IBM Software Group

DB2 for z/OS V8
… are you done yet?

IBM Information Management software

John J. Campbell
Distinguished Engineer
DB2 for z/OS Development
Email: CampbelJ@uk.ibm.com

ON DEMAND BUSINESS™

Topics

- Introduction
- Address Some Myths About V8
- Upgrade to V8 as quickly but as safely as possible
  - Secrets To Successful Migration
  - Pre-Migration Catalog Migration Testing
  - Quick Hits
  - Migration Plan Recommendations
- Value Proposition with V8
  - Improving performance and scalability
  - Prepare for and recover from logical data corruption
  - Better management of very large databases
  - Application Portability

DB2 for z/OS Strategic Direction

- **Continue to Evolve as "The" Secure Enterprise Data Hub Server**
  - **Centralized Processing of Massive Quantities of Data**
  - **DB2 Family Compatibility**
  - **OBDC/CLI, JDBC, SQLJ, Unicode, SOA Stored Procedure, Java, …**
- **Premier Use of Parallel Technology and z/OS – IBM System Integration**
  - **Data Sharing**
  - **Continuous Availability**
  - **Incremental Growth**
  - **Work Load Balancing**
- **Improved Application Enablement Through Extensive SQL Functionality**
  - **Native XML support**
  - **Object-Oriented Extensions - User defined functions and types**
  - **Rules Driven Integrity - referential integrity and table constraints**
  - **Vendor Enabling and Portability**
- **Continual Performance Improvement**
  - **Optimizer Technology**
  - **Transaction, Query, Batch, Utility Concurrency with Application Processing**
  - **Utilities and Tools Focus**

DB2 for z/OS V8: biggest release ever

- Online schema change
- Backup and Recovery enhancements
- Automated space management
- Fast and automatic cached statements invalidation
- Transaction/end-user based accounting and workload management
- Enhancing RUNSTATS with DSTATS
- Long-running, non-committing readers detection
- Lock escalation reporting improvement
- Providing cached statement id in IFCID 124
- Improved LPL recovery
- DRDA and JCC tracing and diagnostics …
- Lifting virtual storage constraints
- Piece wise LOB insert
- More stage 1 predicates
- Index only access path for VARCHARs
- Fast retrieval of the most recent value
- Eliminating lock contention on special purpose tables
- Option to release locks at cursor close
- Allowing updating partitioning key column without partition locks
- Disassociating clustering from partitioning key
- DPSI
- Up to 4096 partitions
- Data sharing improvements
- DRDA performance. . .
- Array inserts and fetches
- Sparse indexes
- Reducing negative impact of host variables at access path selection REOPT(ONCE)
- Transparent ROWID for tables containing LOBs
- Full Unicode support
- Lifting database object names length limits
- Up to 255 tables in FROM
- Materialized Query Tables (a.k.a. Automatic Summary Tables)
- Common Table Expressions
- Recursive SQL
- Multiple DISTINCTs
- DB2 Connect 64-bit client for Linux on zSeries
- Allowing comments in dynamic SQL …

DB2 for z/OS V8: major theme is TCO reduction

- Online schema change
- Backup and Recovery enhancements
- Automated space management
- Fast and automatic cached statements invalidation
- Transaction/end-user based accounting and workload management
- Enhancing RUNSTATS with DSTATS
- Long-running, non-committing readers detection
- Lock escalation reporting improvement
- Providing cached statement id in IFCID 124
- Improved LPL recovery
- DRDA and JCC tracing and diagnostics …
- Lifting virtual storage constraints
- Piece wise LOB insert
- More stage 1 predicates
- Index only access path for VARCHARs
- Fast retrieval of the most recent value
- Eliminating lock contention on special purpose tables
- Option to release locks at cursor close
- Allowing updating partitioning key column without partition locks
- Disassociating clustering from partitioning key
- DPSI
- Up to 4096 partitions
- Data sharing improvements
- DRDA performance. . .
- Array inserts and fetches
- Sparse indexes
- Reducing negative impact of host variables at access path selection REOPT(ONCE)
- Transparent ROWID for tables containing LOBs
- Full Unicode support
- Lifting database object names length limits
- Up to 255 tables in FROM
- Materialized Query Tables (a.k.a. Automatic Summary Tables)
- Common Table Expressions
- Recursive SQL
- Multiple DISTINCTs
- DB2 Connect 64-bit client for Linux on zSeries
- Allowing comments in dynamic SQL …

DB2 9 for z/OS: packed with new features

- Native SQL Procedural Language
- Optimistic locking
- XML support in DB2 engine
- TRUNCATE statement
- MERGE statement
- INSTEAD OF triggers
- SELECT FROM UPDATE / DELETE / MERGE
- ORDER BY and FETCH FIRST in subselect
- IPv6 and SSL support
- Modify early code without requiring an IPL
- APPEND option for inserts
- Relief for sequential key insert
- LOB performance and scalability
- INDEX on expression
- Index page size greater than 4K
- Utilities CPU reduction
- DB2 Trace filtering …

- Point in time recover improvements
- Faster restart of data sharing
- Decimal Floating Number, BIGINT, VARBINARY
- Reordered row format
- CLONE Table: fast replacement of one table with another
- Renaming column, index, and schema
- Table space that can add partitions, as needed for growth
- Improve ability to create an index online and rebuild it
- Online reorganization with no BUILD2 phase
- Parallel unload and reload during REORG
- Consistent recover of a single object
- Converged TEMP space
- Automatic objects creation
- 64-bit exploitation by DDF …

- Automatic collection of performance data for long queries
- Autonomic access path reoptimization
- Automatic query monitoring for most frequent/expensive queries
- Optimization Service Center
- Index compression
- Database ROLEs
- Trusted security context
- Enhancing volume based backup/recovery: object-level recoveries
- Cross query block optimization
- Histogram statistics exploitation
- Generalized sparse index and in-memory data cache method
- Dynamic index ANDing for Star Schema
- EXCEPT and INTERSECT
- New process qualification options for RLF …. .

DB2 9 for z/OS: again TCO reduction major theme

- Native SQL Procedural Language
- Optimistic locking
- XML support in DB2 engine
- TRUNCATE statement
- MERGE statement
- INSTEAD OF triggers
- SELECT FROM UPDATE / DELETE / MERGE
- ORDER BY and FETCH FIRST in subselect
- IPv6 and SSL support
- Modify early code without requiring an IPL
- APPEND option for inserts
- Relief for sequential key insert
- LOB performance and scalability
- INDEX on expression
- Index page size greater than 4K
- Utilities CPU reduction
- DB2 Trace filtering …
- Point in time recover improvements
- Faster restart of data sharing
- Decimal Floating Number, BIGINT, VARBINARY
- Reordered row format
- CLONE Table: fast replacement of one table with another
- Renaming column, index, and schema
- Table space that can add partitions, as needed for growth
- Improve ability to create an index online and rebuild it
- Online reorganization with no BUILD2 phase
- Parallel unload and reload during REORG
- Consistent recover of a single object
- Converged TEMP space
- Automatic objects creation
- 64-bit exploitation by DDF …
- Automatic collection of performance data for long queries
- Autonomic access path reoptimization
- Automatic query monitoring for most frequent/expensive queries
- Optimization Service Center
- Index compression
- Database ROLEs
- Trusted security context
- Enhancing volume based backup/recovery: object-level recoveries
- Cross query block optimization
- Histogram statistics exploitation
- Generalized sparse index and in-memory data cache method
- Dynamic index ANDing for Star Schema
- EXCEPT and INTERSECT
- New process qualification options for RLF …. .

