IBM Software
Technical Whitepeper

August 2014

DB2 11 for z/OS: migration planning and early customer experiences

Contents

Introduction

IBM® DB2® 11 for IBM z/OS® is the 15th release of the enterprise relational database management system (RDBMS) of choice. The purpose of this white paper is not only to help customers move to the new release smoothly, safely and quickly, but also to exploit significant enhancements: performance and scalability improvements, including CPU savings; enhanced resiliency and continuous availability; and the many enhancements to the DB2 utilities.

Customers on the DB2 11 for z/OS Early Support Program (ESP) tested DB2 10 to DB2 11 migration, and DB2 11 enhancements and new features. Their experiences provide a solid foundation for the following objectives:

- Share lessons learned
- Provide migration hints, tips and additional planning information
- Provide usage guidelines and positioning on new enhancements
- Help customers to migrate as fast as possible, without compromising safety

Customer experiences, test results and feedback have enabled the publication of this white paper. To help customers structure their plans to migrate to DB2 11 effectively, this document is composed of the following sections:

- Major themes of DB2 11
- ESP highlights
- Migration considerations
- Availability
- Utilities
- Performance and scalability

DB2 11 Major Themes

Before commencing migration to DB2 11, customers first need to be satisfied that doing so will provide benefits in terms of resilience, availability, scalability, performance and cost effectiveness. Four major DB2 11 themes address those requirements: CPU savings; enhanced resiliency and continuous availability; enhanced business analytics; and simpler, faster DB2 version upgrades.

CPU savings

The DB2 11 theme that has attracted the most attention is the potential to realize CPU savings without the need for application changes , as this has the potential to reduce running costs across the entire IBM System z® software stack for Monthly License Charge (MLC) customers. Although there is no need for application changes to exploit DB2 11 to reduce CPU costs, in most cases a REBIND is required for static SQL plans. The following are CPU savings customers can expect:

- 0 to 10 percent for complex OLTP
- 0 to 10 percent for update intensive batch
- 0 to 40 percent for queries
- Additional performance improvements through use of new DB2 11 features

Further information about DB2 11 CPU savings is available in other IBM white papers, IBM Redbooks®, and DB2 11 for z/OS product documentation.

Enhanced resiliency and continuous availability

DB2 for z/OS has been the high availability RDBMS market leader for many years, and significant DB2 11 resilience and availability improvements ensure that this lead is maintained or even extended. This is important for the DB2 for z/OS customer base because businesses continually demand improvements in the areas of continuous availability and resilience, so that they can deliver a better service to their customers. To help customers minimize the impact of IT incidents, and to eliminate or minimize the impact of application changes, database schema changes, database maintenance, and software changes, DB2 11 introduces a number of improvements:

- Improved autonomics to reduce costs and improve availability
- The ability to make more online changes without affecting application availability
- Online REORG improvements, meaning less disruption
- Two new capabilities, DROP COLUMN, and the online change of partition limit keys
- The capacity of the DB2 log is extended to 1 yottabyte (YB), or 1 billion petabytes (PB), meaning that the DB2 can run for thousands of years before customers have to reset the log
- Allowing BIND/REBIND, DDL and some kinds of online REORG operations to break into persistent threads; this affects online REORG utilities which materialize pending schema changes

Enhanced business analytics

As customers seek to exploit the vast and rapidly increasing amount of data stored in their critical business applications, DB2 11 introduces a range of enhanced business analytics capabilities to help turn that data into valuable information:

- Expanded SQL, XML, and analytics capabilities
- Temporal table enhancements
- Transparent archiving—the ability to automatically insert rows deleted from a base table into an archive table
- SQL PL enhancements, including support for the array data type
- Hadoop integration
- NoSQL and JSON support

Simpler, faster DB2 version upgrades

The number and complexity of DB2 enhancements have resulted in significant changes to the DB2 catalog and directory. This means that migration to new releases has often been complicated and disruptive, and has required careful planning. With each new release, the DB2 for z/OS development team has tried to make migration easier, faster and safer, and have continued to do so with DB2 11. This requirement has been addressed in a number of ways:

- Improved product quality and reliability, achieved via an iterative approach to development: a series of three month cycles composed of 1:N rallies leading to FVT (functional verification testing), SVT (system verification test), performance test, and finally a stabilization phase ahead of the start of the Early Support Program
- Separating application changes from the DB2 system upgrade via the new APPLCOMPAT feature
- Improvements in ensuring access path stability after REBIND

ESP highlights

The DB2 11 Early Support Program (ESP), managed by IBM BetaWorks, started in February 2013, with 21 world-wide customers:

- Eleven in EMEA
- Nine in North America
- One in South America

In terms of industry sector, the breakdown was as follows:

- Seven in banking
- Five in insurance
- Three in healthcare
- Two in financial markets
- One in the automotive sector

After education workshops in Germany and the US, the first code drop was in March 2013. In July 2013, the ‘regular’ service process of maintenance being delivered via PTF and APAR fix commenced. The ESP then entered another phase, with an extra six worldwide customers joining:

- Three in EMEA
- Two in North America
- One in South America

The industry breakdown was as follows:

- Three in banking
- Two in computer services
- One in professional services

Following a highly successful ESP, DB2 11 for z/OS was announced on October 1, 2013, and the product became generally available (GA) on October 25, 2013.

DB2 11 Early Support Program customer feedback

The ESP customers reported that product quality was excellent at this early stage in the release cycle, and that reliability was also very impressive compared to previous releases. The performance improvements were good, with CPU savings for different workload types as follows:

- DRDA workload up to 20 percent CPU reduction
- CICS workload up to 18 percent CPU reduction
- Batch workload up to 20 percent CPU reduction

Although the range of features that were tested inevitably varied from customer to customer, some common themes emerged.

BIND, REBIND, DDL, and online REORG break in
DB2 11 makes it much easier for operations which bind packages or make database schema changes to break in alongside an active workload, even with persistent threads, such as protected CICS entry threads or IMS Pseudo Wait For Input (PWFI) etc. If an operation needs to invalidate or quiesce a package held by a RELEASE(DEALLOCATE) thread, this popular improvement implicitly releases that package. This makes it much easier for the operation to obtain the package locks needed to complete successfully. BIND and REBIND commands, DDL changes, and online REORG used to materialize a pending definition change (often referred to as 'deferred ALTER') all benefit from this enhancement.

Transparent archiving
Many customers need to retain historical or inactive data for longer and longer periods of time, whether for regulatory compliance, data analysis or other reasons. This typically leads to a significant increase in the amount of stored data, which can impact response times and the ability to meet SLAs, and make it more difficult to manage and schedule database housekeeping operations. Transparent archiving offers a solution by enabling applications to use the same SQL to access a base table, with or without its associated archive table, dependent on the setting of a built-in global variable. Another global variable enables a customer to specify whether the archive table is automatically populated by DB2. This is done by transparently moving rows deleted from the base table to the archive table. Alternatively, customers can implement their own set of processes and procedures for archiving data.

IFI 306 filtering by object
Applications, such as Q Replication (QREP) Capture, which read the DB2 log to replicate data updates, can take advantage of IFI filtering to ensure that DB2 only presents log records for the objects being replicated. This can improve performance and save CPU, especially where a small subset of the objects is replicated.

Online schema change
DB2 10 introduced the ability to make several types of changes to objects (indexes and table spaces) while applications are still running, such as segment size, page size, and the MEMBER CLUSTER attribute. DB2 11 adds the ability to alter partition limit keys, and more significantly, the ability in some cases to recover to a point-in-time prior to a successful materializing REORG.

Utility improvements

DB2 11 is the most effective release of DB2 in terms of utility improvements, including online REORG enhancements and improved usability, availability and performance.

Extended RBA/LSRN

Prior to DB2 11, the six-byte log record relative byte address (RBA) has provided 256 TB of log record addressing capacity over the life of a non-data sharing DB2 subsystem or member of a DB2 data sharing group. There is a process for handling the situation where the addressing capacity is exhausted, by resetting the log, but this is highly disruptive, especially for a non-data sharing subsystem. Data sharing users are affected not only by the log RBA constraints, but in some cases they can also wrap the log record sequence number (LRSN). Normally, the LRSN should provide enough capacity to last until 2042. However, if a LRSN delta is needed (for example, where data sharing has been disabled and then re-enabled), then the LRSN can be 'in the future,' meaning it will wrap much sooner.

DB2 11 solves this by expanding the RBA and LRSN to 10-bytes, delivering much greater log capacity; up to 1 Yottabyte in terms of the RBA, and up to 30,000 years in terms of the LRSN. Additionally, the extended LRSN improves LRSN granularity, effectively eliminating the possibility of spinning on the processor to wait for the LRSN to be incremented.

Once in new function mode, customers can determine whether they need to convert to the extended RBA/LRSN, decide on the timing of the conversion, and perform the conversion in piecemeal fashion, object by object.

Other popular improvements

Other enhancements which proved popular with the ESP customers include DB2 optimizer and migration improvements, and GROUP BY grouping sets.

Customer quotes for the DB2 11 Early Support Program

"Overall we are very satisfied and astonished about the system stability of DB2 V11. In V10 we experienced this in another way."

— European Insurance

"We have seen very few problems in [Installation, Migration, and Performance]. Overall, it has been a very pleasant experience!!…The quality of the code is clearly much higher than for the ESP for DB2 10…"

— European Banking/FSS

"Good code stability, no outages, no main failures, only a few PMRs…"

— European Banking

“We have been involved in several DB2 for z/OS ESP’s. This one will rank as one of, if not the smoothest one yet.”

— Large NA retailer

“I saw a significant performance improvement in recovery of catalog and directory (V10 5:53 minutes, V11 2:50 minutes). That rocks! … DB2 11 is the best version I have ever seen.”

—European Gov’t

“Overall, we have been impressed with the new version of DB2.”

— NA Manufacturer

“Higher availability, performance, lower CPU consumption amongst other new features were the benefits perceived by Banco do Brazil with DB2 11 for z/OS. During our testing with DB2 11 we noticed improved performance, along with stability.”

— Paulo Sahadi, IT Executive, Banco do Brasil

“We have seen some incredible performance results with DB2 11, a major reduction of CPU time, 3.5% before REBIND and nearly 5% after REBIND. This will significantly bring down our operating costs.”

— Conrad Wolf, Golden Living

Migration considerations

The key to a successful DB2 11 migration project is careful planning and preparation. This section provides advice and guidance to help with your planning.

Prerequisites—hardware and software

One of the first items in your DB2 11 migration plan should be to make sure that you meet the hardware and software requirements.

Hardware requirements

DB2 11 only runs on IBM zEnterprise® EC12 (zEC12), z196 and z10 processors supporting z/Architecture.

You should also plan for increased real storage requirements compared to DB2 10 for z/OS (typically up to 15 percent). One way to estimate the real storage requirement is to use the information supplied in IFCID 225 to calculate the total real storage used by DB2 10. Next, subtract the total size of the virtual buffer pools, and add an extra 15 percent to be more cautious to the remainder. Add back in the size of the virtual buffer pools, and the result provides an estimate of the real storage requirements for DB2 11.

