

DB2 9 for z/OS Hints and Tips for Database Administrators

par Namik Hrle IBM



GUIDE Share France
Une Association Indépendante d'Utilisateurs IBM

Réunion du Guide DB2 pour z/OS France
Vendredi 27 novembre 2009
Tour Euro Plaza, Paris-La Défense

Disclaimer

© Copyright IBM Corporation 2009. All rights reserved.

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS AND/OR SOFTWARE.

IBM, the IBM logo, ibm.com, DB2 and DB2 for z/OS are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

Other company, product, or service names may be trademarks or service marks of others.

DB2 9 – A Rich, Features Filled Release

- SHRLEVEL(REFERENCE) for REORG of LOB tablespaces
- Online RENAME COLUMN
- Online RENAME INDEX
- Online CHECK DATA and CHECK LOB
- Online REBUILD INDEX
- Online ALTER COLUMN DEFAULT
- More online REORG by eliminating BUILD2 phase
- Faster REORG by intra-REORG parallelism
- Renaming SCHEMA, VCAT, OWNER, CREATOR
- LOB Locks reduction
- Skipping locked rows option
- Tape support for BACKUP and RESTORE SYSTEM utilities
- Recovery of individual tablespaces and indexes from volume-level backups
- Enhanced STOGROUP definition
- Conditional restart enhancements
- Histogram Statistics collection and exploitation
- WS II OmniFind based text search
- DB2 Trace enhancements
- WLM-assisted Buffer Pools management
- ...
- Global query optimization
- Generalizing sparse index and in-memory data caching method
- Optimization Service Center
- Autonomic reoptimization
- Logging enhancements
- LOBs network flow optimization
- Faster operations for variable-length rows
- NOT LOGGED tablespaces
- Index on expressions
- Universal Tablespaces
- Partition-by-growth tablespaces
- APPEND option at insert
- Autonomic index page split
- Different index page sizes
- Support for optimistic locking
- Faster and more automatic DB2 restart
- RLF improvements for remote application servers such as SAP
- Preserving consistency when recovering individual objects to a prior point in time
- CLONE Table: fast replacement of one table with another
- Index compression
- Index key randomization
- ...
- DECIMAL FLOAT
- BIGINT
- VARBINARY, BINARY
- TRUNCATE TABLE statement
- MERGE statement
- FETCH CONTINUE
- ORDER BY and FETCH FIRST n ROWS in sub-select and full-select
- ORDER OF extension to ORDER BY
- INTERSECT and EXCEPT Set Operations
- Instead of triggers
- Various scalar and built-in functions
- Cultural sort
- LOB File Reference support
- XML support in DB2 engine
- Enhancements to SQL Stored Procedures
- SELECT FROM UPDATE/DELETE/MERGE
- Enhanced CURRENT SCHEMA
- IP V6 support
- Unified Debugger
- Trusted Context
- Database ROLES
- Automatic creation of database objects
- Temporary space consolidation
- 'What-If' Indexes
- ...

TCO Reduction

Continuous Operations

Performance

Scalability

SQL

Portability

Reordered Row Format

- Remember tuning recommendations for rows with variable-length columns?
 - Define fixed-length columns preceding variable-length columns
- DB2 9 implements this recommendation internally
 - New **Reordered Row Format** provides for direct access to variable length columns
 - Potential for significant CPU reduction
 - For all newly created tables, at REORG and LOAD REPLACE
 - With some exceptions

Prefix	Fixed Length Cols	Varchar Pointers	Varying Length Cols
--------	-------------------	------------------	---------------------

- Rarely leading to increased logging volume
- LOAD REPLACE and REORG convert rows into RRF
 - KEEPDICTIONARY ignored when the utilities run for the first time in DB2 9
- Related service
 - PK87348
 - REORG option (ROWFORMAT BRF | RRF) to convert a tablespace from RRF to BRF and vice versa
 - Support for BRF by UTS
 - Introducing public zparm RRF instead of hidden SPRMRRF
 - PK78958 – conversion to RRF suppressed for compressed pagesets
 - Namely, RRF can lead to lower compression ratio

Unique DPSI

V8

- Data-Partitioned Secondary Index cannot be created as UNIQUE
- The reason:
Avoiding scanning all the partitions of DPSIs in case when uniqueness need to be enforced.
- An obstacle for wider usage of DPSIs

V9

- If the columns by which the table is partitioned are contained in the DPSI, only the corresponding DPSI partition need to be checked for uniqueness
- So, in DB2 9, DPSI can be unique subject to fulfilling the above condition

Identifying Unused Indexes

V8

- In order to improve the performance of different types of queries, some customers create many indexes just in case one of them can be useful.
- However, there is a significant cost in maintaining such indexes during Insert, Delete, Load, Reorg, ...
- Some of the indexes are no longer needed and they could be dropped, but how can we be sure that they have not been used for a very long time?