Address Some Myths About V8

- V8 is not a complete re-write of engine, significant re-engineering in select places
- Compatibility Mode is not “No New Function Mode”!
- V8 is not two releases in one: Compatibility Mode (CM), New Function Mode (NFM)
- IBM provides first class support for Compatibility Mode
- Can stay in Compatibility Mode as long as you like
- No need to convert all your application data to Unicode
- DFSORT is really required for V8 Utilities Suite (no cost)
- Large number of secondary extents is not necessarily bad for performance
- 5-10% is typical increase in CPU resource consumption assumes zero exploitation
- Tuning options available to mitigate CPU regression
- Significant opportunity for both performance and scalability improvements
- No need to fear migration with proper planning and testing, follow step by step approach followed by other successful customers

Secrets To Successful Migration

- Spend time and effort in testing to keep fire away from production
- Clean up code page issues, analyzing all inputs and outputs, and systematically test to avoid data corruption
- Perform Pre-migration Catalog Migration Testing on clone image
- Perform systematic testing of release fallback toleration
- Perform batch regression testing
- Monitor and control CPU, virtual and real storage usage
  - Before, during, after
  - Build V7 performance baseline prior to V8 CM
  - After migrating to V8 CM
- Use RMF and DB2PE to build performance baseline and monitor

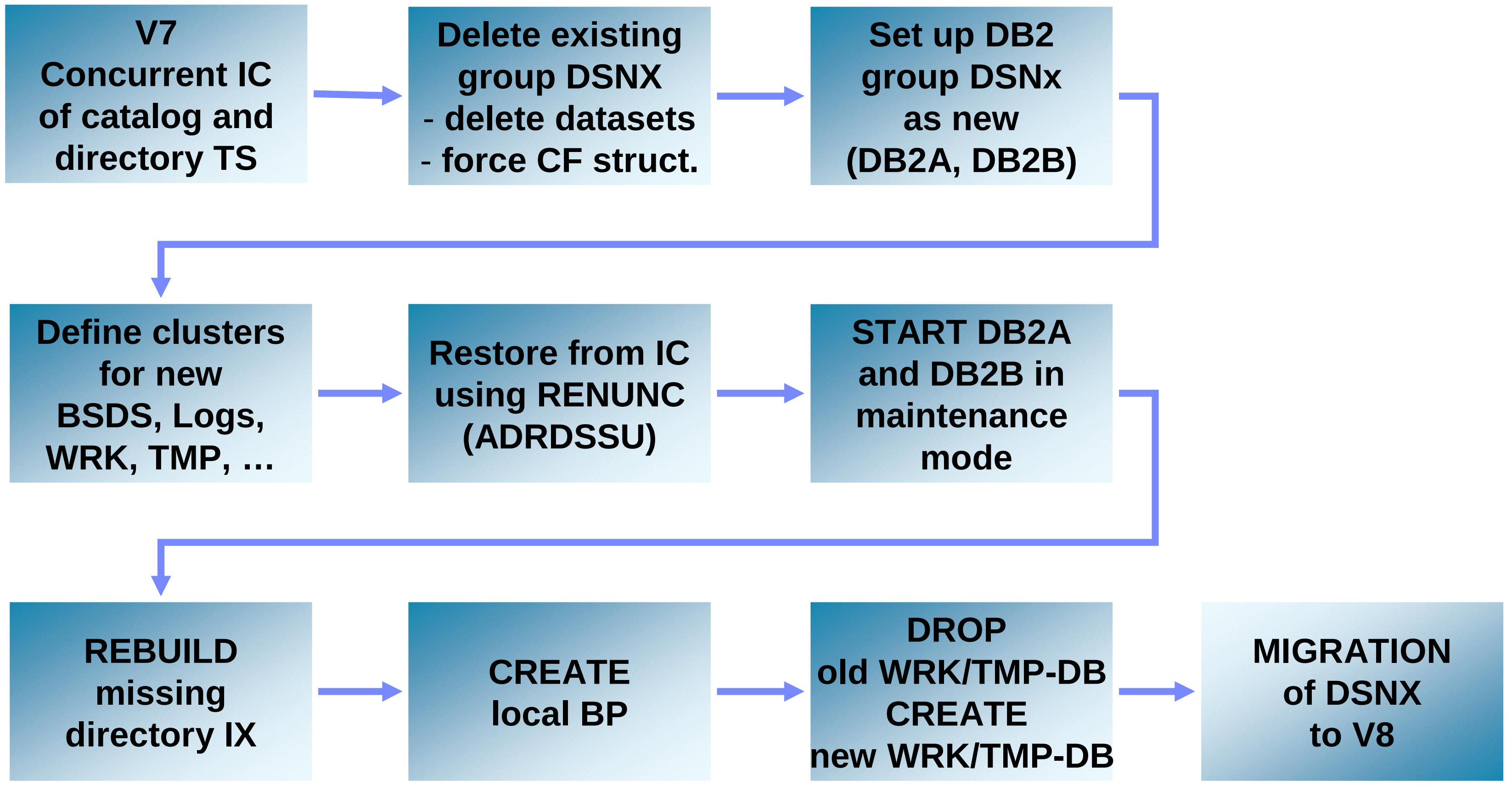
Secrets To Successful Migration …

- If you need data sharing coexistence, keep period reasonable short
- Set zparm ABIND=COEXIST when running in release coexistence
- Run in CM for at least a complete monthly cycle or longer to minimize risk of serious risk after moving on to ENFM
- Schedule an outage for ENFM/NFM and get it out of the way ASAP
- Recommended minimum of DB2 Connect is V8R1 FP10 (V8R2 FP3)
- Maintenance: take advantage of CST/RSU plus Enhanced HOLDDATA
- Rebind all plans which have not been rebound since DB2 V2R3
- Rebind packages and plans which are associated with high volume and performance sensitive applications at/after V8 CM
- Reorganize Catalog and Directory objects under V7 prior to start of V8 migration
- Reorganize Catalog /Directory objects with hash links under V8 CM before entering V8 ENFM

Pre-Migration Catalog Migration Testing

- Clone a production catalog and practice migration on the cloned copy without risk and before migrating the catalog of any productive system
  - Options: Concurrent Copy, Flashcopy, DFSMS Dump, DSNTIJIC
  - Practice migration:
    - V7-> V8 plus CATMAINT
    - V8 CM -> V8 ENFM -> V8 NFM
    - Rebind plans and packages
    - Explain all views
- Possible extend this test to an extended regression test system
- Engage other parties into the migration and pre migration process and testing
  - DBA's, Application developers …