Software requirements

The software requirements for DB2 11 are mainly, but not exclusively, at the OS level:

- z/OS V1.13 Base Services (5694-A01) or higher
- DFSMS V1 R13—the DB2 catalog is SMS managed
- Language Environment Base Services
- z/OS Version 1 Release 13 Security Server (RACF)
- IRLM Version 2 Release 3 (Shipped with DB2 11 for z/OS)
- z/OS Unicode Services and appropriate conversion definitions are required
- IBM DB2 Connect™10.5 Fixpack 2
- IBM InfoSphere® Data Replication (IIDR) 10.2.1 to support the extended LRSN in new function mode (NFM)

Prerequisites—DB2 Connect

For many customers, management of their DB2 Connect estate has been a cause for concern when migrating from one DB2 for z/OS release to another. This is typically the case where there are many workstations running a variety of DB2 Connect releases, which are not centrally managed. It can be very difficult, if not impossible, to ensure that the DB2 Connect level on all known workstations is updated to a supported release, and equally difficult to perform effective testing. This task becomes even harder when it is not possible to identify all the workstations that can connect to DB2.

DB2 11 cannot solve all these problems, but this paper provides an opportunity to outline what support is provided for DB2 Connect customers running with older, out-of-support product releases, and to provide a brief summary of current recommended best practice.

DB2 11 for z/OS in all modes—CM, ENFM and NFM—should operate with existing versions of DB2 Connect, even back to DB2 Connect Version 8. DB2 for z/OS Development will investigate any connectivity-related issues with existing applications which run on older versions of DB2 Connect, and will try to provide a fix. If there are issues which cannot be resolved with a fix or other change in the DB2 for z/OS server itself, then DB2 Connect will have to be upgraded to an in-service level in order to obtain a DB2 Connect fix.

For continuous availability during the migration process, the minimum recommended level of DB2 Connect operating with DB2 10 for z/OS is V9.7 FP6 or V10.1 FP2. This is the level that provides continuous availability to the application server when DB2 for z/OS is migrated from DB2 10 NFM to DB2 11 CM, and on through DB2 11 ENFM to DB2 11 NFM.

The minimum level for full DB2 11 for z/OS exploitation is currently V10.5 FP2. This is required for specific new functions, including support for using arrays as input and output variables with stored procedures, workload balancing and automatic client reroute support with global variables, auto-commit performance improvements, and improved client information. As customer adoption of DB2 11 broadens, it is likely that the recommended level will be set higher, with fixes for defects being included in new driver levels, and with documented best practice being updated with the new DB2 Connect and data server driver release and fix-pack levels.

While it remains the case that most DB2 for z/OS engine features in NFM are supported with any version of DB2 Connect, DB2 for z/OS development is being proactive in recommending that customers move from the DB2 Connect client or runtime client packages to using the data server (ds) driver instead.

There are many customers who periodically update their DB2 Connect release and fix-pack levels because they want to ensure that they are always running on supported versions, even if the application doesn't change. For these customers, and for those who plan to exploit new function, the recommended order of upgrade for maximum stability and availability is the following:

- The DB2 for z/OS Server
- Any DB2 Connect Servers, if present, but customers are encouraged to move to direct connect using the data server driver
- Drivers installed directly on application servers. If you are using the DB2 Connect client or runtime client, then it is recommended that you convert to using the data server driver instead.
- User workstations. Again, it is recommended that you convert from using the DB2 Connect client or runtime client to using the data server driver.

DB2 for z/OS development does recognize that there are, however, customers who push out the drivers first. These are generally driven by the need for specific application enhancements, the most common example being in the .NET arena where customers want the latest tooling and driver support for applications and application servers running on Microsoft platforms.

Pre-migration planning

It is important to identify what work needs to be done to prepare your DB2 10 subsystems and data sharing groups for migration to DB2 11. DB2 provides tooling to help you do this, in the form of the pre-migration job, DSNTIJPM.

DSNTIJPM ships with DB2 11 and should be run on DB2 10 to identify any pre-migration catalog clean-up requirements. The same job is shipped for DB2 10 via APAR PM94057 in the form of job DSNTIJPB to maximize the available time for you to prepare to migrate.

In either case, make sure you have applied APAR PI11236 before running the pre-migration job. Run DSNTIJPM or DSNTIJPB, read the various reports produced to check for situations needing attention before migration and take the actions recommended by the report headers. DSNTIJPM/ DSNTIJPB may provide extra assistance in the form of DDL or utility statements for any required clean-up jobs.

Important preparation

There are some important steps you should take in DB2 10 before starting migration. Any old plans and packages which were rebound prior to DB2 V9.1 or before should be rebound. Otherwise, they will be automatically rebound by DB2 11 on first use, preventing you from exercising control over access path selection.

Any views, MQTs, or table functions with a period specification created in DB2 10 should be dropped, as they are no longer supported in DB2 11. A period specification is only valid if specified on a base table.

Items deprecated in earlier versions, now eliminated

Identify if you are using any features which had been deprecated in earlier releases of DB2, and have now been eliminated:

- Password protection for active log and archive log data sets is no longer supported. Remove password protection prior to migration to DB2 11.
- The DSNH CLIST no longer supports values of V8 or V9 for NEWFUN—use V10 or V11 instead. Update your DSNH procedures when you migrate from DB2 10 NFM to DB2 11 CM.
- Some DB2 supplied routines are no longer supported:
  - The Explain stored procedure, SYSPROC.DSNAEXP is obsolete—use the EXPLAIN privilege and issue EXPLAIN directly.
  - The MQ Application Messaging Interface (AMI) DB2 MQ (DB2MQ) functions in schemas DB2MQ1C, DB2MQ2C, DB2MQ1N and DB2MQ2N are now obsolete—use the MQI-based functions in schemas DMQXML1C and DMQXML2C instead (see APAR PK37290 for further guidance). You should make these changes before migrating to DB2 11.

- The DSNHDECP application programming default parameter CHARSET is no longer supported. This was used to specify whether the character set associated with the default EBCDIC CCSID was ALPHANUM or KATAKANA, and prior to DB2 V8.1 was used by the DB2 parser. However, from DB2 V8.1 onwards, statements are parsed in Unicode instead of EBCDIC, and the parser therefore no longer needs to know whether the character set is alphanumeric or Katakana. Before assembling DSNHDECP at DB2 11, make sure this parameter is not specified in the DSNHDECP source deck.
- The BIND PACKAGE ENABLE and DISABLE (REMOTE) options no longer support the REMOTE sub-option with a list of location names or LU names—specific names cannot be specified. Alter your BIND and REBIND procedures to make sure the REMOTE sub-option is no longer used. Be aware that ENABLE(REMOTE) and DISABLE(REMOTE) will enable or disable all remote connections for the affected package.
- Support for Sysplex query parallelism is removed. Single member parallelism is still supported, and any packages that used Sysplex query parallelism prior to DB2 11 will use CPU parallelism in DB2 11.
- The DSN1CHKR utility is no longer available, as there are no links in the catalog or directory. You should therefore remove DSN1CHKR from your catalog and directory maintenance procedures.

Preparing your current DB2 10 NFM for migration to DB2 11 CM

To prepare your DB2 10 NFM systems for migration to DB2 11, there are some further steps you must take. To begin migration, your DB2 10 subsystem must be at the proper service level. The starting point is to apply the fall-back SPE APAR, PM31841 and any prerequisite fixes. Info APAR II14660 details any further service recommendations, but early on in the lifecycle of DB2 11, it is strongly recommended that you are vigilant on maintenance, paying special attention to HIPER and PE APARs.

If you are non-data sharing, the only requirement is that your DB2 10 NFM system must have been started with the SPE APAR PM31841 applied; otherwise, DB2 11 will terminate when you start it and you will be unable to progress to DB2 11 CM until you have restarted your DB2 10 system with the correct maintenance.

If you are data sharing, before migrating the first member to DB2 11, all other started DB2 10 members must have the fall-back SPE APAR applied for DB2 11 to start. If there are any active members without the SPE applied, DB2 11 will terminate. The recommended practice is to restart all data sharing members with the SPE applied before commencing migration, even if you plan to have a single member active when you start DB2 11 for the first time.

You should also ensure that there are no entries in SYSUTILX prior to starting DB2 11 for the first time, and prior to entering Enable New Functioning Mode (ENFM); otherwise CATMAINT and ENFM will not execute as DB2 11 no longer blindly re-initializes SYSUTILX. This means all utility processing must be cleaned up prior to shutting down DB2 10.

You should run standard online REORGs against the catalog and directory objects prior to commencing ENFM/NFM migration, preferably as part of a different change slot. The objective is to check that REORG can break in, to check the data consistency of catalog and directory, and to improve the performance of the ENFM process.

DB2 11 Migration overview

The following diagram illustrates the DB11 migration process.

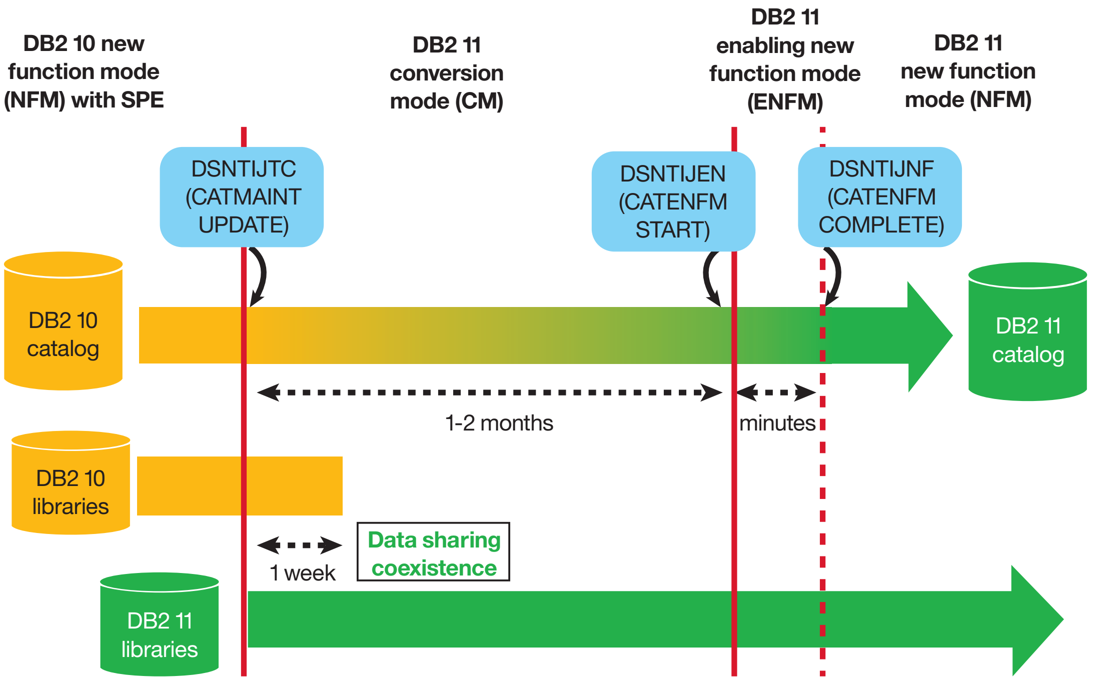
DB2 10 new function mode (NFM) with SPE
DB2 11 conversion mode (CM)
DB2 11 enabling new function mode (ENFM)
DB2 11 new function mode (NFM)
DSNTIJTC (CATMAINT UPDATE)
DSNTIJEN (CATENFM START)
DSNTIJNF (CATENFM COMPLETE)
DB2 10 catalog
DB2 11 catalog
1-2 months
minutes
DB2 10 libraries
1 week
Data sharing coexistence
DB2 11 libraries

Figure 1: DB 11 migration process

Once you have started and stopped DB2 10 with the SPE applied, you can start DB2 11 for the first time. Once restart has completed, run the CATMAINT UPDATE job, DSNTIJTC. This takes you into DB2 11 Conversion Mode (CM). If you are data sharing, you enter a state called ‘data sharing coexistence’ until you have started all data sharing members with the DB2 11 libraries. It is strongly recommended that you minimize the time in data sharing coexistence to avoid side effects such as different access paths being chosen on different members.