V9

- For each index, a new real-time statistics column (LASTUSED) is maintained in table SYSIBM.SYSINDEXSPACESTATS
- It is updated whenever the index is used in
 - SELECT/FETCH
 - searched UPDATE/DELETE
 - Referential Integrity checking
- The column is not updated for INSERT, LOAD, etc.
- If an index has not been used for a period that covers the entire applications life-cycle (e.g. do not forget periodic processes such as period closing), it can be safely dropped
- Be careful with indexes that enforce uniqueness

Larger Index Page Sizes

V8

- In heavy insert applications frequent index pages splitting creates scaling bottleneck
- In a data sharing environment, the indicators are:
 - ✓ Large Accounting Class 3 synchronous log write suspension
 - Each GBP-dependant index page split requires 2 forced log writes
 - Note that these suspensions include also waits for phase 1 of 2-phase commit and P-lock negotiation
 - ✓ High number of latch class 6 contentions
- For non-data sharing, the indicator is large number of latch class 254 contentions (reported in IFCID 57)

V9

- Additional 8K, 16K and 32K index page sizes
- Advantages
 - + It can reduce index page splits by up to 2x, 4x or 8x respectively
 - + Traversing shallow index needs less getpages
 - + Faster index scans
- Disadvantages
 - Larger pages require larger buffer pools
 - If access predominantly random, buffer pool might need to be 2x, 4x or 8x larger
 - For data sharing it can result in more P-lock contentions
- Rule of thumb:
 - ✓ 4K index page and larger PCTFREE for indexes with random insert patterns
 - ✓ larger size index page for indexes with sequential insert patterns.

Comparing Tablespace vs. Index Compression

	Tablespace Compression	Index Compression	Remarks
Unit of compression	Row	Page	Index non-leaf pages are not compressed, but that's typically less than 5% of the entire index
Where is data compressed	on DASD, in buffer pools, in logs	on DASD only	
When does compression happen	at insert and fetch	at I/O	Hence, CPU overhead affected by BP hit ratio. Larger buffer pools strongly recommended for compressed indexes. Do not use page-fix for these buffer pools
Hardware assist	Yes	No	Index compression uses sophisticated software compression algorithms
Page size restrictions	No	Only for 8K, 16K and 32K-page indexes	Page size on DASD: → for tablespaces, equivalent to page size in the associated buffer pool → for indexes, always 4K
Compression dictionary used	Yes	No	
When does compression start	After the first Reorg or Load	At the first insert or update	
Compression CPU cost reported in	Accounting	Accounting and/or DBM1 SRB Statistics	No changes in accounting CPU time if index pages brought in by prefetch
Typical Compression Ratio	10 – 90%	25 – 75%	Maximum CR limited by index page size: 50% for 8K, 75% for 16K and 87.5% for 32K Use DSN1COMP to predict compression ratio

More Efficient Workfiles Usage

V8

- 4K-page workfiles are used whenever row is smaller than 4KB
- Since work file access is often sequential, using larger page size can be more efficient
- E.g. for 2050-byte records:
15 records on one 32K page
vs.
8 records on eight 4K pages

V9

- 32K-page workfiles are used much more aggressively
- 4K-page workfiles are now used only for small records
 - where the limit of 255 rows per page results in waste of space
 - e.g. over 90% wasted space on 32K page for 10-byte records
- Recommendations
 - ✓ Assign a larger 32K workfile buffer pool
 - ✓ Allocate more 32K workfile data sets
 - ✓ If 4K workfile buffer pool activity is significantly lower, then the corresponding buffer pool size and work file datasets can be reduced.
 - ✓ Monitor new statistics on how often more optimal 32K workfiles ran out and 4K workfiles had to be used instead, or vice versa

BACKUP and RESTORE SYSTEM Enhancements

V8

- Off-loading system-level backups to tape and restoring it from tape is a manual process that is not driven by the BACKUP and RESTORE SYSTEM utilities
- Despite availability of system-level backup many sites produce tablespace and index image copies for simpler recoveries of the individual objects
- Physical background copy phase of FlashCopy can take long and impact overall I/O performance
- RESTORE SYSTEM can be used for prior point in time recoveries only

V9

- Full integration of tape into
 - BACKUP SYSTEM as the target for saving system-level backups
 - RESTORE SYSTEM as the source for restoring previously saved system-level backups
- Automatic using of system-level backups as alternative sources for recovering individual tablespaces and indexes
- BACKUP SYSTEM supports Incremental FlashCopy functionality
- DB2 can be restarted in System Point-in-Time Recovery mode without truncating the logs to a prior log point. RESTORE SYSTEM can then be used to recover DB2 to the current end of log.
- MODIFY RECOVERY enhanced to support system-level backups as the primary means of backing up DB2 data

Incremental FlashCopy

- Introduced by DFSMSHsm APAR OA17314 in z/OS 1.8
 - Incremental FlashCopy hardware relationship established for each source volume in copy pool with corresponding target volumes
 - Only changed tracks are copied to the corresponding tracks in the initial backup
 - Multiple FlashCopy versions are not supported
 - Significantly reduces duration and I/O impact of the FlashCopy's physical background copy phase

- Incremental FlashCopy support added to BACKUP SYSTEM utility
 - DB2 APAR PK41001
 - New keywords on BACKUP SYSTEM utility
 - ESTABLISH FCINCREMENTAL
 - Passes FCINCREMENTAL keyword to DFSMSHsm
 - Establishes incremental relationship
 - Initial FlashCopy creates a full backup
 - Subsequent FlashCopy requests copy only changed tracks
 - END FCINCREMENTAL
 - Takes one more incremental FlashCopy
 - Withdraws the incremental FlashCopy relationship

RESTORE SYSTEM to Currency

- Additional safety net for multi-volume corruptions in case that storage-fault tolerance fails
- Original system point-in-time recovery designed to truncate DB2 logs to a prior log point
- DB2 APAR PK51979 provides for using RECOVER SYSTEM for recovering to currency
 - DSNJU003 changed to allow a SYSPITR conditional restart without truncating logs to a prior log point
 - CRESTART CREATE, SYSPITR=FFFFFFFFFFFFFF
 - DB2 will be restarted in System Recover Pending mode without truncating the logs. No database update activity will be allowed, and DB2 will be in restricted access mode.
 - The RESTORE SYSTEM Utility can now be used to recover DB2 to the current end of log.