Pre-Migration Catalog Migration Testing …


V7 Concurrent IC of catalog and directory TS
Delete existing group DSNX - delete datasets - force CF struct.
Set up DB2 group DSNx as new (DB2A, DB2B)
Define clusters for new BSDS, Logs, WRK, TMP, …
Restore from IC using RENUNC (ADRDSSU)
START DB2A and DB2B in maintenance mode
REBUILD missing directory IX
CREATE local BP
DROP old WRK/TMP-DB CREATE new WRK/TMP-DB
MIGRATION of DSNX to V8

Quick Hits

- DFSORT and V8 Utilities
  - V8 will only use the DFSORT package: SORT, MERGE functions only
  - Aim is to improve reliability and performance
  - DFSORT code is part of a standard z/OS install
  - DB2 UDB for z/OS Utilities Suite has a license API to use DFSORT
  - DFSORT code must be available via LNKLST/STEPLIB
  - DFSORT 1.5 (z/OS 1.5) is recommended for better performance
  - APAR II14047 is mandatory reading for anyone who is not already using DFSORT
- Console Security and SYSOPR
  - Commands via console no longer come through with authid of SYSOPR
  - zparm SYSOPRn=SYSOPR no longer allows any user to enter commands from console
  - Use new support for RACF Groups and secondary authids

Quick Hits …

- Unicode Catalog
  - Run DSNTIJP8 (V7) to check that there are not multiple CCSIDs within one encoding
  - V8 start up checks for non-zero, valid CCSIDs
    - EBCDIC<->Unicode round trip conversion
  - Must configure Unicode Conversion Services for z/OS
    - Unicode CCSIDs (367,1208,1200) <-> ASCII/EBCDIC CCSIDs
    - Client CCSIDs <-> Unicode CCSIDs (367,1208,1200)
    - Must also add to/from Unicode CCSIDs (367,1208,1200)
      - CCSID 37 for DBRMs provided with DB2 install
      - CCSID 500 for DRDA
      - CCSID 1047 for Unix System Services (USS)
    - Rebuild conversion image, new image picked up by DB2 'on the fly'
  - IBM System z (z890 and z990) and z/OS R4 for best performance

Quick Hits …

- Old COBOL
  - Modified COBOL source generated by V8 Precompiler is not supported by old COBOL compilers
    - OS/VS COBOL has been out of support since roughly 1994
    - VS COBOL II went out of support in 2001
  - Old COBOL load module running in a LE environment will still work
  - Must re-link old COBOL load modules to run in a LE environment for serviceability
  - COBOL V3.2 (EOS Oct 2005), Enterprise COBOL V3.2 or later supports V8
- Dumps and Standalone Dumps
  - Increase MAXSPACE to avoid partial dumps
  - Allow enough REAL/paging space
  - Allow enough DASD
  - Use Striped Volumes for dumps greater than one volume
  - Use Dump Groups so that volumes can be written concurrently

Quick Hits …

- Many zparm defaults have changed
- Still have option of PDS vs PDSE for DSNLOAD
- Bimodal Migration Accommodation is not supported
- ICSF Modules to avoid Nasty Grams - SCSFMOD0 in Link List
- Use ISPF APAR OA07685 to browse Unicode DBRMs
- Tools Certification is important, don't wait to verify
  - http://www.ibm.com/support/docview.wss?rs=434&context=SSZJXP&uid=swg21162152&loc=en_US&cs=utf-8&lang=en+en
- After the V8 migration, every problem becomes your problem
  - Collect SMF data and reports
  - EXPLAINs of key packages
  - Understand system resource utilization before the migration
  - Have prepared standard job stream that exercises the system

Migration Plan Initial Thoughts

- Run in V8 CM for at least one month based on typical application cycle
  - To flush out any implementation, performance, stress conditions
  - Before migrating to V8 ENFM/NFM
- Most customers do not have an operational proving test environment and application regression workloads
  - Regression and stress test to flush out implementation issues
- Do not want to run for an extended period with different code and function levels across 'test' and 'production' environments
- Have clear separation between 'toleration' and 'exploitation' of features in the new release
  - Exploitation project would follow on from toleration project when the new release has been running stable in production for a reasonable period (typically complete one processing month cycle) and there is little chance of fallback to previous release of DB2
  - Exploitation project will focus on a prioritized list of new features offering business value. During the toleration period customer will run with release N-1 of the DB2 Precompiler to prevent new SQL features being used
- Distinguish carefully
  - Data sharing ‘coexistence’
  - ‘Compatibility mode

Migration Plan Recommendations

1. Migrate V7 test data sharing group to V8 CM and hold. If want to roll in the new release member by member, migrate the first 2 members, wait, then roll through the remaining members quickly.
2. After (1) has been proven for extended period, migrate production data sharing group to V8 CM and hold. If want to roll in the new release, migrate the first 2 members, wait, then roll through the remaining members quickly.
3. Run through complete monthly processing cycle in production to flush out implementation/regression issues - can fallback/come forward again if necessary.
4. After (3) is successful, start to migrate V8 test data sharing group towards V8 NFM mode.
5. After (4) is successful, start to migrate V8 production data sharing group towards V8 NFM mode.

CPU Performance Regression - Antidotes

- Dramatic new function delivered in V8
- Typical customer workload regression is expected to be 5 to 10% higher on average assuming NO changes to existing applications or system environment to exploit new V8 features
- However …

CPU Performance Regression – Antidotes …

- Options with significant potential to offset expected increase
  - Use Long Term Page Fix for IO Intensive Bufferpools
  - Turn off SMF 89 Detailed Recording (zparm SMF89=NO)
  - Apply PK28561 to reduce package accounting overhead
    - Overhead incurred is critically dependent on how frequently packages are invoked, how many bufferpools, …
  - If no longer constrained on DBM1 VSTOR below 2GB bar
    - Set zparm CONTSTOR=MINSTOR=NO
  - Implicit Multi Row Fetch for DRDA
  - Rebind of all plans and packages
    - New access paths, re-instate fast column processing, avoid puffing overhead
  - For data sharing customers
    - Rebuild GBPs after last V7 member gone to V8 CM
    - Quiesce data sharing group after entry to V8 NFM to switch to Locking Protocol 2