Once you are confident that you will not fall back to DB2 10, you can then enter ENFM, by running the CATENFM START job, DSNTIJEN, and enter new function mode (NFM) by running the CATENFM COMPLETE job, DSNTIJNF. Once this last job is complete, your catalog will have been fully converted to a DB2 11 catalog.

Migration and fall-back paths

DB2 11 always enables you to drop back to the previous stage of the migration process. However, you cannot fall back to DB2 10 NFM after entry to DB2 11 ENFM, but you can return to DB2 11 CM*. The star (*) indicates that at some point you entered a later phase of the migration process, but decided to fall back. Note that, as the following diagram illustrates, you can fall back from NFM to either CM* or ENFM*. The numbers by the arrows refer to the jobs you have to run when you move from one phase or mode to another.

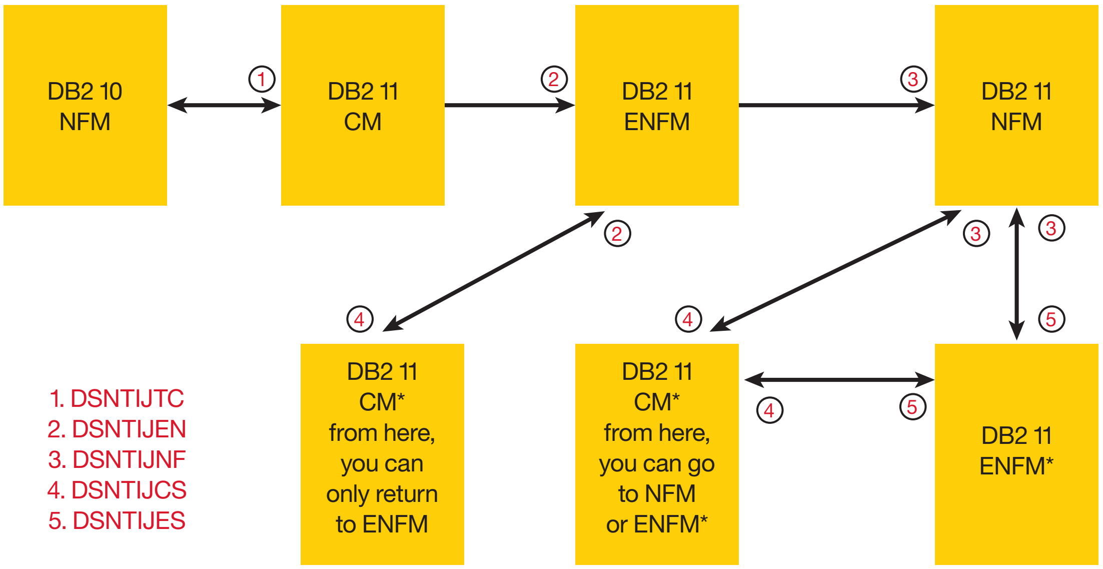

Figure 2: Migration and fall-back paths

APPLCOMPAT—Application compatibility

Customers have typically faced a significant challenge when preparing to migrate to a new DB2 release: dealing with any SQL incompatibilities introduced by the new release that need to be addressed before migration. To provide a solution to this problem, the requirements must first be defined.

The requirement

To eliminate the need to deal with incompatible SQL DML and XML by making application changes during migration requires that the process of changing application programs be de-coupled from the system migration to the new DB2 release that introduced the incompatible SQL DML and XML changes.

To be useful to customers, any solution should provide a mechanism to identify application programs affected by incompatible SQL DML and XML changes, and should also provide a mechanism to introduce changes at an individual application program (package) level. This means enabling support for phasing in application program changes over a much longer time, providing support for mixed DB2 release co-existence in data sharing and enabling support for up to two back level releases of DB2 (N-2).

The solution

APPLCOMPAT delivers on those requirements. It separates DB2 system migration to the new DB2 release from application program migration to deal with incompatible SQL DML and XML introduced by the new release.

DB2 11 provides a new APPLCOMPAT ZPARM that specifies the default value for the new BIND and REBIND APPLCOMPAT option. As this is the first release of DB2 to support APPLCOMPAT, there are only two valid values, both for the ZPARM and for the BIND/REBIND option: V10R1 for DB2 10 SQL DML and XML behavior; and V11R1 for DB2 11 SQL DML and XML behaviour. The default ZPARM setting is V11R1 for new installs and V10R1 for migration.

The APPLCOMPAT BIND/REBIND option can be used to override the default value specified by the ZPARM, and for dynamic SQL the CURRENT APPLICATION COMPATIBILITY special register performs the same function.

You should be aware of the limitations of APPLCOMPAT. It does not address issues with new reserved words or other incompatibilities that could only be resolved by having multiple levels of the DB2 parser, and the BIF_COMPATIBILITY ZPARM introduced in DB2 10 is independent of APPLCOMPAT.

Because APPLCOMPAT affects behavior at the application program level, new SQL functionality available in DB2 11 NFM cannot be used until the relevant package is bound with an APPLCOMPAT value of V11R1.

When migrating from DB2 10 NFM, APPLCOMPAT is automatically set to V10R1 prior to NFM. If you attempt to override this setting with BIND or REBIND, DB2 will cause the operation to fail and returns an error message:

```
DSNT225I  -DSN BIND ERROR FOR PACKAGE location.collid.member
APPLCOMPAT(V11R1) OPTION IS NOT SUPPORTED
```

DB2 11 delivers new instrumentation to identify where DB2 10-specific functionality is used, via IFCID 376. This provides a summary of DB2 10 function usage, and IFCID 366 provides detailed information, identifying the packages involved. It is expected that changes necessary to avoid V10R1 usage will only be made after reaching NFM.

One way to start converting applications to be tagged with V11R1 is to rebind those applications that don't have a dependency on DB2 10-specific functionality first, and then move on to convert those applications which are dependent. A workaround to distinguish packages that have to run as V10R1 until they are corrected is to annotate the package using

```
SQL COMMENT ON PACKAGE colid.name."version" IS 'V10R1'
```

If version is a pre-compiler timestamp then the double quotes are necessary. This information is stored in the REMARKS column of the SYSIBM.SYSPACKAGE table and can be queried and exploited by housekeeping to ensure that APPLCOMPAT(V10R1) is used for these packages until they are converted and the comment is removed or updated.

Availability

A significant factor influencing customers to migrate to DB2 11 is the opportunity to improve application availability, and there are a number of areas where customers need to plan to exploit this opportunities.

BIND/REBIND/DDL/Online REORG Break In With Persistent RELEASE(DEALLOCATE) Threads

Making use of persistent threads running packages bound with RELEASE(DEALLOCATE)—such as IMS Pseudo WFI regions and CICS Protected ENTRY threads—can potentially provide very significant performance benefits. Prior to DB2 11 they can block processes such as BIND, REBIND and DDL scripts, and can also block two types of the online REORG utility, REORG REBALANCE, and a materializing REORG which invalidate packages.

DB2 11 addresses this with a new form of behavior, which is on by default but can be controlled by ZPARM PKGREL_COMMIT. It enables BIND, REBIND, DDL and online REORG to 'break-in' to persistent threads running packages bound with RELEASE(DEALLOCATE) and complete successfully. Break-in is not guaranteed and is performed on a best efforts basis. It can handle idling threads that are at a transaction boundary—that is, where commit or abort was the last thing performed.

Several factors determine whether break-in is successful or not. The persistent thread must COMMIT for the mechanism to work, and the timing and frequency of COMMIT are both key in allowing break-in. To improve the chances of a successful break-in, consider increasing the ZPARM for the IRLM resource timeout interval (IRLMRWT).

The break-in mechanism does not apply when you are running packages bound with KEEPDYNAMIC(YES), you have open WITH HOLD cursors defined WITH HOLD at the time of COMMIT, or if the COMMIT happens inside a stored procedure. However, operations such as BIND, REBIND, DDL and some types of online REORG would not be able to break-in under these conditions, even with threads running packages bound with RELEASE(COMMIT).

To be able to best exploit the break-in mechanism, it's important to understand how it works, step by step.

Step One.
When BIND/REBIND/DDL/Online REORG is initiated it can wait for up to three times the IRLM timeout limit (IRLMRWT) for a package lock, after which it will time-out.

Step Two.
At half of the IRLM timeout limit, DB2 will get notified by IRLM that a process is stuck waiting for a package lock. If the holder of the lock has an S-lock, then DB2 will post a system task to take further action.

Step Three.
The DB2 system task is awakened and checks to see if a 'recycle' of threads has been done in the last ten seconds—the following steps describe the recycle process. If no recycle has been done, then the break-in operation will proceed. The reason for the ten-second threshold is that DB2 is trying to avoid flooding the system with recycle requests during BIND/REBIND/DDL/Online REORG activity.

Step Four.
In data sharing, the member where BIND/REBIND/DDL/Online REORG is running sends a broadcast to all the other members in the DB2 data sharing group to perform a recycle of locally attached threads.

Step Five.
At this stage, the task loops through all locally attached threads (not distributed ones) to see if the time since they last issued commit or abort was more than half that of the IRLM timeout limit (IRLMRWT). If so, then the chances are that BIND/REBIND/DDL/Online REORG is waiting on them.

Step Six.
Next, DB2 tests to see if it can do anything to release the package locks. To pass this test, each thread must be at a transaction boundary; that is commit or abort was the last DB2 operation carried out. If so, DB2 can process the thread recycle request.

Step Seven.
DB2 'fences' the API for the thread to ensure the applications do not drive any new DB2 requests, takes control of the agent structure and drives a 'dummy COMMIT.' The commit is safe since the thread is already at a transaction boundary. DB2 acts as the coordinator as this is a single-phase commit, and releases the agent structure. On the COMMIT, RDS can see that there is a waiter for a package lock held by this agent and switches behavior to RELEASE(COMMIT) for this commit cycle. The package lock is then freed, taking the BIND/REBIND/ DDL/Online REORG one step closer to breaking in.