Preserving Consistency at Recovery to a Prior Point in Time

V8

- When recovering a DB2 object to a point in time at which there were uncommitted changes for that object, the recovered object is left in an inconsistent state
- In order to ensure that prior point in time recoveries result in consistent objects, frequent quiesce points need to be established. However:
 - Quiesce can be very disruptive process for concurrent transactions
 - Even when quiesce points exist, they might not coincide with the point to which an object needs to be recovered

V9

- RECOVER utility enhanced to automatically detect uncommitted transactions running at the recovery point in time and roll them back
- RECOVER TOLOGPOINT and TORBA will always recover with consistency
- RECOVER TOCOPY, TOLASTCOPY and TOLASTFULLCOPY using SHRLEVEL CHANGE copy will continue working as in V8

Universal Tablespaces

V8

A tablespace needs both partitioned and segmented organization:

- it's larger than 64GB
- inter-partition parallelism or independent processing is needed
- partition scope operations (ADD,ROTATE) apply
- rows are variable in length and optimal space utilization is preferred
- mass delete operations should be fast

V9

- A hybrid between partitioned and segmented organization

```
CREATE TABLESPACE ...  
  SEGSIZE integer  
  NUMPARTS integer
```

- One table per UTS only
- Two types:
 1. Partitioned-by-growth
 - always UTS
 - described on the next page
 2. Partitioned-by-range
 - traditional partitioned tablespaces
 - optionally UTS
- Incompatible with MEMBER CLUSTER

Partition By Growth

V8

A table's growth is unpredictable (it could exceed 64GB) and there is no convenient key for range partitioning.

Partitioning by a ROWID column introduces additional tablespace administration overhead:

- estimating optimal number of partitions
- ADDing partitions if necessary
- less than optimal space utilization)

V9

CREATE TABLESPACE ... *explicit specification*
MAXPARTITIONS integer

CREATE TABLE ... *implicit specification*
PARTITIONED BY SIZE EVERY integer G

Associated SYSTABLESPACES columns

MAXPARTITIONS = max number of partitions
PARTITIONS = actual number of partitions
TYPE = G

- Only single-table tablespace
- Universal Tablespace organization: although the table space is partitioned, the data within each partition is organized according to segmented architecture
- Incompatible with MEMBER CLUSTER, ADD PARTITION, ROTATE PARTITION
- REORG considerations

Partition By Growth

- MAXPARTITIONS can be ALTERed but observe the limits that are dependent from page size and DSSIZE

Page Size \ DSSIZE	4K	8K	16K	32K
1 – 4 GB	4096	4096	4096	4096
8 GB	2048	4096	4096	4096
16 GB	1024	2048	4096	4096
32 GB	512	1024	2048	4096
64 GB	256	512	1024	2048

- REORG can add new partitions (except if there are LOB or XML columns).
- REORG will not remove empty partitions, but it can shrink them to contain a header and space map page, again, subject to absence of LOB and XML columns

Automatic Creation of Database Objects

V8

- Database objects that have different meanings for different database platforms must be transparent to database-platform agnostic applications, e.g. database and tablespace
- Some of the DB2 database objects must be manually created although all their attributes are implicitly known to DB2, e.g. indexes that enforce primary and unique keys

V9

Unless explicitly specified, DB2 will implicitly create the following database objects:

- Database
- Tablespace
- Index that enforces primary key
- Index that enforces unique key
- ROWID index
 - For ROWID columns defined as GENERATED BY DEFAULT)
- LOB tablespace
- LOB auxiliary table
- LOB auxiliary index

Automatic Creation of *Database*

- Automatically (implicitly) created databases are used when IN clause is omitted at CREATE TABLE. There are two possible outcomes:
 1. A new database is created
 - Namespace DSN00001 – DSN10000
 - PK62178 changes the maximum from 60000 to 10000
 - Schema: SYSIBM
 2. An existing implicitly created database is used
 - Start reusing existing databases after reaching 10000
 - Wrap-around fashion
 - Therefore, implicitly created databases can contain multiple tablespaces
- Explicitly created tables and tablespaces in implicitly created databases are not allowed
- Implicitly created databases can be explicitly dropped

Automatic Creation of *Tablespace*

If CREATE TABLE does not include explicit tablespace specification, a tablespace is automatically (implicitly) created with the following attributes:

- Partition-by-growth universal tablespace
 - SEGSIZE=4
 - DSSIZE=4G
 - MAXPARTITIONS=256
- Compression is controlled by zparm IMPTSCMP
 - Default is COMPRESS=NO
- Dataset creation is controlled by zparm IMPDSDEF
 - Default is DEFINE=YES
- Sliding scale for optimizing secondary extent allocation is used by default (MGEXTSZ=YES)
- The default buffer pool is determined based on the zparm settings.
 - Buffer pool page size is derived automatically using larger page size if max record size reaches 90% of capacity of the needed size

Tighter Integration With DFSMS

V8

- DFSMS constructs such as data class, storage class and management class are used by DB2 page sets, but not visible to DB2 administrators
- Mapping of DB2 page sets to these storage constructs is defined in the ACS routines which are typically maintained by storage administrators