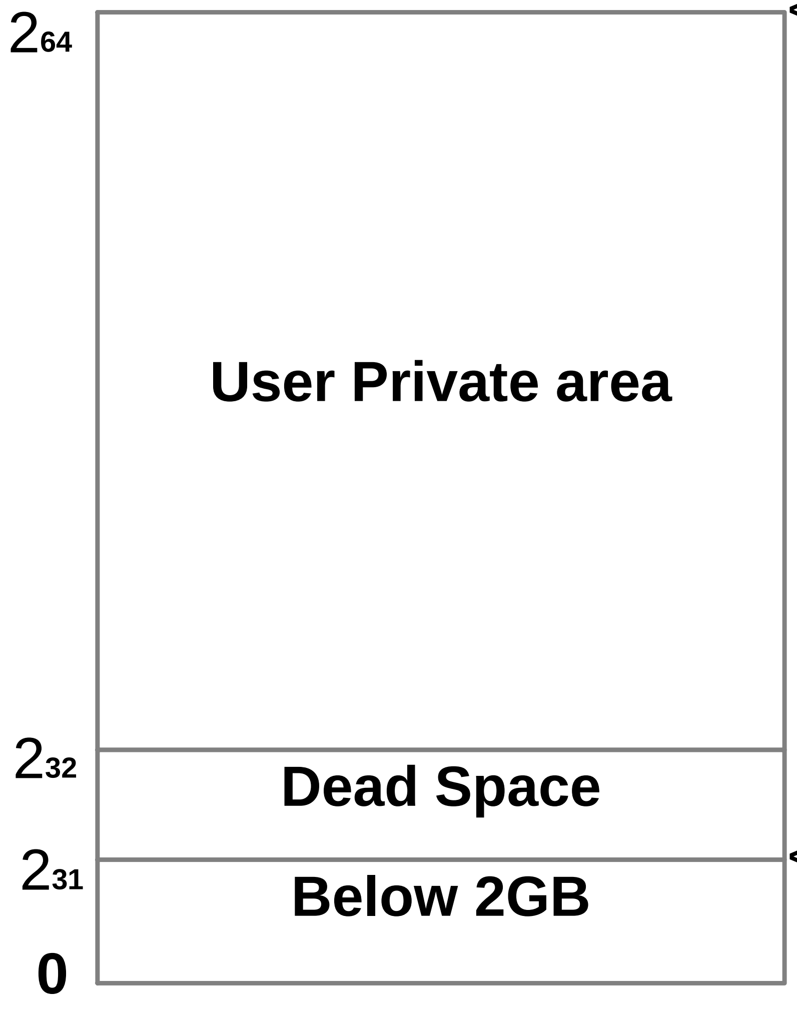
CPU Performance Regression …

- The following types of V7 workloads tend to have less regression:
  - zparms MINSTOR=YES and CONTSTOR=YES
  - Dataspace bufferpool or Hiperpool
  - SMF89 active
  - IRLM PC=YES
  - CURRENTDATA NO
  - BIND RELEASE COMMIT option in Data Sharing
  - Update intensive batch processing in Data Sharing
  - I/O intensive workload

Long Term PGFIX

- Strong recommendations
  - Ensure bufferpool 100% backed by real storage
  - Use Long Term Page Fix by Bufferpool on IO intensive bufferpools to help compensate for CPU performance regression
  - Apply PTFFIX for APAR OA17114 (still open)
    - z/OS Real Storage Manager backs some long term fixed pages in below the 2GB bar real causing IRA400E messages
- Typically results in the range of 0 to 8% improvement
- -ALTER BPOOL(x) PGFIX(YES) plus reallocation of bufferpool
- Page fix all buffers just once
  - Avoids expensive page fix and page free for every IO in 64-bit
- Performance benefit is inversely proportional to the buffer hit ratio
  - Higher the hit ratio, the lower the benefit
- Apply to small buffer pools with lots of IOs e.g., sequential work
- Greatest potential benefit with relatively high IO Intensity
- IO Intensity calculated as (#Pages Read+Written) / #Buffers

DBM1 64 bit Virtual Memory Map (not to scale)



<- 16EB "beam"(?)

- Buffer pools, BM control blocks
- Castout buffers
- RID pool
- EDM DBD cache (OBDs)
- Global dynamic stmt cache
- Sort pool
- Trace tables (Global, Lock, BB)
- Accounting blocks
- Compression dictionaries
- IRLM locks

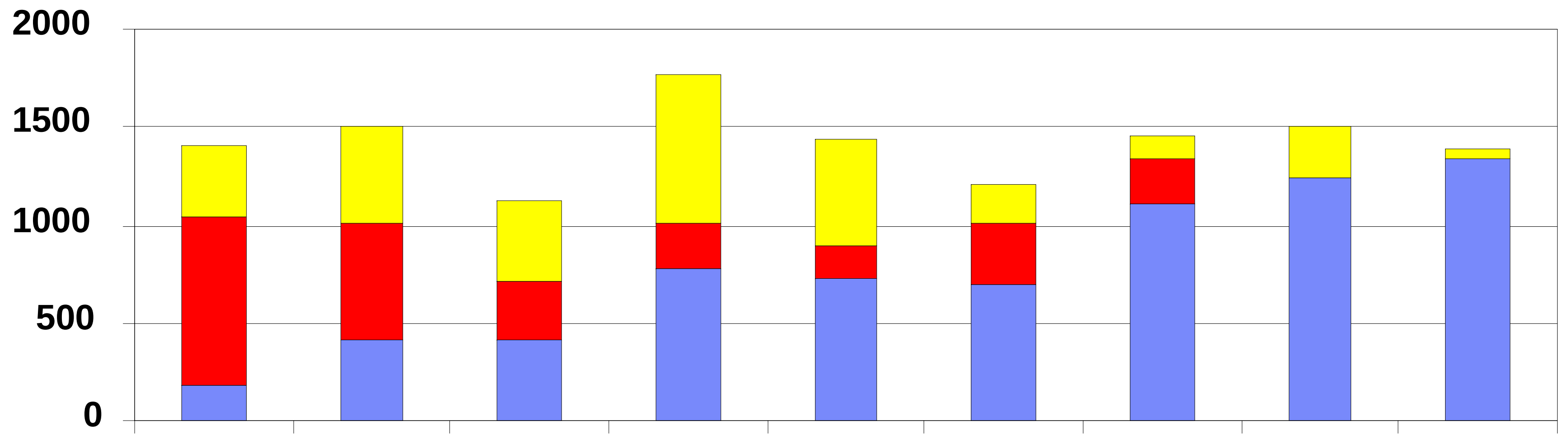
<- 2GB "bar"

- Most thread storage. Local DSC
- Other EDM and RID component

DBM1 64-bit and Thread Storage

- Most of the thread storage stays below the 2GB bar
  - Agent Local, Stack Storage
  - Local Dynamic Statement Cache
  - Expect some regression
- Regression estimates:
  - Thread storage:
    - +40% for system threads
    - +30 to 40% for static SQL
    - +50 to 100% for dynamic SQL
  - Stack storage: +100%
  - Local Dynamic Control Blocks: -75%
  - Local Dynamic Statement Cache: +100%
  - EDM pool: +30 to 50%
  - RID pool: -75%
  - Others: -100%