Step Eight.
DB2 repeats this process for all qualifying threads.

Step Nine.
BIND/REBIND/DDL/Online REORG should break-in, provided that there are no blocking threads holding the package lock that could not be recycled; for example, a long running read-only application process without any intermediate commits.

Step Ten.
If the application starts using the thread during the recycle processing, it is blocked at the API level. DB2 causes the thread to spin in a timer-based wait loop until the recycle is complete – DB2 waits approximately a millisecond between polls to see if the thread is available to the application. To protect the recycle process, DB2 also fences end-of-task conditions such as cancelling an application TCB, or end-of-memory conditions such as forcing down the home ASID, for the duration of the recycle.

Persistent thread break-in—ESP customer testing experiences

Customers on the DB2 11 ESP tested persistent thread break-in for BIND, REBIND, DDL and online REORG extensively, and the following table summarizes the results of that testing.

Action	Thread Type	Results
BIND	Batch, no Commit	No break-in
BIND	Batch, frequent Commit	Break-in
BIND	50* CICS ENTRY	Break-in
DDL	CICS ENTRY	Break-in
Create Index	CICS ENTRY	Break-in
Drop Index	CICS ENTRY	Break-in
Alter Table Add Column	CICS ENTRY	Break-in
Alter Index (NO) cluster	CICS ENTRY	Break-in
Alter Tablespace to UTS	CICS ENTRY	Break-in
Alter Partition	CICS ENTRY	Break-in

Improved control when altering partition limit keys

Altering limit keys for partitioned table spaces can seriously impact availability, and the behavior is different depending on how the table partitioning is controlled.

With table-controlled table partitioning, alteration of limit keys is a pending ALTER, which is instantiated by a materializing REORG. Dropping of these alters can occur at any time prior to the REORG.

With index-controlled table partitioning, if the limit key change is done via ALTER INDEX ALTER PARTITION, the partition goes into 'hard' reorg pending (REORP), and access to the partition is restricted until REORG is run, impacting availability. The tablespace remains index-controlled. In terms of control, the ALTER cannot be withdrawn once it has been issued.

Again with index-controlled table partitioning, if the alter is done by ALTER TABLE ALTER PARTITION, there are two possible behaviors. If the partition is not empty, then it goes into 'hard' reorg pending (REORP), and the tablespace is automatically converted to table-controlled partitioning. As before, the ALTER cannot be withdrawn after it has been issued. If the partition, on the other hand, is empty, the ALTER is executed immediately, and the tablespace is again converted to table-controlled partitioning. In either case, the conversion may be unwelcome.

DB2 11 doesn't change the above behavior, but it introduces controls which can prevent changes being injected into the system that can affect availability. Two new ZPARMs are introduced: PREVENT_ALTERTB_LIMITKEY and PREVENT_NEW_IXCTRL_PART.

PREVENT_ALTERTB_LIMITKEY causes any attempt to change partition limit keys using ALTER TABLE ALTER PARTITION to fail with SQLCODE -876. However, changing partition limit keys using ALTER INDEX ALTER PARTITION is still possible – the recommendation is that you do not use this in production because of the impact of the REORP condition.

PREVENT_NEW_IXCTRL_PART stops you from creating new index-controlled partitioned tablespaces, thereby ensuring you gain the availability benefits of table-controlled partitioning.

In addition to these benefits, a materializing REORG can now break-in to a persistent thread running a package bound with RELEASE(DEALLOCATE).

REORG REBALANCE, in which DB2 alters the partition limit keys, can now be run SHRLEVEL CHANGE. It can't be run for any partitions with pending ALTER LIMITKEY changes, but will work for the other partitions, and will work for partitions which are 'hard' reorg pending (REORP).

DROP COLUMN

DROP COLUMN has been a long-standing customer requirement, and is introduced in DB2 11. ESP customers reported that this feature works well. You can convert to a UTS tablespace and concurrently DROP COLUMN in the same materializing REORG. The materializing REORG can be run at the partition level, but the tablespace remains in advisory REORG pending status (AREOR) until all partitions have been reorganized. Note that all packages touching the table will be invalidated.

There are some restrictions associated with DROP COLUMN. It cannot be used against the classic tablespace type – DB2 will fail the SQL statement with SQLCODE -650. Any attempt to drop a column contained in an index or view will fail with SQLCODE -478. If you have a pending DROP COLUMN, you cannot add another column before the materializing REORG. You cannot create a view on a table with a pending dropped column. In both of these cases, DB2 will return SQLCODE -20385. If you try to drop the same column a second time before the materializing REORG, DB2 will return SQLCODE -205. You cannot unload from an image copy taken before the materializing REORG—utility message DSNU1227I will be issued. Finally, if you attempt to recover to a PIT before the materializing REORG, the utility will fail with message DSNU556I.

Utility enhancements

DB2 11 introduces a number of utility enhancements which improve availability and usability.

REORG enhancements

A new SWITCHTIME option avoids the need for multiple jobs to control the start of the drain for the switch phase. You can specify a time at which the switch phase must occur, enabling you to automatically synchronize this phase with an application quiesce point or quiet time.

Prior to DB2 11, when you reorganized a subset of partitions, an image copy of the entire tablespace was taken. This can substantially increase the space required for image copy data sets. DB2 11 enables you to take partition level image copies when reorganizing a subset of partitions. This includes tape support, but there is no support yet for STACK YES. It introduces an additional benefit for RECOVER processing by permitting parallel recoveries at the partition level. Keep in mind that exploiting this feature will require changes to your existing jobs.

DB2 11 enables even more control over REORG processing by adding RECLUSTER YES|NO support for REORG SORTDATA NO. The combination of SORTDATA NO and RECLUSTER NO bypasses sort, and avoids using the clustering index to unload data. This can be useful for materializing REORG utilities, and will speed up conversion to the extended format RBA/LRSN. This is a useful feature, but it only saves time if the data is already in clustering order. It is recommended that you do not use this feature on huge tables with a clustering index where the data rows are not already clustered, as the REORG could run for a long time. A new utility message indicates the method used to unload the table:

DSNU2904I DATA RECORDS WILL BE UNLOADED
VIA unload-method

where unload-method is one of:

- CLUSTERING INDEX
- TABLE SCAN
- TABLE SPACE SCAN

DB2 11 introduces another REORG option, DRAIN_ ALLPARTS YES (not the default), which has the potential to significantly reduce the impact on application availability. The objective of this option is to avoid deadlocks between drains and claims across non-partitioning indexes (NPIs) and data partitions when reorganizing a subset of the partitions.

This is achieved by momentarily obtaining the tablespace-level drain first, before draining the partitions and indexes being reorganized. This is more likely to be successful in getting a DRAIN to make the SWITCH than the old method of serially draining all the target partitions. DB2 11 customers have seen big reductions in the elapsed time needed to complete DRAIN and SWITCH processing, and REORGs should run with less problems using this feature.

To help customers get a better understanding of REORG progress, new REORG utility message DSNU1138I provides information about drain begin and end times.

To enhance control of REORG, a new PARALLEL option helps you to control the degree of utility parallelism by specifying the maximum number of subtasks that DB2 can start.

As always with migration, you should be aware of changed REORG defaults. For example, NOPAD YES is now the default for REORG DISCARD.

A new LOGRANGES NO option for REORG enables you to specify that REORG should bypass SYSLGRNX processing when determining what ranges of log records need to be processed during the LOG APPLY phase. This option should only be used when SYSLGRNX is known to be logically corrupt or has to be reset.

The ability to run REORG REBALANCE SHRLEVEL(CHANGE), has already been touched on and is a big step in the right direction for 24/7 availability. REORG will also build a compression dictionary for empty partitions as well as the ability to break-in on persistent threads running packages bound with RELEASE(DEALLOCATE).

DB2 11 delivers on a customer requirement that was raised as soon as UTS tablespaces were introduced, and that is partition pruning for UTS Partition By Growth (PBG) tablespaces. Setting new ZPARM REORG_DROP_PBG_PARTS=ENABLE provides the option to physically remove empty UTS PBG partitions. This ZPARM is disabled by default, so you need to make a conscious decision to enable it because there is no support for point-in-time recovery to a point in time prior to the SWITCH phase for a tablespace where empty partitions have been pruned.

Automatic mapping tables

DB2 11 delivers an often-requested feature; automatic building of the mapping table used by REORG. The behaviour is different, depending on the DB2 mode, CM or NFM.

In DB2 11 CM, if there is an existing mapping table in the DB2 10 or DB2 11 format, it will be reused. If a mapping table does not exist, the mapping table will be automatically created and will be dropped at the end of REORG.

In DB2 11 NFM, if there is an existing mapping table in the DB2 11 format, it will be reused. If it's in the DB2 10 format, a new mapping table will be automatically created in the same database as the original mapping table. If a mapping table does not exist, one will be automatically created in either the database specified by ZPARM REORG_MAPPING_DATABASE, or in the database specified by the MAPPING DATABASE option in the REORG job, or in DSNDB04.

This can be a useful feature, but it is recommended that this feature be used judiciously. You should still predefine and preserve mapping tables for regularly scheduled REORG jobs, in order to avoid SQL DDL contention on the catalog. It is also recommended that you use a single database, as specified in ZPARM REORG_MAPPING_DATABASE, for all mapping tables. You should modify the structure of your existing mapping tables to the DB2 11 format as part of the process of migrating to NFM, using the following SQL as a template:

```
ALTER TABLE TBMAP ALTER COLUMN LRSN SET DATA TYPE
CHAR(10) NOT NULL;
```

At the time of writing, automatic mapping table APAR PI08339 is still open, and it is recommended that you wait for the APAR to be closed and apply the PTF, when available, if you want automated building of mapping tables.

RUNSTATS and RTS enhancements

DB2 11 enables extra statistics to be collected inline by the REORG and LOAD utilities. This is acceptable, but inline statistics are estimates and do not provide the same quality of information as a stand-alone RUNSTATS utility. When collecting inline statistics, customers sometimes get the message `DSNU602I STATISTICS ARE NOT COLLECTED FOR NONPATITIONED INDEX` when running REORG PART. This can now be avoided by using the SORTNPSI option with REORG job, or by setting the REORG_PART_SORT_NPSI ZPARM to AUTO or YES. Either of those options can cause REORG to sort all of the non-partitioned index keys before building the index, if the amount of data being reorganized relative to the size of the objects exceeds internal thresholds. YES specifies that if REORG estimates, based on the thresholds, that sorting all keys of the non-partitioned secondary indexes would improve the elapsed time, all keys are sorted. AUTO is very similar, and causes all keys to be sorted if REORG estimates that elapsed time and CPU performance would be improved. The recommended option is to use AUTO, to maximize the probability of collecting inline statistics for NPIs.