V9

```
CREATE | ALTER STOGROUP  
  DATACLAS dcname  
  MGMTCLAS mcname  
  STORCLAS scname
```

- The existing VOLUME clause is now optional: it can be omitted if any of the DFSMS classes is specified
- If explicitly specified to DB2, the new attributes are recorded in SYSIBM.SYSSTOGROUP

Online REBUILD INDEX

V8

- During rebuilding of an index the tablespace is unavailable for all data modifying operations
- This is especially restrictive for very large tables where the rebuild process typically takes considerable amount of time

V9

```
REBUILD INDEX ...  
  SHRLEVEL REFERENCE  
  DRAIN_WAIT integer  
  RETRY integer  
  RETRY_DELAY integer
```

```
REBUILD INDEX ...  
  SHRLEVEL CHANGE  
  DRAIN_WAIT integer  
  RETRY integer  
  RETRY_DELAY integer  
  MAXRO integer | DEFER  
  LONGLOG CONTINUE | TERM | DRAIN  
  DELAY 1200 | integer
```

Renaming SCHEMA, VCAT, OWNER and CREATOR

V8

- Some administrative operations (such as cloning a DB2 system) often includes the need to change any or all of the following:
VCAT, SCHEMA, OWNER, CREATOR
- These are error prone, time consuming and risky operations:
 - The changes are spread throughout DB2
 - Preserving coherency between the catalog, the dictionary and actual physical objects is absolutely crucial

V9

CATMAINT UPDATE ...

SCHEMA SWITCH (schema_name) TO (new_schema_name)

VCAT SWITCH (vcat_name) TO (new_vcat_name)

OWNER FROM (owner_name) TO ROLE

- Performs authorization/semantics checking and serialization
- Updates catalog and directory to reflect the new names
- Invalidates plan, packages and statement cache
- Names to be changed must not have view, function, MQT and trigger dependencies, cannot be 'SYSIBM'
- Use SCHEMA option to change owner, creator and schema names

Resource Limit Facility Improvements

V8

The qualifiers used by RLF to identify processes for which CPU time is governed are not specific enough for most 'middleware' servers such as SAP:

- Plan name is fixed to DISTSERV
- CLI and JDBC drivers use a common set of packages
- Authorization ID is typically the same for the entire workload

V9

- RLF governed processes can be now qualified by typical 'client-side' identifiers:
 - End-User ID
 - Transaction name
 - Workstation name
- Additionally, the process can be qualified by the IP address of the server that initiated request
- This applies to both the predictive and reactive governor
- The existing CLI and JDBC APIs are used to set the client-side identifiers.

Not Logged Tablespaces – Use It for Extreme Cases Only

V8

Performance degradation for workloads involving very large number of parallel inserts, updates and deletes due to:

- Log write latching
- Log data sets write bottlenecks

Very large volume of log data generated for these workloads

At the same time **recoverability of data is not required**, e.g. the data can be recreated from its original source rather than from backups and logs

V9

New tablespace attribute that controls whether Undo and Redo information for that tablespace and associated indexes are logged or not.

`CREATE or ALTER TABLESPACE ...
LOGGED|NOT LOGGED`

Typical process sequence:

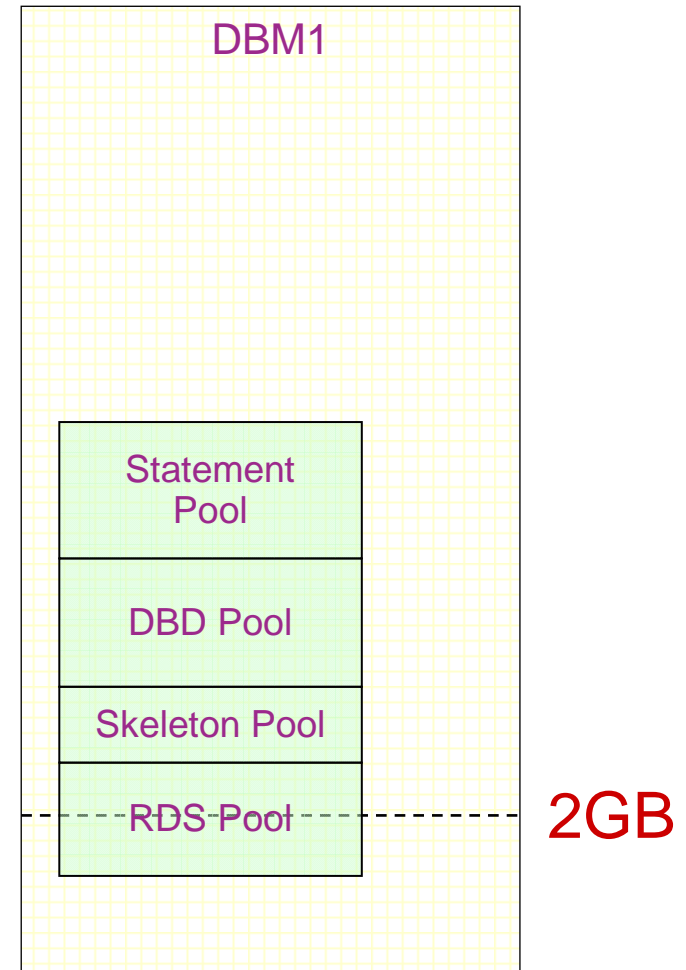
1. Take full image copy
2. Turn off logging
3. Make massive changes
4. Ensure that no other concurrent, non-repeatable changes happen
5. Turn on logging
6. Take full image copy

Get familiar with ramifications to rollbacks, restarts, lock contentions, data sharing, long running transactions

EDM Statement Pool

The dynamic statements are cached in the EDM Statement Pool which belongs to EDM storage.