Potential Value of 64-bit Virtual



Projecting V8 from V7 Statistics Trace

	V7 measured (MB)	V8 estimated (MB)		Notes:
Virtual buffer pool	15	0		0
Buffer pool control blocks	95	0		0
Dataspace lookaside buffer	48	0		0
EDM pool	88	124		+30 to 50%
Compression dictionary	54	0		0
Castout buffers	28	0		0
System thread storage	89	125		+40%
User thread storage	114	217		+30 to 40% static SQL, +50 to 100% dynamic SQL
RDS OP pool	8	0		0
RID pool	78	20		-75%
Pipe Manager subpool	1	1		Same
Local Dynamic Control Blks	70	18		-75%
Local Dynamic Stmts	8	16		+100%
BM/DM trace table	18	9		-50%
Fixed storage	3	3		Same
Stack storage	58	116		+100%
Total	775	649		
% VSCR		16		

64-bit Messages

- 64-bit support will not absolutely eliminate constraint
- But will provide valuable additional relief, which will vary by installation
- Will be able to exploit all available processor storage on latest processor models (current DB2 limit of 1TB)
- May be able to support additional active threads and DBATs ?
- May be able to set zparms CONTSTOR=NO and MINSTOR=NO
- Must have sufficient real storage to fully back increased total virtual storage usage: above and below 2GB bar
- Installations must continue to capacity plan for, monitor, tune and optimize VSTOR usage below 2GB bar

Real Storage Use

- Important subsystems such as DB2 should not be paging to auxiliary storage (DASD)
  - Recommendation to keep paging rates low (near zero)
  - Monitor using RMF Mon III
- Backing rate is dense for 31-bit storage (as before in V7)
- Common misconception
  - Backing rate is low for 64-bit storage (<10%)
- Increase in real storage with V8 is to be expected as control blocks bigger for 64-bit storage
  - Stack 100%
  - System Threads 40%
  - User Threads: 30-40% static, 50-100% dynamic

  plus above the 2GB bar management
- Have observed significant growth in real storage demand in some installations

Real Storage Use …

- Recommendation to apply critical service
  - PK19769
    - Add MVS discard to pool reset
  - OA15666 and PK25427
    - Discard real frames without hitting AVQLOW condition
  - PK21237
    - Drop number of Write Engines back to pre V8 levels
    - Single 64-bit pool for all Buffer Manager engines
  - PK21892
    - Throw away cached stack on thread deallocation
  - PK25326
    - Contract PLOCK engines after use

Real Storage Use …

- Recommendation to estimate real storage requirement
  - Method assumes critical maintenance has been applied
  - Subtract out fixed areas
    - Bufferpools, EDM Pools
  - Below the 2GB bar
    - Assume V=R
    - V7 -> V8
      - Stack +100%
      - User Threads +30-40% static, +50-100% dynamic
      - System Threads +40%
  - Above the 2GB bar
    - Assume 1MB per active thread (pessimistic)
  - Add back the fixed areas

Query Performance

- All practical reasonable cases can now take advantage of index access
- Recommendation to run RUNSTATS with KEYCARD option (not the default)
- RUNSTATS collection of Distribution Statistics can help fix many access path selection problems
- Recommendation to rebinds all plans and packages within the life of any given release
- ‘Dummy Collection’ method (explained on next slide) can be used to provide fallback to old access path assuming no use of SET CURRENT PACKAGESET

Query Performance …

- ‘Dummy Collection’ method
  - Current situation Plan A with Package B in Collection V7COL
  - At some point prior to V8:
    - BIND PACKAGE (V7COL) MEMBER (B) ACTION (ADD) ...
    - BIND PLAN(A) PKLIST(V7COL, …) ACTION(ADD) ...
  - Under V8 using new collection V8COL:
    - BIND PACKAGE (V8COL) COPY (V7COL.B) ACTION (ADD) ...
    - REBIND PLAN (A) PKLIST (V8COL,V7COL, …) ...
  - Program B will now use new Package B in new Collection V8COL ahead of old Package B in old Collection V7COL
  - If the access path changed and the actual run time performance is unacceptable, delete Package B from Collection V8COL:
    - FREE PACKAGE (V8COL.B)
  - Program B will then fallback to using old Package B in original Collection V7COL and fallback to using the old access path

Major Performance Opportunities

- 10 to 100 times improvement possible from
  - Materialized Query Table
  - Stage 1 and indexable predicate for unlike data types
  - Distribution statistics on non-indexed columns
  - Other access path selection enhancements
- 2 to 5 times improvement possible from
  - Multi-row Fetch, cursor Update/Delete, Insert
  - Star join with work file index, in-memory work file, more parallelism
  - DBM1 virtual storage constraint relief
  - Partition Load/Reorg/Rebuild with DPSI

Supersize Bufferpools and ESA Compression

- More use of ESA Compression provided fully backed by real storage
  - Less DASD space
  - Faster sequential scan
  - Potential for better bufferpool hit ratio
    - Reduce sync IO
    - Reduce DB2 Activity Time
    - Fewer threads to maintain same throughput (VSCR)
- Super size bufferpools provided fully backed with large real storage
  - Potential for better bufferpool hit ratio
    - Reduce sync IO
    - Reduce DB2 Activity Time
    - Fewer threads to maintain same throughput (VSCR)

Multi-Row SQL Operations

- Avoids CPU overhead of SQL application interface 'round trip'
- Multi-Row INSERT
  - Up to 29% better for 10 rows
  - Dramatic reduction in network traffic
    - Avoids message send/receive pair for each row
    - Potential for huge elapsed time reduction (up to 8x)
- Multi-Row FETCH
  - Up to 33% better for 10 rows
  - Automatically enabled for DRDA when
    - Read-only
    - Ambiguous cursor with CD(N)

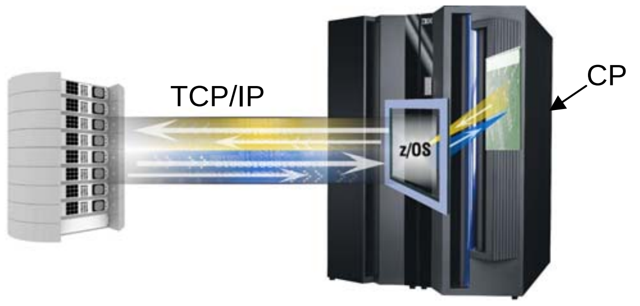
Positioned to Exploit IBM zIIP

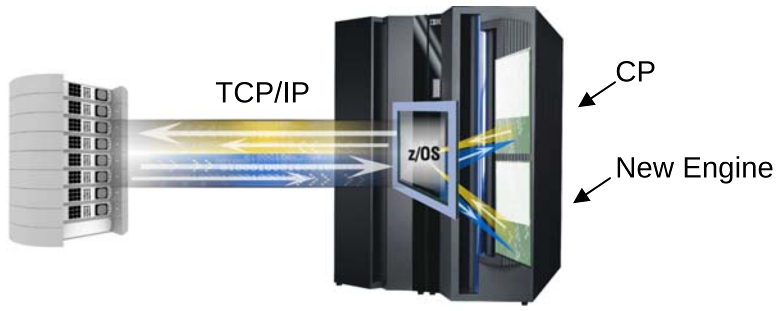
- New specialty engine for the System z9 mainframe designed to help eligible data serving workloads:
  - Improve resource optimization
  - Lower the cost of ownership
- z/OS manages and directs work between the general purpose processor and the zIIP
  - Transparent to DB2 for z/OS V8 applications
  - Number of zIIPs per z9-109 not to exceed number of standard processors
- DB2 for z/OS V8 in any mode will be first exploiter of the zIIP with:
  - System z9 109
  - z/OS 1.6 or later

