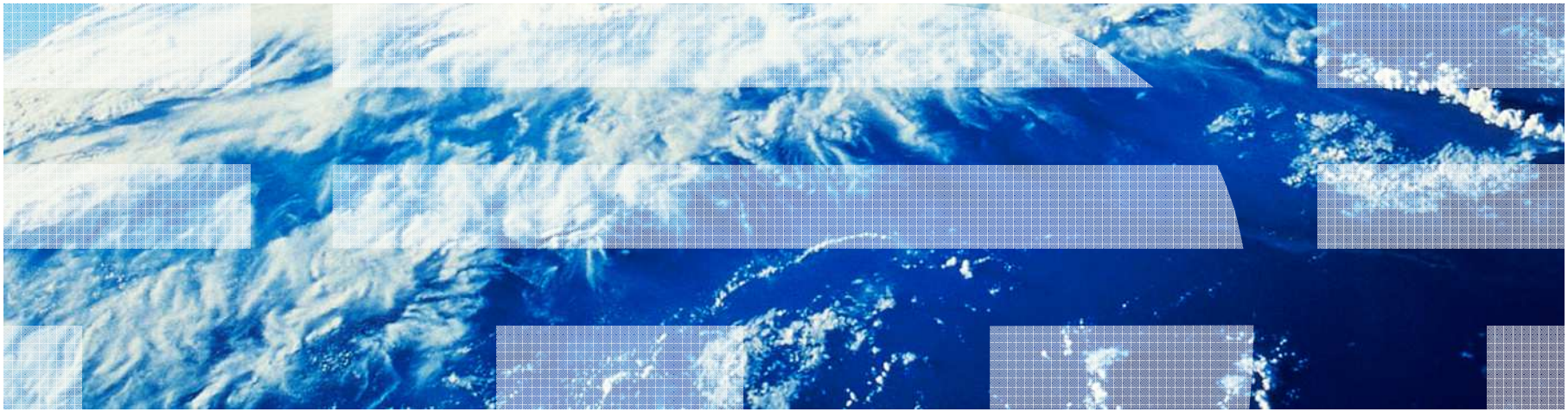
Matt Anglin, Tivoli Storage Manager Server Development

TSM Symposium, September 2011

Exploring the TSM V6 Database: Everything You Need to Know



Abstract

This session will explore the TSM V6 database from deployment and performance recommendations through best practices and troubleshooting. The deployment discussion will cover performance and configuration recommendations for new and upgraded servers. Server upgrade capabilities and considerations will also be covered. This session will provide details about the server database, lessons learned since 2009 when the new database was introduced, and other hints and tips for a successful experience.

TSM Database Comparison: V5 to V6

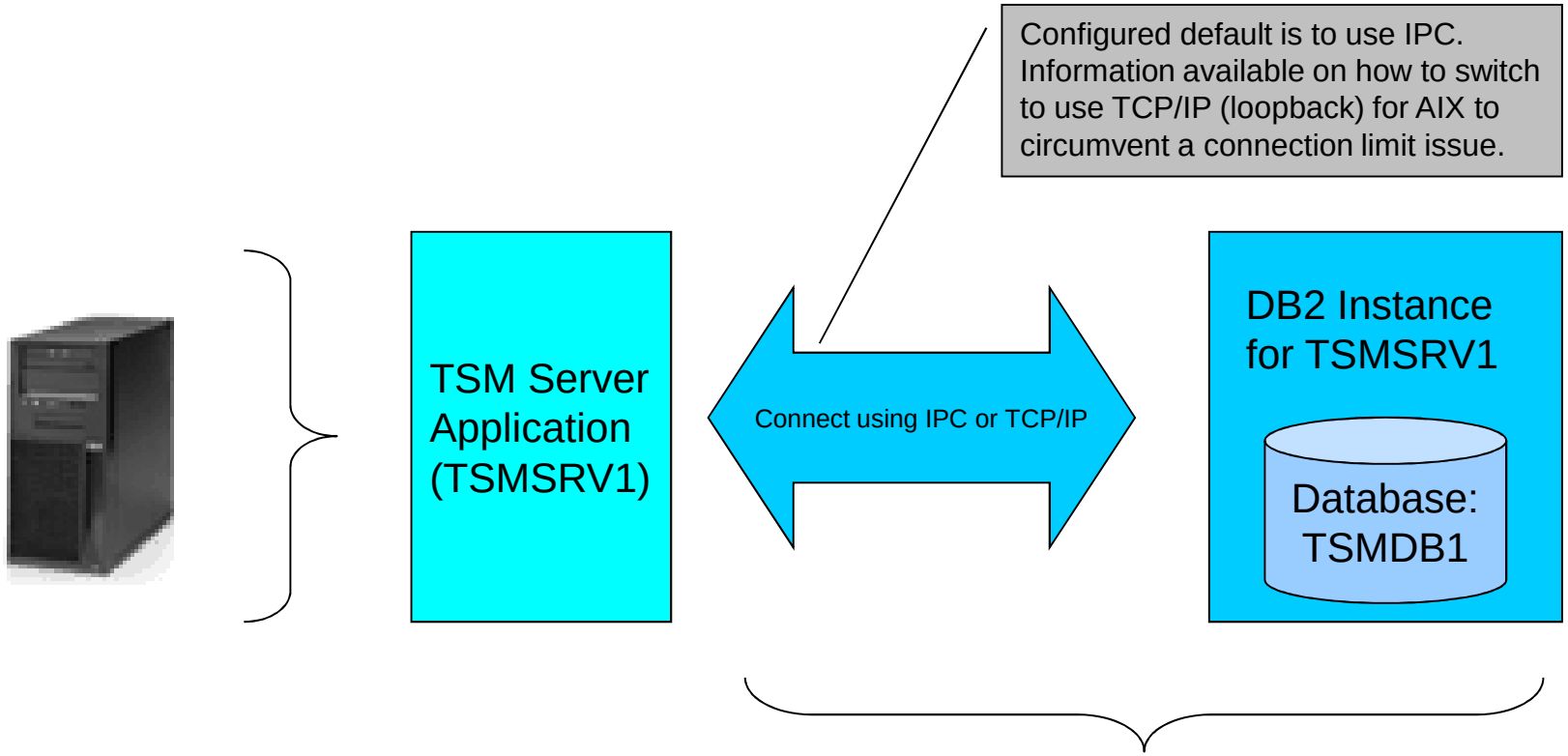
TSM <= V5	TSM >= 6.1	Presentation Topic
dsmserv.dsk (bootstrap/info for db and log volumes)	Server instance directory	INSTANCE
Database volume(s)	Directories assigned for the database	DB
Recovery log volume(s)	▪ Directory assigned for the active log ▪ Directory assigned for the archive log	LOG
Options: bufpoolsize/logbufpoolsize	Option: dbmempercent	MEMORY
Unload/loadformat/load Utilities for database reorganization	Online table and index reorganization	REORGANIZATION
Virtualized SQL iterator (administrative SQL)	Views and Staging tables with passthru to DB2 SQL processing	SQL
	Database upgrade to V6, what tools and considerations are available.	UPGRADE