- EDM Statement Pool
 - Skeleton dynamic statements (SKDS)
 - zparm EDMSTMTC
 - As of DB2 V8 above-the-bar
- EDM DBD Pool
 - Database Descriptors (DBDs)
 - zparm EDMDBDC
 - As of DB2 V8 above-the-bar
- EDM Skeleton Pool
 - Skeleton Cursor (SKCT) and Package Tables (SKPT)
 - zparm EDM_SKELETON_POOL
 - As of DB2 V9 above-the-bar
- EDM RDS Pool
 - Cursor (CT) and Package Tables (PT)
 - zparm EDMPOOL
 - As of DB2 V9 partially above-the-bar



Thread Virtual Storage

- A large portion (up to 50%) of virtual storage that holds locally cached statements (particularly INSERTs) moved above-the-bar in V9
- Changed defaults for two zparms that control thread virtual storage utilization
 - CACHEDYN_FREELOCAL
 - Introduced in DB2 V8, APAR PK21861
 - Enables freeing cached statement and releasing associated virtual storage upon statement's completion, i.e. does not wait for a commit to do that.
 - If the statement is re-referenced after being freed, DB2 performs implicit prepare, so the process is transparent to the applications
 - Unless there are concurrent DROPs or ALTERs affecting associated objects, in which case sqlcode -204 could be returned to the application executing the freed statement
 - Applies only to statements bound with KEEP_DYNAMIC(YES)
 - Possible values
 - 0 - Feature is disabled. Default value in DB2 V8.
 - 1 - Feature is enabled. Default value in DB2 9.
 - 2, 3, 4 are serviceability values
 - MINSTOR
 - If set to YES, DB2 actively works on reducing thread's virtual storage footprint
 - Default is NO in DB2 V8, but changed to YES in DB2 9

Plan Stability

V8

- REBINDs can cause access path changes
- Most of the time, this improves query performance, but when it does not
 - No easy way to undo the REBIND
 - Can lead to a lot of grief to our customers and to IBM
- Existing approaches have their weaknesses
 - Preventing REBINDS altogether
 - Rebinding into alternate collections
 - Using hints

V9

- At REBIND, DB2 saves old copies of the packages
- If the event of a performance regression, users will have a way to fallback to an older copy
- Post-GA (PK52523) feature

Disclaimer

© Copyright IBM Corporation 2009. All rights reserved.

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS AND/OR SOFTWARE.

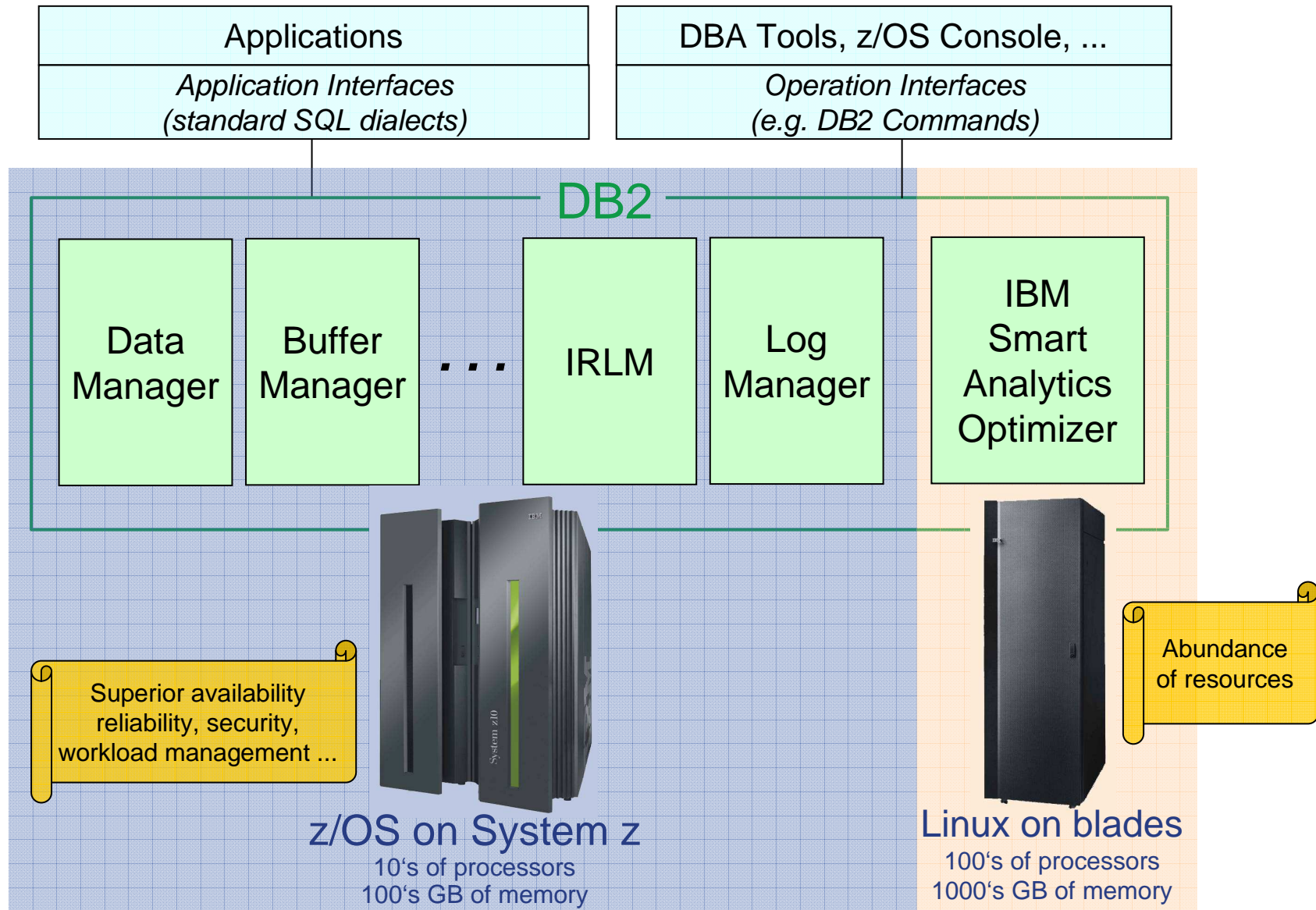
The information on the new product is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information on the new product is for informational purposes only and may not be incorporated into any contract. The information on the new product is not a commitment, promise, or legal obligation to deliver any material, code or functionality. The development, release, and timing of any features or functionality described for our products remains at our sole discretion.