One of the problems customers and IBM support have to deal with when resolving access path problems is that using different profiles or specifying different options over time can result in conflicting catalog statistics. This not only makes diagnosis of access path problems difficult, but can also mean that inaccurate information is used by the optimizer when selecting an access path. DB2 11 provides a new RESET ACCESSPATH option for RUNSTATS, which will reset all access path statistics for an object in the catalog. This does not affect space statistics in the catalog or RTS. Once the statistics have been reset, you then need to gather fresh access path statistics with RUNSTATS.

If you are using statistics profiles, make sure that the statistics profile exists. If you get message DSNU1363I THE STATS PROFILE FOR TABLE table-name NOT FOUND, indicating that the profile is not found, then RUNSTATS will use fixed defaults that might be inappropriate. Another consideration when using statistics profiles is that there is no support for USE PROFILE with inline statistics in REORG and LOAD. In this case, you need to make sure the RUNSTATS options specified in the profile are consistent with those specified for REORG and LOAD inline statistics, otherwise you risk ending up with conflicting statistics.

With DB2 and vendor products making more use of real-time statistics (RTS), there are customer demands for user control over the externalization of the RTS in-memory blocks. DB2 11 allows you to do this with the new command, -ACCESS DATABASE (DB) SP(TS) MODE(STATS).

RECOVER enhancements

DB2 11 adds Fast Log Apply (FLA) support for RECOVER INDEX. Previously, DB2 would wait until a log record was to be applied before reading the associated index page into the local bufferpool where it would then be cached. Now, DB2 uses list prefetch for all the index pages that log apply will update before applying any log records. FLA for RECOVER INDEX has the potential for significant savings in elapsed time, and means customers should now reconsider their recovery strategy for indexes, especially large NPIs. The decision to make is whether to run RECOVER INDEX in parallel with RECOVER TABLESPACE [PART], or to wait for RECOVER TABLESPACE [PART] to complete and then run REBUILD INDEX. This enhancement has been taken back to DB2 9 and DB2 10 via APAR PI07694.

DB2 11 RECOVER TABLESPACE when running under DIAGNOSE TYPE(607) provides an optimization when performing PIT recovery for a list objects. Recovery of the objects is not necessary when there have been no updates after the PIT. This optimization is only available when performing PIT recovery when TOLOGPOINT or TORBA is specified. It does not apply to log -only recoveries, RECOVER BACKPOUT or recovers to current.

Performance and scalability

DB2 11 delivers many performance and scalability improvements, and these have been discussed in detail in other Redbooks and white papers. Some of these are briefly described in this document, identifying what actions are required, if any, to benefit from them.

Performance enhancements—No REBIND needed (CM)

There are many performance improvements available in conversion mode (CM), even without rebinding your static SQL packages. The improvements described here are presented in no particular order, and the benefits you receive will be workload dependent.

- More and more applications connect to DB2 via DRDA, so the DDF performance improvements will be particularly welcome. DDF overhead can be significantly lower because DB2 11 reduces SRB scheduling for TCP/IP receive by exploiting new z/OS CommServer capabilities. This feature can reduce CPU usage in the DIST address space, and reduce network latency. DRDA OLTP auto-commit performance is improved by enabling a DB2 for z/OS server to initiate commit after the result set is exhausted in the initial reply to the client. This can help reduce network flows, latency and CPU consumption.

- DB2 11 exploits the z/OS V1.13 support for 64-bit code execution by moving xProcs generated for fast column processing above the bar, reducing data movement costs.
- All SRB-mode DB2 system agents that are not response-time critical are zIIP enabled, potentially reducing the total cost of ownership (TCO).
- CPU cycles are saved by using shared memory and avoiding the cross-memory overhead for creating log records in the log output buffer (OUTBUFF).
- Data decompression performance is improved by only decompressing the portion of the row needed either for predicate evaluation, or for returning the columns back to the application.
- INSERT performance is critical for many customers, and DB2 11 continues to deliver improved performance in this area. Design changes mean that latch contention is reduced, and CPU is reduced for INSERT column processing and log record creation. More progress is made in data sharing by avoiding LRSN spin when creating log records. There is potential for improved INSERT performance because page fix/free is avoided when writing to the group bufferpool (GBP).
- Sort performance is improved, as is XML performance and DPSI merge performance. There are also performance improvements when accessing objects with a large number of partitions.
- RELEASE(DEALLOCATE) execution is optimized so that it performs consistently better than RELEASE(COMMIT).
- Data replication capture performance is improved by the introduction of IFI 306 filtering capabilities to ensure that only the log records for the objects being replicated are presented to the capture task.
- Some workloads can experience performance degradation because of the growth in pseudo-deleted index entries, and this is addressed with the new automatic index pseudo delete clean-up feature, which is described in more detail later on in this document.
- Again for distributed applications, with ODBC/JDBC Type 2 performance improvements, and Java stored procedure performance is improved by adding support for more efficient multi-threaded JVMs, and the 64-bit JVM.
- Performance of the ACCESS DATABASE command is improved by opening multiple objects in parallel.
- DGTT performance is improved by avoiding incremental binds, reducing the CPU overhead.
- With more and more applications exploiting Unicode, optimized P-procs for LIKE predicates against Unicode tables also deliver improved performance.
- Other improvements include improved performance for ROLLBACK TO SAVEPOINT; zEC12 exploitation; other high n-way scalability improvements; and a number of data sharing performance improvements.

Performance enhancements requiring REBIND

Some performance benefits are available as soon as CM when you REBIND your static SQL packages, with or without APREUSE. These include:

- Most in-memory techniques
- Non-correlated subquery with mismatched length
- Select list do-once
- Column processing improvements
- RID overflow to workfile is handled for data manager 'set' functions
- Performance improvements for common operators
- DECFLOAT data type performance improvements

Some performance benefits are available as soon as CM when you REBIND your static SQL packages without APREUSE. In brief, these include:

- Query transformation improvements—less expertise required to write performant SQL
- Enhanced duplicate removal
- Some DPSI and page range performance improvements
- Optimizer CPU and I/O cost balancing improvements

Performance enhancements requiring DBA or application effort (NFM)

DB2 11 delivers a number of performance improvements that require DBA or application programmer effort:

- 'Suppress-null' indexes
- The new PCTFREE FOR UPDATE attribute to reduce indirect references
- Global variables
- Optimizer externalization of missing/conflicting statistics
- Extended optimization—selectivity overrides (filter factor hints)
- The open data set limit raised to 200K

Optional enhancements requiring NFM and DBA effort

There are two very important improvements, which are examined in more detail later in this document, which require that you have migrated to NFM, and also require DBA effort.

The first of these is optional, and that is to convert the BSDS to exploit extended 10-byte RBAs using sample job DSNTIJCB. To do this, a -STOP DB2 MODE(QUIESCE) is required.

The second of these, also optional, is to convert the DB2 catalog and directory table and index spaces to extended 10-byte RBA format using sample job DSNTIJCV. This reorganizes all catalog and directory table spaces using SHRLEVEL CHANGE, and can be split up to run the REORG utility jobs in parallel.

DB2 lab performance measurement summary

To get an idea of the possible performance improvements, the following chart summarizes measurements taken by the DB2 for z/OS development lab:

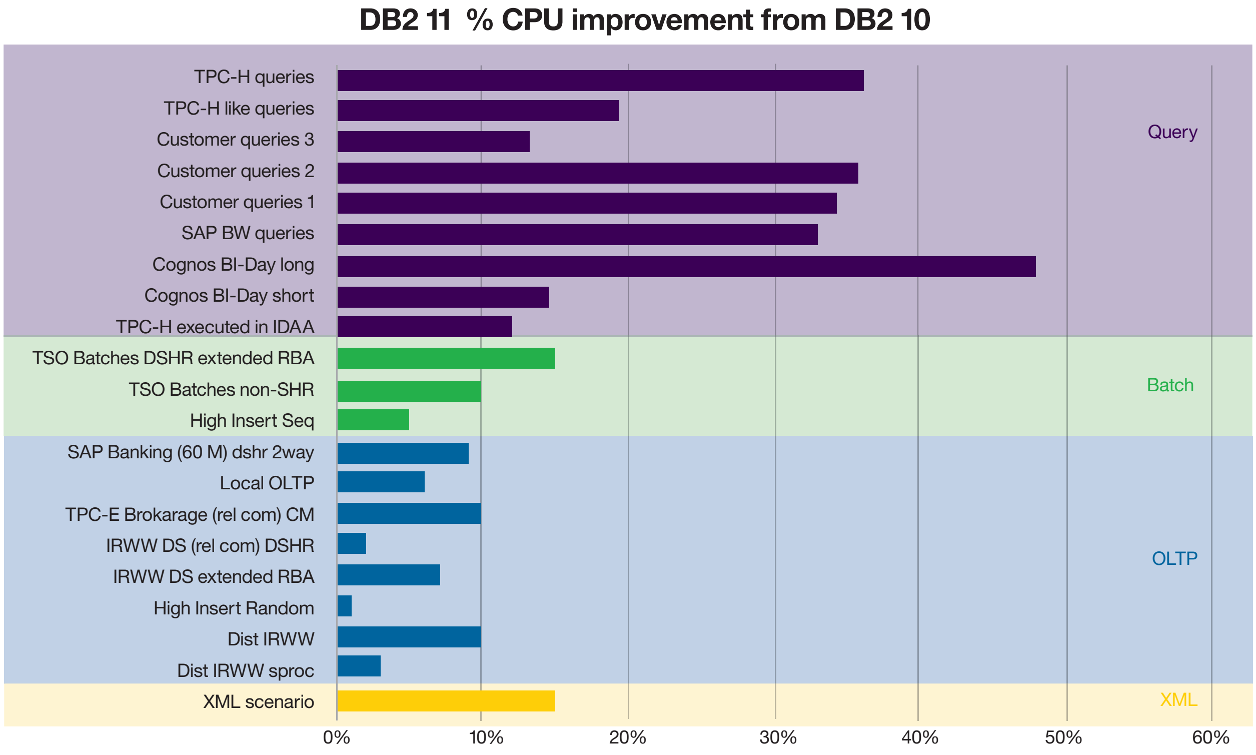

Figure 3: Measurements taken by DB2 for z/OS development

Example of customer performance testing

Customers on the DB2 11 ESP dedicated a lot of time and resource to performance testing. This section reviews not only the results of one of the customers in some detail, but also what measurements they took, and when they took them, so that you can exploit this methodology for your own evaluation of the performance benefits of DB2 11.

