancement to Q Repl

Requires source Q Capture to ensure always sending Foreign Key data values for child tables even if those columns did not change.

- Q Capture sending RI column data values automatically controlled by a new optional column SEND_RI_COLUMN_VALUES (='Y'/'N') in IBMQREP_SENDQUEUES table :

```
ALTER TABLE !CSH.IBMQREP_SENDQUEUES ADD COLUMN
SEND_RI_COLUMN_VALUES CHAR(1) WITH DEFAULT NULL;
```

- Q Apply doing RI dependency by values is controlled (at RECVQ level) by a new optional column RI_DEPENDENCY_BY_VALUES (='Y'/'N') in IBMQREP_RECVQUEUES table :

```
ALTER TABLE !ASH.IBMQREP_RECVQUEUES ADD COLUMN RI_DEPENDENCY_BY_VALUES CHAR(1) WITH
DEFAULT NULL
```

Tables with triggers that update other tables

Tables with triggers that update other tables can be replicated; however you must choose one of the following actions if both the table that includes the trigger and the table that receives the triggered value need to be replicated:

Unidirectional configurations

If you can remove the trigger at the target, do so. Nothing else needs to be done. Create Q subscriptions for both tables and let replication propagate the logged changes to both tables. This method can be attractive, because it removes the overhead of firing the trigger at the target and the cost of maintaining it.

All configurations

- Keep the triggers at both sites, create Q subscriptions for both tables, but tell replication not to replicate changes that were made by a trigger by using the Q Capture *igntrig* parameter. When you specify this parameter, Q Capture does not replicate any change to a table that were the result of a trigger on another table.
- Keep the triggers at both sites, but create a Q subscription only for the table that includes the trigger. This method is possible only if you do not need to replicate any other changes to the table that receives the triggered value.

Tip: For fallback after a prolonged outage, drop triggers at the database that is being brought back up for faster resynchronization. Restore them before you restart the applications on that database

Agenda

- Database Migration Planning
- Resource Sizing
- Cluster Manager Impact
- Database Migration tools
  - Replication Challenges
- Db2 P-Series optimization
- Db2 RHEL optimization
- Guidelines

Db2 P-Series “isims”

- Db2 does exploit several compiler optimizations for P-Series
- NX842 coprocessor can be used to offload backup/log compression
- Db2 makes extensive use of the P-Series SIMD functionality for Columnar databases

Agenda

- Database Migration Planning
- Resource Sizing
- Cluster Manager Impact
- Database Migration tools
  - Replication Challenges
- Db2 P-Series optimization
- Db2 RHEL optimization
- Guidelines

Db2 RHEL on X86 “isims”

- Preferred OS for Db2 development
- RHEL Kernel changes are made as part of DB2 installation

Agenda

- Database Migration Planning
- Resource Sizing
- Cluster Manager Impact
- Database Migration tools
  - Replication Challenges
- Db2 P-Series optimization
- Db2 RHEL optimization
- Guidelines

Guidelines

- If the database is < 250GB, use db2move
- For larger databases or those with very small outage windows consider using Q Repl.
- This is the time to restructure your layout and configuration, take advantage of it.

IDUG VIRTUAL

2021 EMEA **Db2** Tech Conference

Speaker: Dale McInnis
Company: IBM Canada Ltd.
Email Address: dmcinnis@ca.ibm.com

Please fill out your session evaluation!