IBM, the IBM logo, ibm.com, DB2, and DB2 for z/OS are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

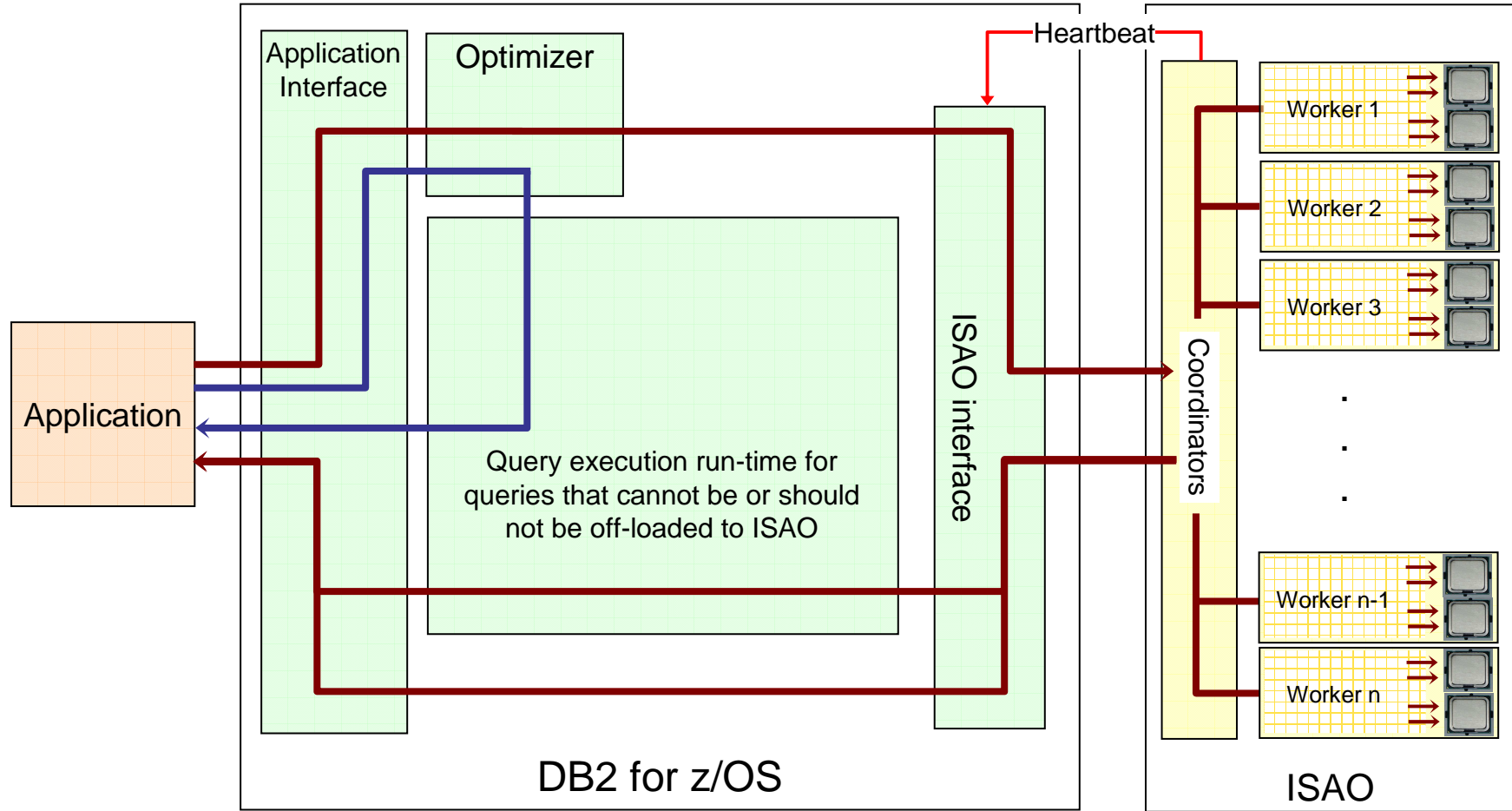
Other company, product, or service names may be trademarks or service marks of others.