Starting with the times at which performance measurements were taken:

- DB2 10 NFM baseline
- DB2 11 CM before REBIND
- DB2 11 CM after REBIND
- DB2 11 NFM (no need for further REBIND)
- DB2 11 NFM after REORG (to migrate object to extended LRSN)
- DB2 11 NFM Extended LRSN

When collecting performance measurements, you should follow the example of this customer and make sure that the CPU numbers are normalized across those intervals (use CPU milliseconds per commit). It is relatively easy to combine statistics and accounting by stacking the various components of CPU resource consumption:

- MSTR TCB / (commits + rollbacks)
- MSTR SRB / (commits + rollbacks)
- MSTR IIP SRB / (commits + rollbacks)

- DBM1 TCB / (commits + rollbacks)
- DBM1 SRB / (commits + rollbacks)
- DBM1 IIP SRB / (commits + rollbacks)

- IRLM TCB / (commits + rollbacks)
- IRLM SRB / (commits + rollbacks)

- Average Class 2 CP CPU * occurrences / (commits + rollbacks)
- Average Class 2 SE CPU * occurrences / (commits + rollbacks)

This customer tested both CICS transactions and batch jobs, and to understand the results you need to understand the workload.

Starting with the test CICS Transaction Profile, a variety of fetch intensive transactions (E-Bank logon, balance check, financial statement history, and account search) were used. On average, in SQL DML terms, these transactions performed: 3 x INSERT; 7 x SELECT; 5 x OPEN; and 103 x FETCH. Typical bufferpool activity saw 65 Getpages and 13 synchronous reads. CPU consumption at DB2 10 NFM was 3.2 Class 1 msec and 2.3 Class 2 msec.

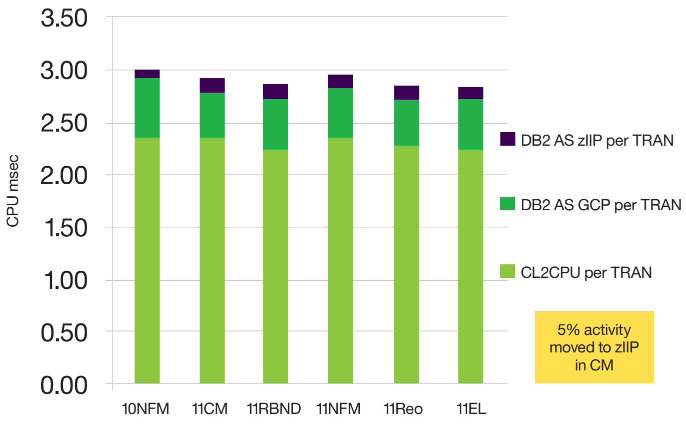

Figure 4: Progression of CICS transaction CPU time

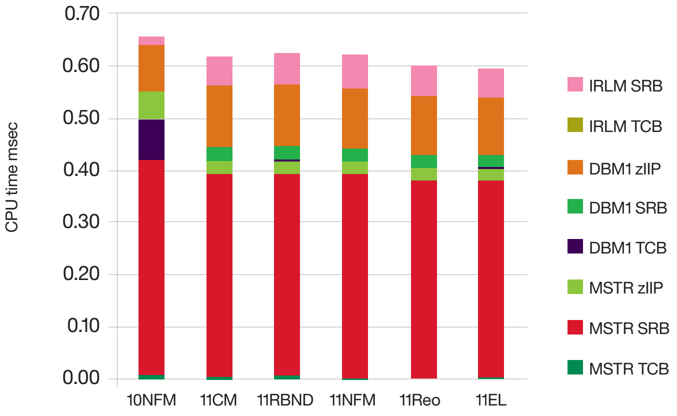

Figure 5: DB2 system addess space CPU per CICS transaction

Moving on to the test batch job profile, typical SQL DML activity consisted of 4528 commits, 37613 deletes, affecting 482836 rows, 45099 updates, 119773 selects, 525884 inserts and 548947 fetches. Bufferpool activity saw 4.8 M GETPAGE requests, 133 K synchronous reads, and 57 K dynamic prefetch requests. Class 1 CPU consumption was 69 seconds, and class 2 was 67 seconds. Class 1 elapsed was 07:02 min. As can be seen from the DML figures, this application is INSERT and DELETE intensive.

The next chart shows the CPU time progression for the batch job; the CPU time reduction with the extended RBA/LRSN is due to the elimination of LRSN spin:

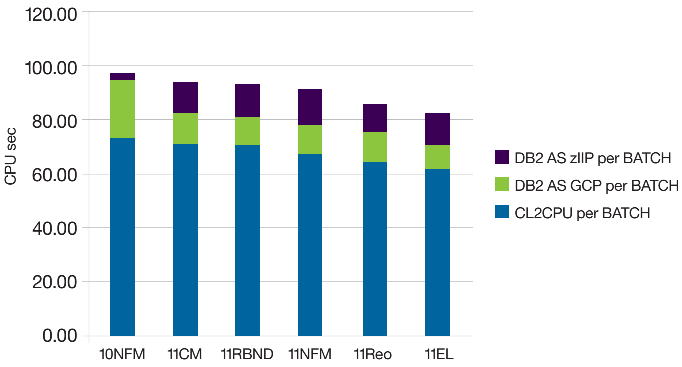

Figure 6: CPU time progression per batch

The chart for the DB2 system address space CPU shows two important things, the movement of processing to zIIP engines in CM, and LRSN spin elimination with the extended LRSN/RBA:

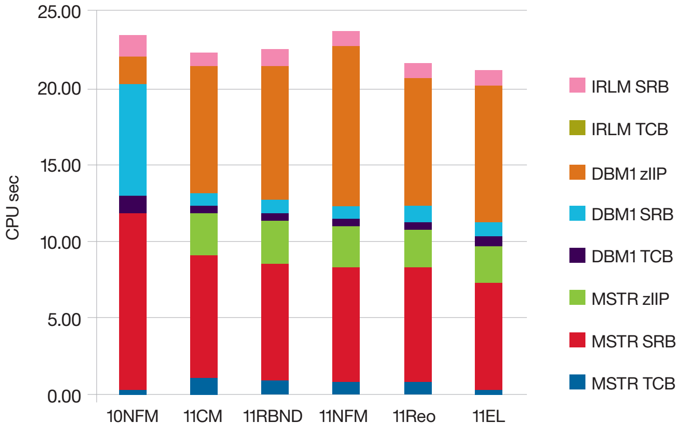

Figure 7: DB2 system address space

Automatic pseudo-deleted index entry clean-up

In looking at automatic clean-up of pseudo deleted index entries in more detail, it is useful to recap on the impact of pseudo-deleted index entries to understand the importance of this feature.

As the number of pseudo-deleted index entries grows, the size of the index increases. This means more getpages and lock requests are required, and leads to increased CPU costs and possibly longer elapsed times for SQL requests using index access. In addition, applications might encounter deadlocks and timeouts during INSERT/UPDATE/DELETE processing, because of collisions with committed pseudo-deleted index entries, and because of RID reuse by INSERT following DELETE.

Prior to DB2 11, there are mechanisms to remove pseudo-deleted entries during mainline operations. Insert and delete operations would try to remove pseudo-deleted entries from index pages. SQL running with isolation level RR (repeatable read) can also remove pseudo-deleted entries. Now, pages that only contain pseudo-deleted index entries are called ‘pseudo-empty’ index pages, and DB2 attempted to clean up pseudo-empty index pages as part of DELETE processing.

Because most customers run with isolation CS (cursor stability), pseudo-deleted index entry clean-up was often not very effective, meaning customers had to use REORG INDEX. This removes pseudo-empty index pages and pseudo-deleted entries that were not cleaned up by mainline processing. For some customers, this was problematic because of the extra CPU overhead of running REORG INDEX solely to clean up pseudo-deleted index entries and because of the difficulty in scheduling the REORG jobs.

DB2 11 provides an autonomic solution in CM, which is by default automatically turned on for all indexes 24*7. This provides automatic clean-up of pseudo-deleted index entries in index leaf pages, and automatic clean-up of pseudo-empty index pages. It is designed to cause minimal or no disruption for concurrent DB2 work. Clean-up is done under system tasks, which run as enclave SRBs and are zIIP eligible. A parent thread (there is one per DB2 member) loops through RTS to find candidate indexes for clean-up, building a list of index leaf-pages as they are accessed by index look-up or scan. The parent task schedules child threads to do the clean-up, but the child threads only clean up an index if it already is already opened for INSERT, UPDATE or DELETE on that DB2 member, to avoid making those indexes GBP dependent.

Although automatic pseudo-deleted index entry clean-up runs all the time by default, customers can minimize potential disruption by managing the number of clean-up threads or by specifying the time of day when indexes are subject to clean-up. Customers can control the number of concurrent clean-up threads or even disable the function by using the new ZPARM INDEX_CLEANUP_THREADS. The function is disabled by setting the ZPARM to 0; the other allowable values range from 1 to 128, with 10 being the default.

To control automatic pseudo-deleted index entry clean-up at the individual index level, customers can insert entries into the new catalog table, SYSIBM.SYSINDEXCLEANUP. This allows you to specify whether clean-up is enabled or disabled for an individual index, and if enabled, when clean-up is allowed to run. This provides very fine-grained control over the scheduling of clean-up.

Automatic pseudo-deleted index entry clean-up can provide significant benefits. Measurements showed up to 39 percent DB2 CPU reduction per transaction in DB2 11 compared to DB2 10, up to 93 percent reduction in pseudo-deleted entries in DB2 11, and consistent performance and less need of REORG.

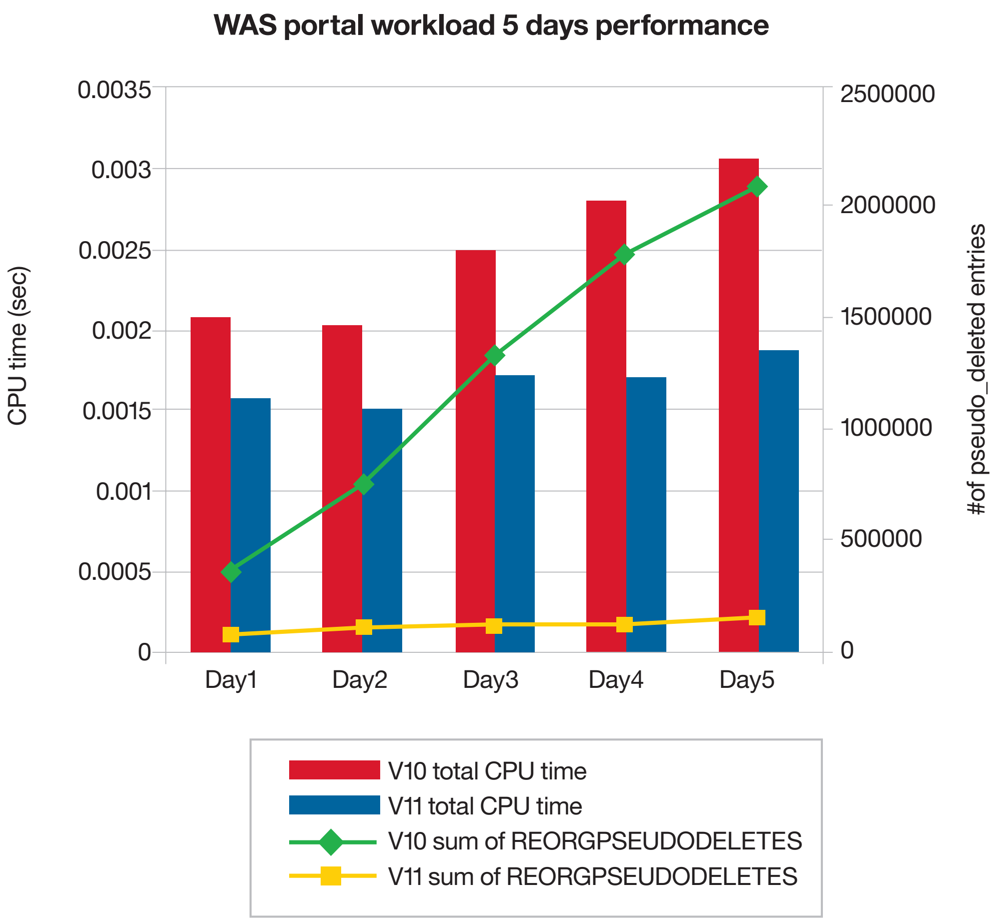

Figure 8: WAS portal workload — 5 days performance

Performance enhancements

A number of other performance enhancements have already been briefly discussed, and for some of these there are important usage considerations.