Types of Workload that may benefit from IBM zIIP

1 -ERP or CRM application serving*

- For applications, running on z/OS, UNIX, Linux, Intel, or Linux on System z, that access DB2 for z/OS V8 on a System z9 109, via DRDA over a TCP/IP connection DB2 gives z/OS the necessary information to have portions of these SQL requests directed to the zIIP





2 - Data warehousing applications*

- Requests that utilize DB2 for z/OS V8 *ALL* parallel queries may have portions of these SQL requests directed to the zIIP when DB2 gives z/OS the necessary information – not just star schema, not just DRDA over TCP/IP, cuts in at 100ms of z9 CPU

3 – Some DB2 for z/OS V8 utilities*

- A portion of DB2 utility functions used to maintain index maintenance structures (example LOAD, REORG, and REBUILD INDEX) typically run during batch, can be redirected to zIIP.

* The zIIP is designed so that a program can work with z/OS to have all or a portion of it's Service Request Block (SRB) enclave work directed to the zIIP. The above types of DB2 V8 work are those executing in SRB enclaves, portions of which can be sent to the zIIP.

Online Schema Evolution (Dynamic ALTER)

- Extend CHAR(n) column lengths
- Change column data types
  - Must remain within domain type (numeric, char, graphic)
  - OK if referenced by a view
  - OK if indexed
  - Immediate index availability if CHAR, VARCHAR, GRAHPIC, VARGRAPHIC
  - Delayed index availability if numeric
    - Index placed in rebuild pending (RBDP) status
    - Availability to index delayed until index is rebuilt
    - But RBDP index will not block dynamic SQL for:
      - Deletes
      - Updates or inserts for non-unique index
      - Queries (APS won't chose a RBDP index)
- Add column to index
  - Index immediately available if column added to table in same transaction
  - Otherwise index put into RBDP status

Online Schema Evolution (Dynamic ALTER) …

- Add partition to the end
  - Extends the limit value
- Rotate partitions
- Automatic rebalancing of partitions during REORG
- DROP partitioning index, or allow create of a partitioned table without a partitioning index
  - "Table based" partitioning
- Change the clustering index
  - Historically, the partitioning index had to be the clustering index
  - Now an NPI can be the clustering index
  - If no explicit clustering index specified
    - The first index created is used for clustering

Data Partitioned Secondary Index

- Overview
  - DPSI = physically partitioned secondary index
  - #parts(DPSI) = #parts(table)
  - Keys in part 'n' of DPSI refer only to rows in part 'n' of table
- Benefits include:
  - More efficient utility processing
  - Higher availability
  - Streamline partition level operations
  - Potential for lower data sharing overhead
  - Potential impact to query performance
- 3 kinds of indexes now:
  - Partitioning Index (PI).
    - As today, except optional in V8 and may or may not be partitioned
  - Data Partitioned Secondary Index (DPSI) - New in V8
  - Non Partitioned Secondary Index (NPSI) - As today's NPI

Up to 4096 Partitions

- Max number of parts raised from 254 to 4096
  - Tablespaces and indexes
  - Tablespace must have LARGE or DSSIZE to go beyond 254 parts
- Max table size remains 16TB
- Dataset naming convention
  - 'Axxx' - partitions 1-999
  - 'Bxxx' - partitions 1000-1999
  - 'Cxxx' - partitions 2000-2999
  - 'Dxxx' - partitions 3000-3999
  - 'Exxx' - partitions 4000-4096
- Max number parts allowed depends on page size and DSSIZE
  - 4K page size, DSSIZE=1GB => 4096 parts allowed, 4TB max table size
  - 4K page size, DSSIZE=64GB => 256 parts allowed, 16TB max table size
- ALTER TS ADD PART adds partitions to the end

System Level Point In Time Recovery

- New HSM construct called a COPYPOOL
  - Named set of SMS storage groups
  - Each DB2 system defines one COPYPOOL for data, one for logs
  - z/OS 1.5
- New DFSMS construct called "copy target" storage group
  - Storage group that's reserved to be target of FlashCopy
- BACKUP SYSTEM utility
  - Invokes HSM to take FlashCopy of database COPYPOOL, and optionally Log COPYPOOL
  - Copy recorded in BSDS
  - Certain activities are quiesced before copy is invoked
    - 32K page writes
    - Dataset extend, create, delete, rename
    - Pseudo close (R/O switching)
    - System checkpoint advancement of restart/redo log points
  - Less disruptive than today's SET LOG SUSPEND