Deep DB2 Integration within zHybrid Architecture

IBM Smart Analytics Optimizer as a Virtual DB2 Component



Query Execution Process Flow



- Heartbeat (ISAO availability and performance indicators)
- Queries executed without ISAO
- Queries executed with ISAO

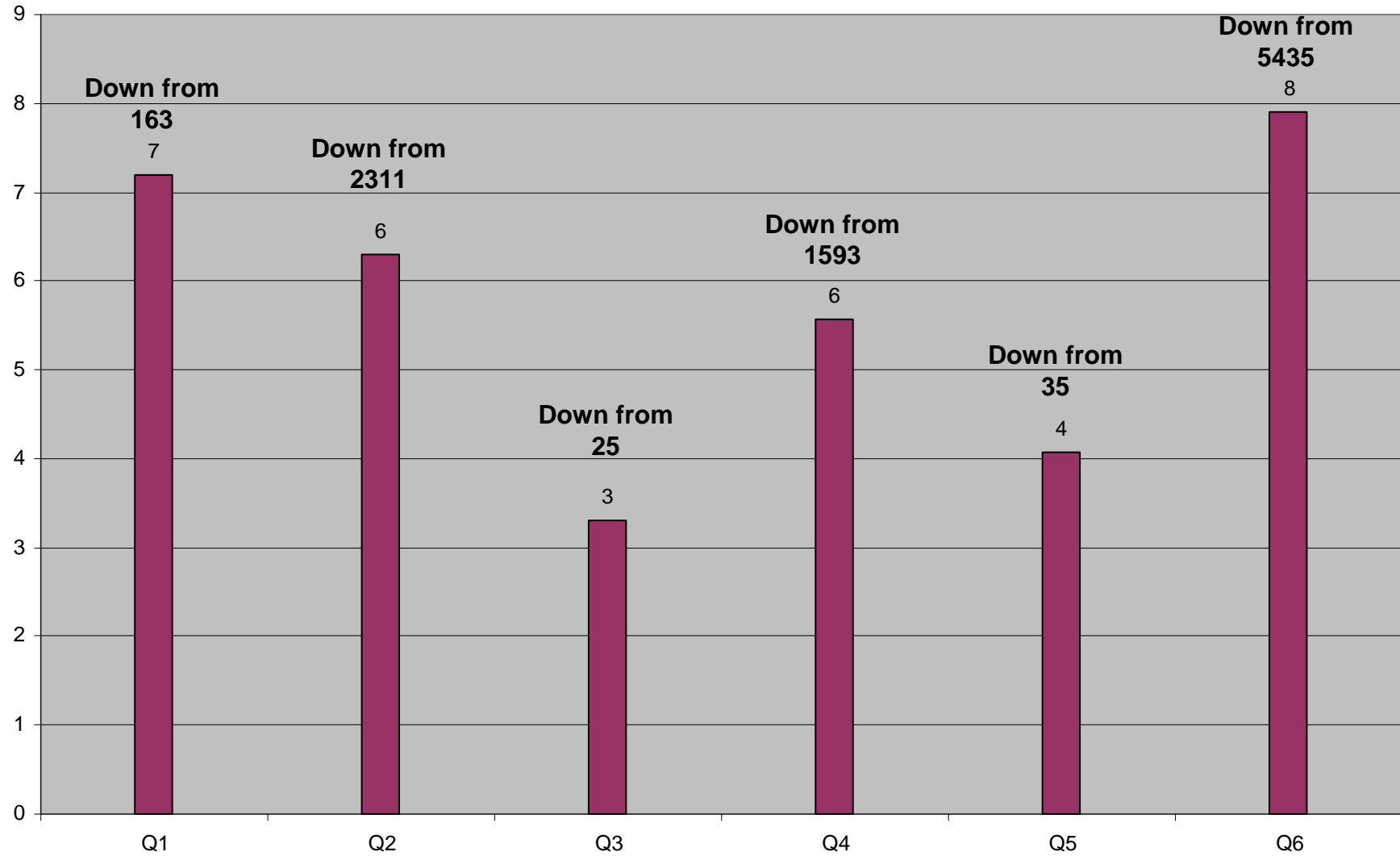
Testing Results

- The problem queries provided by a customer
- Expert database tuning done on all the queries
 - Q1 – Q6 even after tuning run for too long and consume lots of resources
 - Q7 improved significantly – no ISAO offload is needed
- The table shows elapsed and CPU times measured in DB2 (without ISAO)