QReplication log filtering

IFI 306 log filtering is targeted at QReplication, but any replication product reading the DB2 log will benefit from this enhancement. This reduces the cost of IFI log read by allowed qualification of the objects being replicated by DBID/PSID. There are additional benefits if the objects are compressed. This enhancement moves filtering from Qreplication capture task to the DB2 engine, with potential for significant reduction in the number of log records replicated. To be able to exploit this feature, IBM InfoSphere Data Replication Q Replication or Change Data Capture 10.2.1 is required. Other replication products will also need modification to be able to benefit as well.

Archive transparency

Archive transparency is a useful new feature, which allows data that has been archived from the base table to a history table—either automatically by DB2, or by a user application—to be accessed via UNION ALL. This is dependent on the setting of the SYSIBMADM.GET_ARCHIVE global variable, without having to modify the application program.

If you are planning to use this feature, you need to examine the additional cost of ORDER BY sort when accessing the archive very carefully. When the application only fetches a limited number of rows from the result set, then the cost can increase significantly if the archive is also accessed. This feature will typically be used by customers selectively on a case by case basis.

Optimizer enhancements

The optimizer delivers a number of important enhancements, providing improved performance for legacy application programs. There is a better chance of achieving matching index scan, and there is no need to rewrite SQL to get most of the improvements. However, it is still important to choose the right data type in your application program to avoid implicit casting, and it is still important to run RUNSTATS.

GROUP BY grouping sets

GROUP BY grouping sets is an important feature for data analysis, using the CUBE and ROLLUP functions delivered in DB2 11. All processing is performed in a single pass over the table, but there are some performance differences compared to the old GROUP BY with the same result set. Consider the following:

```
SELECT C1, COUNT(*) FROM T1 GROUP BY C1
```

In this case, sort is not performed if the access path uses an index with leading column ‘C1’.

However, in the following case, a sort is always performed:

```
SELECT C1, COUNT(*) FROM T1 GROUP BY GROUPING SETS ((C1))
```

Extended LRBA/LRSN

This is one of the most significant changes in DB2 11, and if you are considering using it, there a number of implications for DB2 11 CM and DB2 10 NFM. To understand those, you first need to know about the extended LRBA/LRSN (LRBA, or log relative byte address, is often used today instead of the more traditional RBA acronym).

Before DB2 11, the LRBA/LRSN used a 6-byte format: x'LLLLLLLLLLLL'. The new 10-byte extended format LRBA/LRSN has addressing capacity of 1 yottabyte or 2**80—an incredibly large number. The 10-byte extended format LRSN with DB2 11 is in the form x'00LLLLLLLLLLLL000000,' being extended on the right by three bytes to provide much finer timestamp granularity, and on the left by one byte to provide over 30,000 years of capacity. The 10-byte extended format LRBA with DB2 11 is in the form x'00000000RRRRRRRRRRRR,' being extended on the left by four bytes.

The LRBA/LRSN is recorded and found in a number of places:

- In the DB2 catalog: SYSCOPY, SYSxxxxPART …
- In the DB2 directory: SYSUTILX, SYSLGRNX …
- In the BSDS: pointer, active and archive log values ...
- In the DB2 logs: active and archive logs
- In DB2 page-sets: the catalog and directory, and all user-page-sets.

Given the large number of places that the LRBA/LRSN is stored and accessed, there are some important considerations.

DB2 11 CM

DB2 internal coding deals with 10-byte extended format LRBA/LRSN values only. The LRSN in utility output is shown in 10-byte extended format with precision '000000'—that is, extended on the right by three bytes of zeroes—except for the QUIESCE utility, which externalizes the LRSN in 10-byte extended format with precision 'nnnnnn.' The RECOVER utility handles 10-byte extended format LRBA/LRSN input.

Column 'RBA_FORMAT' in SYSIBM.SYSxxxPART is set to 'B' for newly defined objects, or objects which have been reorganized or loaded with the REPLACE option. Possible values for RBA_FORMAT are 'B,' blank, 'U,' and 'E.'

DB2 11 CM / DB2 10 NFM Coexistence in Data Sharing

There is full toleration for the 10-byte extended format LRBA/LRSN value as input to the RECOVER utility. Sanity checks are included for 'wrongly used 6-byte format LRBA/LRSN.'

DB2 11 NFM

During migration to DB2 11 NFM via DSNTIJEN, catalog and directory table LRBA/LRSN columns are altered to the 10-byte extended format. SYSIBM.SYSLGRNX entries are now stored as 10-byte extended format LRBA/LRSN values.

SYSIBM.SYSCOPY is updated to convert all LRBA/LRSN values for existing data to the 10-byte extended format with leading byte '00,' precision '000000' for LRSN values, and right justified right with leading '00000000' for LRBA values. New data is stored in 10-byte extended format with precision 'nnnnnn.'

LRBA/LRSN values for all utilities are now in the 10-byte extended format, but LRBA/LRSN values are still written to the DB2 logs and to DB2 page-sets in the 6-byte format.

The BSDS can only be converted to the 10-byte extended format LRBA/LRSN in NFM, via job DSNJCNVT.
Be aware that once the BSDS has been converted to the new format, there is no way to convert back to the 6-byte format LRBA/LRSN. Once this happens, LRBA/LRSN values are written to DB2 logs of the subject DB2 member in the 10-byte extended format with precision 'nnnnnn.'

However, LRBA/LRSN values are still written to DB2 page-sets in the 6-byte format. To use the extended format, page-set conversion has to be done, although you can convert in either direction (10 to 6 or 6 to 10-byte). This means that, until a page-set is converted, LRSN Spin can still happen. DSN1LOGP and REPORT RECOVER output will show the 10-byte extended format LRBA/LRSN, even though the 10-byte format is not externalized to the page-sets. DSN1PRNT will still report the 6-byte format for unconverted page-sets. Conversion of the page-sets can be done after entry to V11 NFM, in piecemeal fashion over an extended period of time.

The catalog and directory page-sets can be converted to 'extended format,' once in NFM, via REORG. This can be done at any time, regardless of the BSDS and user page-set formats. Once this is done, LRBA/LRSN values are written to the converted page-sets in the 10-byte extended format: the LRSN has precision 'nnnnnn,' if the update is done on a DB2 member with a 10-byte extended format BSDS, and precision '000000,' if the update in done in a member with a 6-byte format BSDS. Column 'RBA_FORMAT' in SYSIBM. SYSxxxPART is updated to 'E' for the converted objects. Be aware that LRSN Spin could still happen for a DB2 member with a 6-byte format BSDS. The catalog and directory can be converted back to the 6-byte format, if needed at part level.

Just as for the catalog and directory, you can reorganize user page-sets to the extended format LRBA/LRSN once in NFM. This can be done whenever you want, regardless of the BSDS or catalog and directory page-set formats. From this point onwards, LRBA/LRSN values are written to the converted page-sets in the 10-byte extended format: the LRSN has precision 'nnnnnn,' if the update is done on a DB2 member with a 10-byte extended format BSDS, and precision '000000,' if the update is done in a member with a 6-byte format BSDS. Column 'RBA_FORMAT' in SYSIBM. SYSxxxPART is updated to 'E' for the converted object. Again, be aware that LRSN Spin could still happen for a DB2 member with a 6-byte format BSDS. As with the catalog and directory, user page-sets can be converted back to the 6-byte format.

User page-set conversion is done using REORG TABLESPACE or INDEX, LOAD REPLACE or REBUILD INDEX with the 'RBALRSN_CONVERSION EXTENDED' option, or if ZPARM OBJECT_ CONVERTED is set to EXTENDED. Be aware that RECOVER TOCOPY using a 6-byte image copy can reset the format back to 'basic.'

ESP-inspired enhancements

ESP customer experience and feedback resulted in the delivery of several enhancements to improve usability:

- DSNJCNVT is prevented from converting the DB2 10 NFM BSDS to extended format.
- DB2 10 NFM supports 10-byte extended format input.
- DB2 performs sanity checks to guard against invalid LRSN values — 6-byte LRSN values with a leading byte of zeros — to prevent PIT recoveries using a bad LRBA/LRSN from failing. Instead, RC=8 is returned in the UTILINIT phase.
- Sanity check also performed in DB2 10 NFM when coexisting with DB2 11 CM.
- Support is provided for a 'NOBASIC' value for the OBJECT_CONVERSION ZPARM to prevent page-sets already in extended format from being converted back to basic format. This also implies 'EXTENDED' and the catalog column for the object is <> 'E' (that is, conversion to the extended format is the default behavior).
- LRSN values are added to the archive log information in REPORT RECOVERY utility output.
- A technical white paper is being produced to explain about '6/10-byte LRBA/LRSN handling.'
- There have been several enhancements to the DB2 11 product documentation.

Recommended best practice migration strategy

Because there is a lot of flexibility in the way conversion to the 10-byte extended LRBA/LRSN can be done, many customers have asked for guidance as to the recommended best practice migration strategy, all the way from DB2 10 NFM to DB2 11 NFM, including conversion to the 10-byte extended LRBA/LRSN. This can be summarized at a high-level as follows:

1. Run the pre-migration jobs and other steps to clean-up DB2 10 NFM
2. Migrate to DB2 11 CM
3. Migrate to DB2 11 NFM
4. Convert ALL BSDS of your data sharing group within 'n' weekends
5. Reorg ALL directory and catalog page-sets to convert to the extended LRBA/LRSN format
6. Set the OBJECT_CREATE and UTILITY_CONVERSION ZPARMs to EXTENDED:
   - New objects will be created in 10-byte extended format
   - REORG, LOAD REPLACE and REBUILD will convert user objects to extended format without the need to change utility control statements
7. Reorg all objects to convert them to the extended LRBA/LRSN format by executing your normal REORG jobs or some additional jobs:
   - Perform a regular check for progress in this task by selecting rows where RBA_FORMAT = 'E' in SYSIBM.SYSxxxxPART
8. Once all objects have been converted, set the OBJECT _CONVERSION ZPARM to NOBASIC

Converting a 10-byte LRSN to a Timestamp

Customers are accustomed to converting 6-byte LRSN values to timestamps, but are not familiar with doing so with a 10-byte LRSN. This section reviews the methods used by customers, then describes how to convert an extended LRSN in DB2 11 CM and DB2 11 NFM.