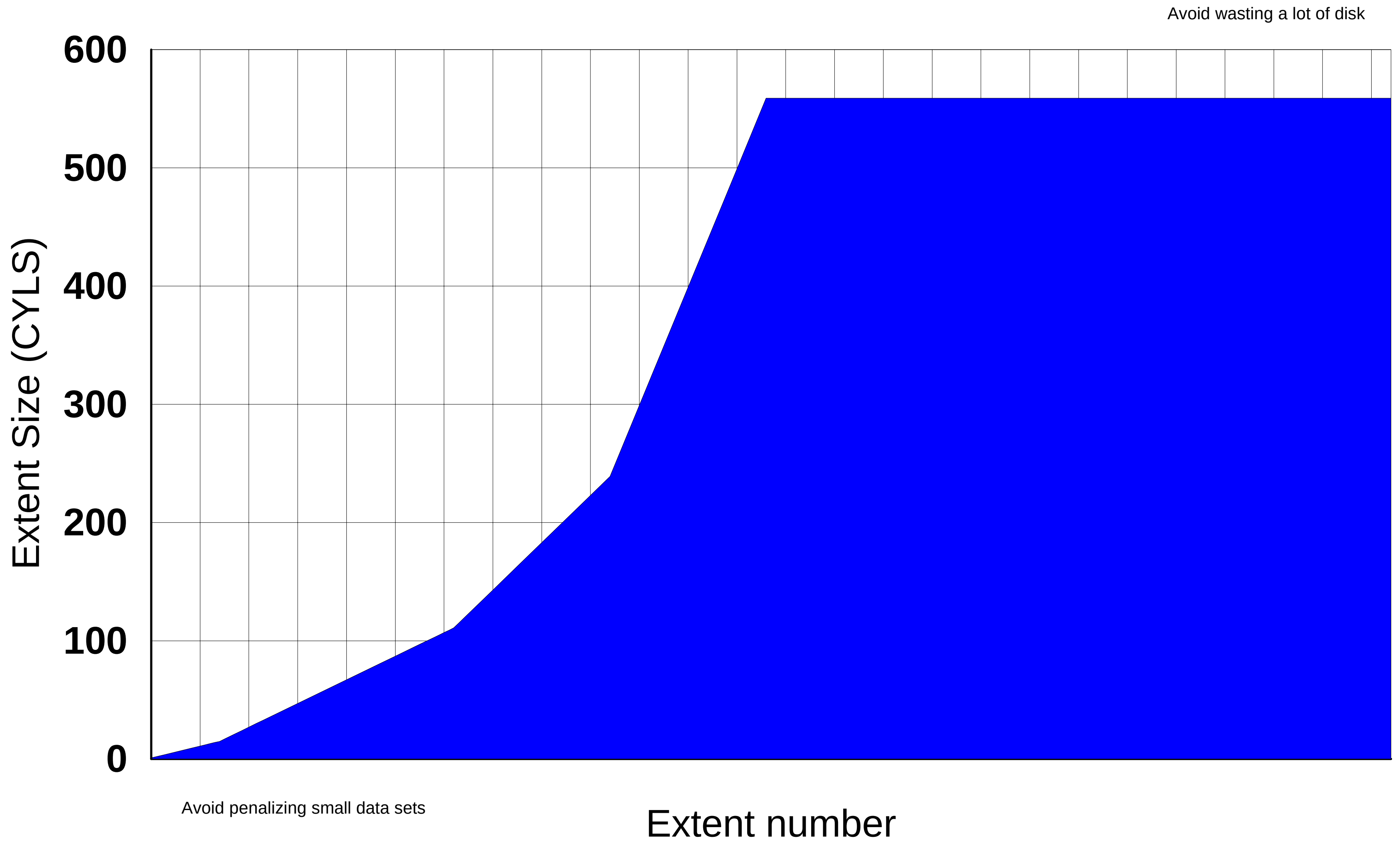
System Level Point In Time Recovery …

- RESTORE SYSTEM
  - Run to restore a DB2 system to a prior point in time
- Procedure to recover system to any arbitrary PIT
  1. Use new type of conditional restart to truncate the logs to the desired time to which to recover and to precondition for next step
  2. Run RESTORE SYSTEM utility
     - Restores the database volumes from the appropriate COPYPOOL backup
     - Handles objects created since restore point
     - Handles LOG NO events
     - Deletes any objects dropped since restore point
     - Handles data sharing or non data sharing
     - Progress is checkpointed, restartable
- Much easier and more efficient process than possible before
- ERP/CRM customers have especially expressed great interest in this

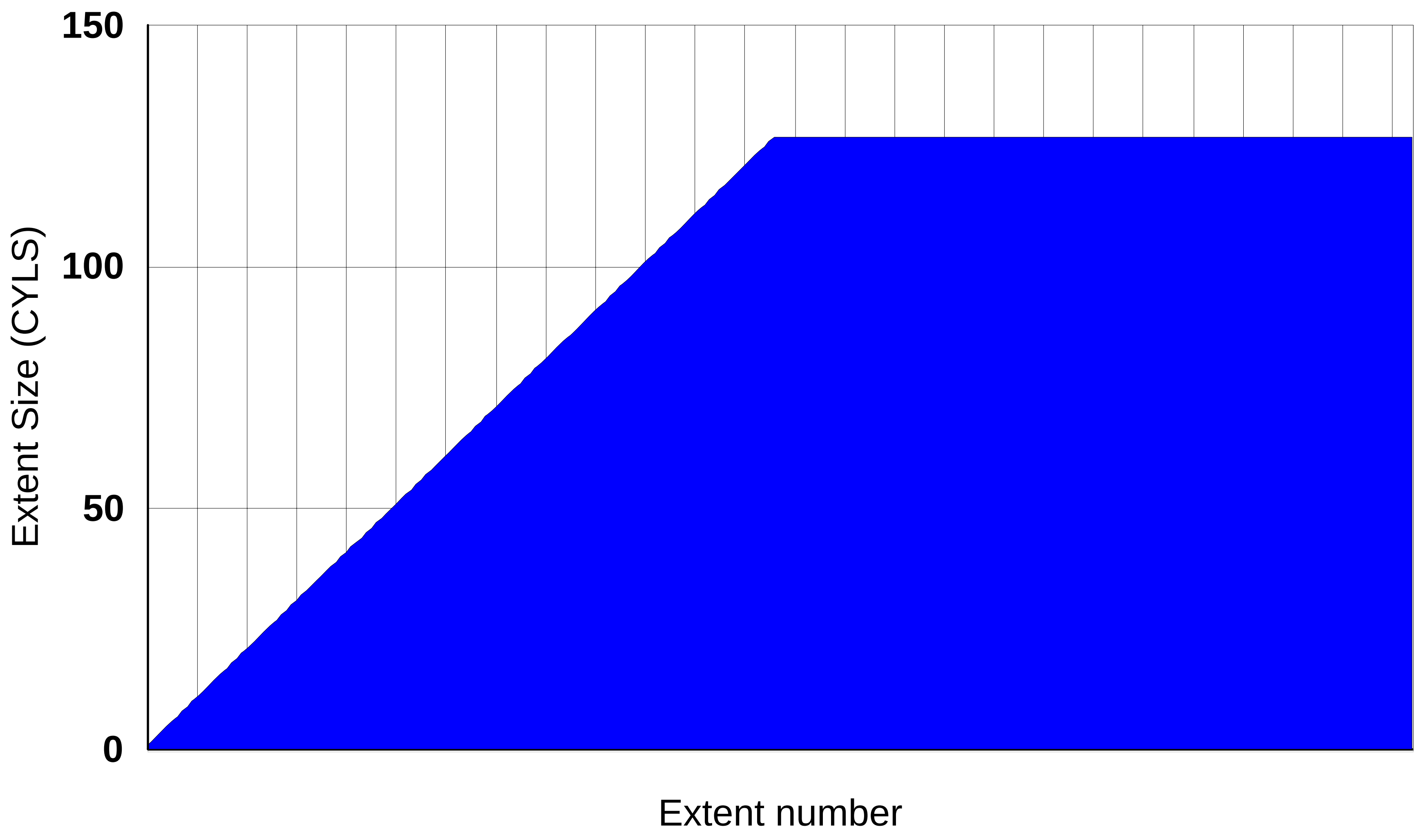
Automatic Space Management

- SMART enhancement: sliding secondary allocation quantity size
- Applies to DB2 managed pagesets only
- Tries to avoids VSAM maximum extent limit errors
- Can reach maximum dataset size before running out of extents
- Default PRIQTY
  - 1 cylinder for non-LOB tablespaces and indexes
  - 10 cylinders for LOB tablespaces
- Improved SQL SELECT and INSERT performance
  - 2x improvement relative to track allocation
- Can be used for
  - New pagesets: No need for PRIQTY/SECQTY values
  - Existing pagesets: SQL ALTER PRIQTY/SECQTY values to -1 (minus) plus schedule a REORG

Sliding Scale for 32GB and 64GB Data Sets



Sliding Scale for Other Data Sets



Limits: DB2 for z/OS

Breaking through limitations

– Virtual Storage 2 GB	231	to 264
– Table name sizes	18	to 128
– VIEW & ALIAS names	18	to 128
– Column name sizes	18	to 30
– Partitions	254	to 4096
– SQL statement length	32K	to 2 MB
– Index key size	255	to 2000
– Character Literals	255	to 32704
– Hex literal digits	255	to 32704
– Predicates	255	to 32704

Image of Earth from Moon, Source: NASA (Public Domain)

Limits: DB2 for z/OS …

Breaking through limitations ...