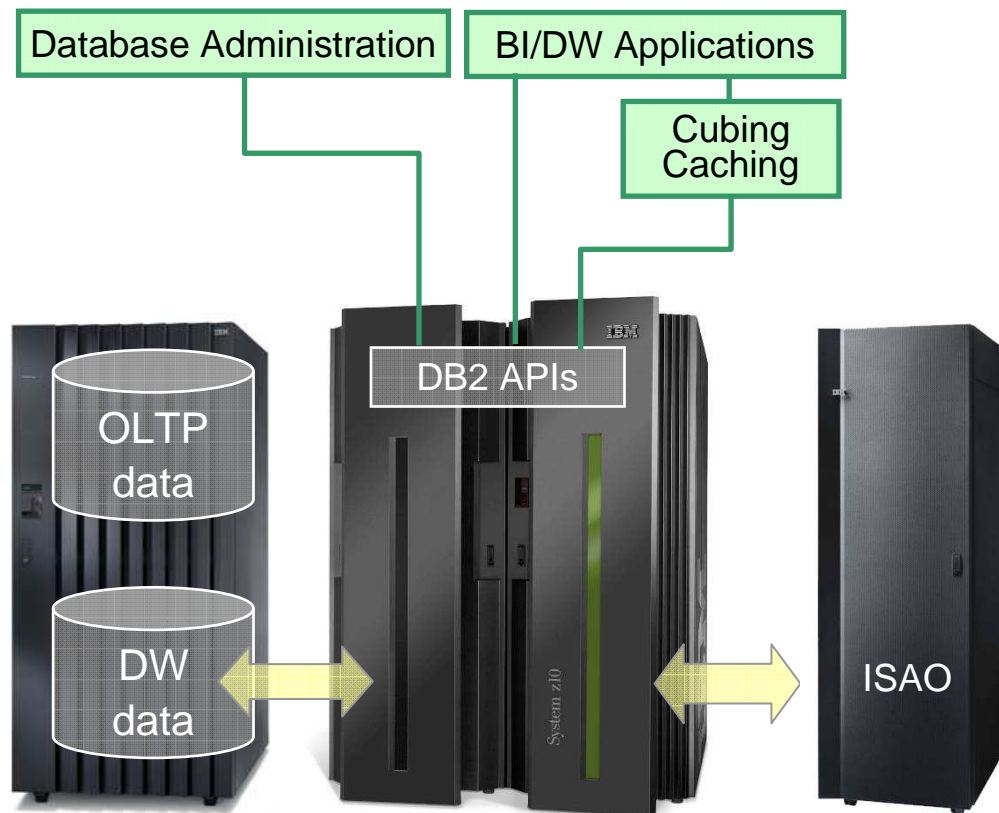
Query	Times measured in DB2 without ISAO			
	Total Elapsed	CP	zIIP	Total CPU Time
Q1	0:02:43	0:03:52	0:02:39	0:06:31
Q2	0:38:31	0:11:52	0:36:10	0:48:02
Q3	0:00:25	0:00:04	0:00:15	0:00:19
Q4	0:26:33	0:13:43	0:20:50	0:34:33
Q5	0:00:35	0:00:09	0:00:29	0:00:38
Q6	1:30:35	5:53:30	1:29:56	7:23:26
Q7	0:00:02	0:00:02	0:00:00	0:00:02

Testing Results: Performance Improvement after Adding ISAO

seconds



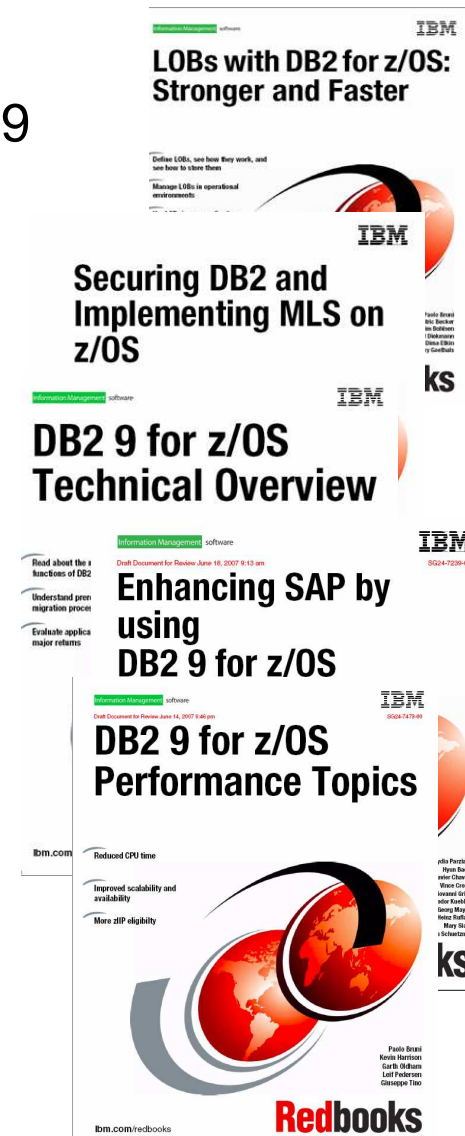
IBM Smart Analytics Optimizer



- Seamless integration of new computing paradigms into proven technology
 - Massive multi-core and vector processing
 - In-memory database
 - Hybrid row- and column-based store
 - No changes to the applications, applications continue to attach to DB2
 - Preserving traditional System z and DB2 quality of service, full fencing and protection of DB2 against possible ISAO failures
- Order of magnitude performance improvement
 - Linear scaling with the number of CPUs
- Reducing need for tedious tuning of DB2 (MQTs, aggregates, indexes, etc.)
- Appliance characteristics
 - User/reference guide assisted installation, initial configuration
 - Hands free operations
- Providing building block for Dynamic DW and Operational BI

DB2 9 for z/OS RedBooks & RedPapers

- Powering SOA with IBM Data Servers SG24-7259
- LOBs with DB2 for z/OS: SG24-7270
- Securing DB2 & MLS z/OS SG24-6480-01
- DB2 9 Technical Overview SG24-7330
- Enhancing SAP - DB2 9 SG24-7239
- Best practices SAP BI - DB2 9 SG24-6489-01
- DB2 9 Performance Topics SG24-7473
- DB2 9 Optimization Service Center SG24-7421
- Index Compression with DB2 9 for z/OS paper
- DB2 9 Stored Procedures SG24-7604



DB2 9 for z/OS

Hints and Tips for Database Administrators

par Namik Hrle IBM
hrle@de.ibm.com