With DB2 10 NFM or less, use the TIMESTAMP function. The LRSN format is a 6-byte format, wherever used; for example, ‘CBE2B5955DCF.’ This can be converted by:

```
SELECT TIMESTAMP(X’CBE2B5955DCF ‘ || X’0000’) FROM …
```

The practice in DB2 11 CM is very much the same. There are two possible LRSN formats: a 6-byte format in the logs, the catalog and directory, and in data pages, e.g. ‘CBE2B5955DCF’; and a 10-byte format in all message and utility output (except for DSN1PRNT), such as ‘00CBE2B5955DCF086C00.’ In both cases, this should be converted by:

```
SELECT TIMESTAMP(X’CBE2B5955DCF ‘ || X’0000’) FROM …
```

Note the function requires an 8-byte string, hence the truncation of the 10-byte value.

Once in DB2 11 NFM, you should still use the TIMESTAMP function, but there are some subtle differences, and two use cases:

In the first use case, there are two possible LRSN-formats: a 6-byte format for non-converted data pages, as reported by DSN1PRNT, such as ‘CBE2B5955DCF’; and a 10-byte format in the catalog and directory and in all message and utility output, such as ‘00CBE2B5955DCF086C00.’ This can be converted by:

```
SELECT TIMESTAMP(X’CBE2B5955DCF ‘ || X’0000’) FROM …
```

A 6-byte LRSN can be used by ‘cut and paste,’ and a 10-byte LRSN can be used, if the first 2-digits are cut and digits 3 to 14 are used. However, the first two digits must be‘00’ otherwise this conversion is NOT usable.

You can also use:

```
SELECT TIMESTAMP(BX’CBE2B5955DCF0000’) FROM …
```

or

```
SELECT TIMESTAMP(BX’CBE2B5955DCF086C’) FROM …
```

A 6-byte LRSN can be used by ‘cut and paste’ and padded with ‘0000’ at the right (the first example). A 10-byte LRSN can be used, if first 2-digits are cut and digits 3 to 18 are used (the second example), but only if first two digits are ‘00’ otherwise this conversion is NOT usable.

You have another option in DB2 11 NFM, using the ‘binary hex’ function and APPLCOMPAT(V11R1) behavior:

```
SELECT TIMESTAMP(BX’00CBE2B5955DCF086C00000000000000’)
FROM ...
```

A 6-byte LRSN can be used by ‘cut and paste,’ padded with ‘00’ on the left, and padded with ‘000000000000000000’ at the right. A 10-byte LRSN can be used by ‘cut and paste’ and right padded with ‘000000000000.’ BX’ can be replaced by BINARY(X’ or VARBINARY(X’ if desired. This converts a 10-byte LRSN to a timestamp. It works well, but your package needs to be bound with APPLCOMPAT(V11R1).

Other performance recommendations

There are several other important performance recommendations that you should take into account when migrating to DB2 11.

Make sure that HVCOMMON—64-bit common—configured in IEASYSxx is large enough to accommodate the log output buffer, which now uses 1MB page frames. In practice, this means you should configure additional storage in the 1MB LFAREA (IEASYSxx) for maximum benefit.

Remember that LRSN spin avoidance requires that both the BSDS and the user objects are converted to the expanded LRBA/LRSN in NFM. If this is an issue for you, then you should prioritize converting the objects which are affected by LRSN spin for conversion. Once the conversion process is underway, you should monitor log I/O performance, as the log record size increases. The increase is dependent on a number of factors, including row length, the number of columns updated, column length and so on. Customers have observed an increase in log record size from 3 percent to 40 percent following BSDS conversion.

It is essential to make sure there is enough zIIP capacity available before migrating to DB2 11 CM. zIIP 'Help Function' IIPHONORPRIORITY should be set to YES in case there is a shortage of zIIP capacity. You should continue to monitor zIIP capacity thereafter, focusing on zIIP redirect back to the general-purpose processors, as this is a more important measure than zIIP utilization.

In DB2 9, re-classification of sequential buffers read in by prefetch to random if they were accessed by a random GETPAGE, was discontinued. This change is reversed in DB2 11, so pages which have been prefetched are again reclassified as random after a random GETPAGE. This means you might need to re-evaluate the VPSEQT setting for some local bufferpools where it has been increased because of the change introduced in DB2 9.

There are two other changes in the area of buffer management. Previously, pages read in by the COPY utility were managed according to the MRU (most recently used) algorithm, to avoid flooding the bufferpool with pages read by COPY and damaging the hit ratio. This is now used more extensively; for example, pages read by the UNLOAD utility, the unload phase of REORG, and the unload phase of REBUILD are all now managed using the MRU algorithm.

Secondly, using large frames to back bufferpools was tied to page-fixing the buffers to save CPU by avoiding page-fix and page-free for data-set and coupling facility I/O. DB2 11 enables more granular control by introducing the new FRAMESIZE parameter, which is independent from PGFIX parameter, and allows you to specify a page frame of 4KB, 1MB or 2GB.

Performance Summary

When estimating what kind of performance improvement you can expect in DB2 11, you have to take into account a number of factors. DB2 11 certainly presents an opportunity for improved performance for legacy application programs, but your mileage will vary based on the SQL profile of your application workload as the benefits introduced by certain features are specific to certain workloads. While immense CPU savings were observed for some workloads, highly optimized static SQL and/or simple SQL may not see much benefit.

You should expect to see more benefit for a workload characterized by more complex SQL, that is, SQL which does more than read a single row by primary key. Therefore, it is important than you do not sell (or buy) the CPU savings before they have been measured and validated for your particular workload over a period of time.

Other enhancements

The final section of this document discusses two other enhancements which will be of interest to some customers, removing package security vulnerabilities, and archive transparency.

Remove package security vulnerabilities

To understand what this enhancement offers, it's useful to describe the problem use case. Consider the case where each main application routine has its own plan, and where the name of the program and the plan are the same. All packages are bound into a single collection, and each plan is bound with PKLIST(col.*). If the EXECUTE privilege is granted on one plan, the authid/user with that privilege can effectively run any main application program.

DB2 11 solves this dilemma by introducing a new BIND PLAN option, PROGAUTH, which is supported by a new Catalog table, SYSIBM.DSNPROGAUTH. To ensure that a main program 'M' can only be executed with plan 'P', you need to insert a row into SYSIBM.DSNPROGAUTH with column PROGNAME set to 'M', PLANNAME set to 'P', and ENABLED set to 'Y'. To make this specification, you then need to bind plan P with the new option PROGAUTH(ENABLE).

Archive transparency

Archive transparency provides the capability to create an archive (or history) table and connect the base table to the archive table. This is done using the ALTER TABLE DDL statement, using the ENABLE ARCHIVE clause. This requires that the archive table and base table have exactly the same columns. No additional columns are allowed, such as archive-timestamp.

If you set new SYSIBMADM.MOVE_TO_ARCHIVE global variable to either 'Y' or 'E,' then DB2 automatically moves deleted rows to the archive table. If it is set to 'Y,' then any attempted update to rows in the archive-enabled base table will fail with SQLCODE -20555. On the other hand, if it is set to 'E,' the attempted update will work for active rows in the base table, but not for rows in the archive table. In either case, deletion of active rows in the base table will result in those rows being inserted into the archive-table.

If the SYSIBMADM.MOVE_TO_ARCHIVE global variable is set to 'N,' then any rows deleted from the base table are lost. As 'N' is the default, it's important to check that global variable is set to 'Y' or 'E' if you want DB2 to automatically move deleted base table rows to the archive table.

Another new global variable, SYSIBMADM.GET_ARCHIVE, must be set to 'Y' for a query to search both the rows from the base table and the archive table. Remember that any update only applies to active rows in the base table, so any subsequent query might see updated rows from the base table, and not-updated rows from the archive table.

A new package BIND option is available, ARCHIVESENSITIVE (YES|NO), with NO being the default. Again, this only affects whether or not rows are read from the archive table as well as the base table. Deleted rows are only moved to the archive table if the MOVE_TO_ARCHIVE global variable is set correctly.

If you run REORG DISCARD on the base-table, REORG generates a LOAD statement that can be used to load the discarded rows into the archive table. The DISCARD dataset can then be used as input to the LOAD utility, and in most cases this is the right course of action.

There are two important considerations. First, dynamic scrollable cursors cannot be used with archive-enabled tables. Second, the package owner must have the WRITE privilege for the respective global variables to be able to set them.

For more information

To learn more about the IBM DB2 11 for z/OS, please contact your IBM representative or IBM Business Partner, or visit: **ibm.com**/software/data/db2/zos/family/

Additionally, IBM Global Financing can help you acquire the IT solutions that your business needs in the most cost-effective and strategic way possible. We'll partner with credit-qualified clients to customize an IT financing solution to suit your business goals, enable effective cash management, and improve your total cost of ownership. IBM Global Financing is your smartest choice to fund critical IT investments and propel your business forward. For more information, visit: **ibm.com**/financing.

Author contact information

John Campbell, *Distinguished Engineer*
IBM DB2 for z/OS Development
campbelj@uk.ibm.com

Gareth Jones
IBM DB2 for z/OS Development
jonesgth@uk.ibm.com

DB2 11 Resources

A number of resources are available for you to use when migrating to DB2 11, or when exploiting the performance benefits.

- The IBM Information Center / Knowledge Center
- The DB2 11 Technical Overview Redbook (SG24-8180)
- DB2 11 links: **ibm.com**/software/data/db2/zos/family/db211/
  - Links to the DB2 11 Announcement Letter, to webcasts and customer case studies
  - Whitepaper: "DB2 11 for z/OS: Unmatched Efficiency for Big Data and Analytics"
  - Whitepaper: "How DB2 11 for z/OS Can Help Reduce Total Cost of Ownership"
- DB2 11 Migration Planning Workshop at **ibm.com**/IlJxw8

IBM Corporation
Software Group
Route 100
Somers, NY 10589

Produced in the United States of America
August 2014

The content in this document (including currency OR pricing references which exclude applicable taxes) is current as of the initial date of publication and may be changed by IBM at any time. IBM Business Partners set their own prices, which may vary. Not all offerings are available in every country in which IBM operates.

The client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions. It is the user's responsibility to evaluate and verify the operation of any other products or programs with IBM products and programs. THE INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT ANY WARRANTY, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NONINFRINGEMENT. IBM products are warranted according to the terms and conditions of the agreements under which they are provided.

The client is responsible for ensuring compliance with laws and regulations applicable to it. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the client is in compliance with any law or regulation. Statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

[1] The byte units are: Kilobyte, Megabyte, Gigabyte, Terabyte, Petabyte, Exabyte, Zettabyte, Yottabyte

♲ Please Recycle