– Tables in a join	15 to 225
– Active logs	31 to 93
– Archive logs	1000 to 10,000
– Maximum table size (partitioned, 32K page)	16 TB to 128 TB
– Current optimization	8 to 128
– CURRENT PACKAGESET	18 to 128
– CURRENT PATH	254 to 2048
– SCHEMA	8 to 128

Image of Earth from Moon, Source: NASA (Public Domain)

SQL, DB2 Family and Portability

- Common Table Expression
- Recursion
- Multi-row INSERT & FETCH
- GET DIAGNOSTICS
- INSERT within SELECT
- IDENTITY Column enhancements
- SEQUENCES
- DYNAMIC SCROLLABLE CURSORS
- CURRENT PACKAGE PATH
- SCALAR FULLSELECT
- MATERIALIZED QUERY TABLES
- UNICODE SQL, MULTIPLE CCSIDs
- XML Publishing
- ...

DB2 SQL

z z/OS V7
common
LUW Linux, Unix & Windows V8.2

z Range partitioning

common Inner and Outer Joins, Table Expressions, Subqueries, GROUP BY, Complex Correlation, Global Temporary Tables, CASE, 100+ Built-in Functions, Limited Fetch, Insensitive Scroll Cursors, UNION Everywhere, MIN/MAX Single Index Support, Self Referencing Updates with Subqueries, Sort Avoidance for ORDER BY, and Row Expressions, Call from trigger, statement isolation

LUW Updateable UNION in Views, ORDER BY/FETCH FIRST in subselects & table expressions, GROUPING SETS, ROLLUP, CUBE, INSTEAD OF TRIGGER, EXCEPT, INTERSECT, 16 Built-in Functions, MERGE, Native SQL Procedure Language, SET CURRENT ISOLATION, BIGINT data type, file reference variables, SELECT FROM UPDATE, DELETE & MERGE, multi-site join, 2M Statement Length, GROUP BY Expression, Sequences, Scalar Fullselect, Materialized Query Tables, Common Table Expressions, Recursive SQL, CURRENT PACKAGE PATH, VOLATILE Tables, Star Join Sparse Index, Qualified Column names, Multiple DISTINCT clauses, ON COMMIT DROP, Transparent ROWID Column, FOR READ ONLY KEEP UPDATE LOCKS, SET CURRENT SCHEMA, Client special registers, long SQL object names, SELECT from INSERT

DB2 SQL

z z/OS V8
common
LUW Linux, Unix & Windows V8.2

z Multi-row INSERT, FETCH & multi-row cursor UPDATE, Dynamic Scrollable Cursors, GET DIAGNOSTICS, Enhanced UNICODE for SQL, join across encoding schemes, IS NOT DISTINCT FROM, Session variables, range partitioning

common Inner and Outer Joins, Table Expressions, Subqueries, GROUP BY, Complex Correlation, Global Temporary Tables, CASE, 100+ Built-in Functions including SQL/XML, Limited Fetch, Insensitive Scroll Cursors, UNION Everywhere, MIN/MAX Single Index Support, Self Referencing Updates with Subqueries, Sort Avoidance for ORDER BY, and Row Expressions, 2M Statement Length, GROUP BY Expression, Sequences, Scalar Fullselect, Materialized Query Tables, Common Table Expressions, Recursive SQL, CURRENT PACKAGE PATH, VOLATILE Tables, Star Join Sparse Index, Qualified Column names, Multiple DISTINCT clauses, ON COMMIT DROP, Transparent ROWID Column, Call from trigger, statement isolation, FOR READ ONLY KEEP UPDATE LOCKS, SET CURRENT SCHEMA, Client special registers, long SQL object names, SELECT from INSERT

LUW Updateable UNION in Views, ORDER BY/FETCH FIRST in subselects & table expressions, GROUPING SETS, ROLLUP, CUBE, INSTEAD OF TRIGGER, EXCEPT, INTERSECT, 16 Built-in Functions, MERGE, Native SQL Procedure Language, SET CURRENT ISOLATION, BIGINT data type, file reference variables, SELECT FROM UPDATE, DELETE & MERGE, multi-site join

DB2 SQL

z z/OS V9
common
LUW Linux, Unix & Windows V8.2

z

Multi-row INSERT, FETCH & multi-row cursor UPDATE, Dynamic Scrollable Cursors, GET DIAGNOSTICS, Enhanced UNICODE for SQL, join across encoding schemes, IS NOT DISTINCT FROM, Session variables, range partitioning, TRUNCATE, DECIMAL FLOAT, VARBINARY, optimistic locking, FETCH CONTINUE, ROLE, MERGE

common

Inner and Outer Joins, Table Expressions, Subqueries, GROUP BY, Complex Correlation, Global Temporary Tables, CASE, 100+ Built-in Functions including SQL/XML, Limited Fetch, Insensitive Scroll Cursors, UNION Everywhere, MIN/MAX Single Index Support, Self Referencing Updates with Subqueries, Sort Avoidance for ORDER BY, and Row Expressions, 2M Statement Length, GROUP BY Expression, Sequences, Scalar Fullselect, Materialized Query Tables, Common Table Expressions, Recursive SQL, CURRENT PACKAGE PATH, VOLATILE Tables, Star Join Sparse Index, Qualified Column names, Multiple DISTINCT clauses, ON COMMIT DROP, Transparent ROWID Column, Call from trigger, statement isolation, FOR READ ONLY KEEP UPDATE LOCKS, SET CURRENT SCHEMA, Client special registers, long SQL object names, SELECT from INSERT, UPDATE, DELETE & MERGE, INSTEAD OF TRIGGER, Native SQL Procedure Language, BIGINT, file reference variables, XML, FETCH FIRST & ORDER BY in subselect and fullselect, caseless comparisons, INTERSECT, EXCEPT, not logged tables

LUW

Updateable UNION in Views, GROUPING SETS, ROLLUP, CUBE, 16 Built-in Functions, SET CURRENT ISOLATION, multi-site join, MERGE

Summary

- Address Some Myths About V8
- Upgrade to V8 as quickly but as safely as possible
  - Secrets To Successful Migration
  - Pre-Migration Catalog Migration Testing
  - Quick Hits
  - Migration Plan Recommendations
- Value Proposition with V8
  - Improving performance and scalability
  - Prepare for and recover from logical data corruption
  - Better management of very large databases
  - Application Portability