The table above shows the topics that will be discussed and the slide heading keyword that will be used. This table of contents approach is provided in order to use this as a reference when needed.

INSTANCE: Instance model

- An Instance is everything required to run a server
  - Database files
  - Log files
  - Configuration files (dsmserv.opt, volume history, etc.)
  - Storage pool volumes
- Each instance requires it own directory
  - dsmserv.opt is **always** located in instance directory
  - Database and log files should be stored to separate and different directory and storage space.
- You can have more than one instance per system
  - Each instance is separate from every other instance
  - Each instance has a separate single database
  - The DB2 instances are named and unique to each TSM instance.
    - Within the instance, the server database is TSMDB1.

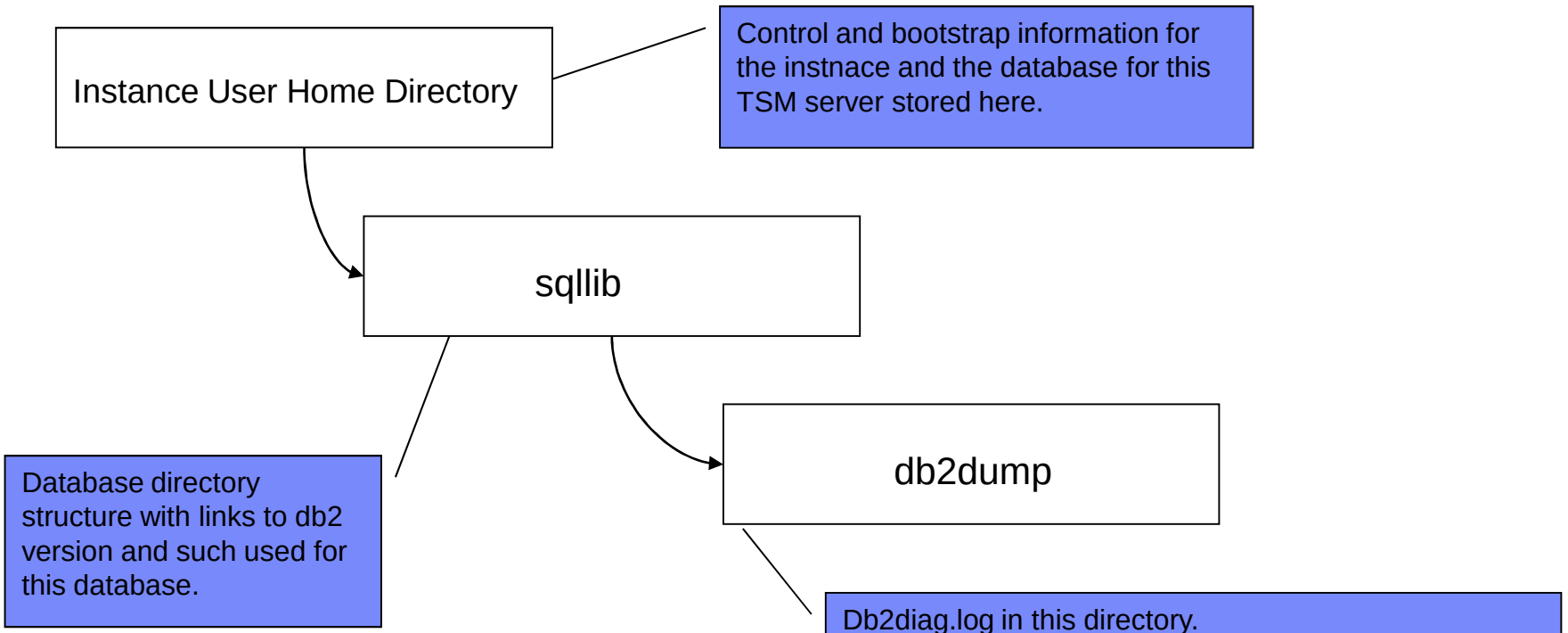
INSTANCE: Illustrated



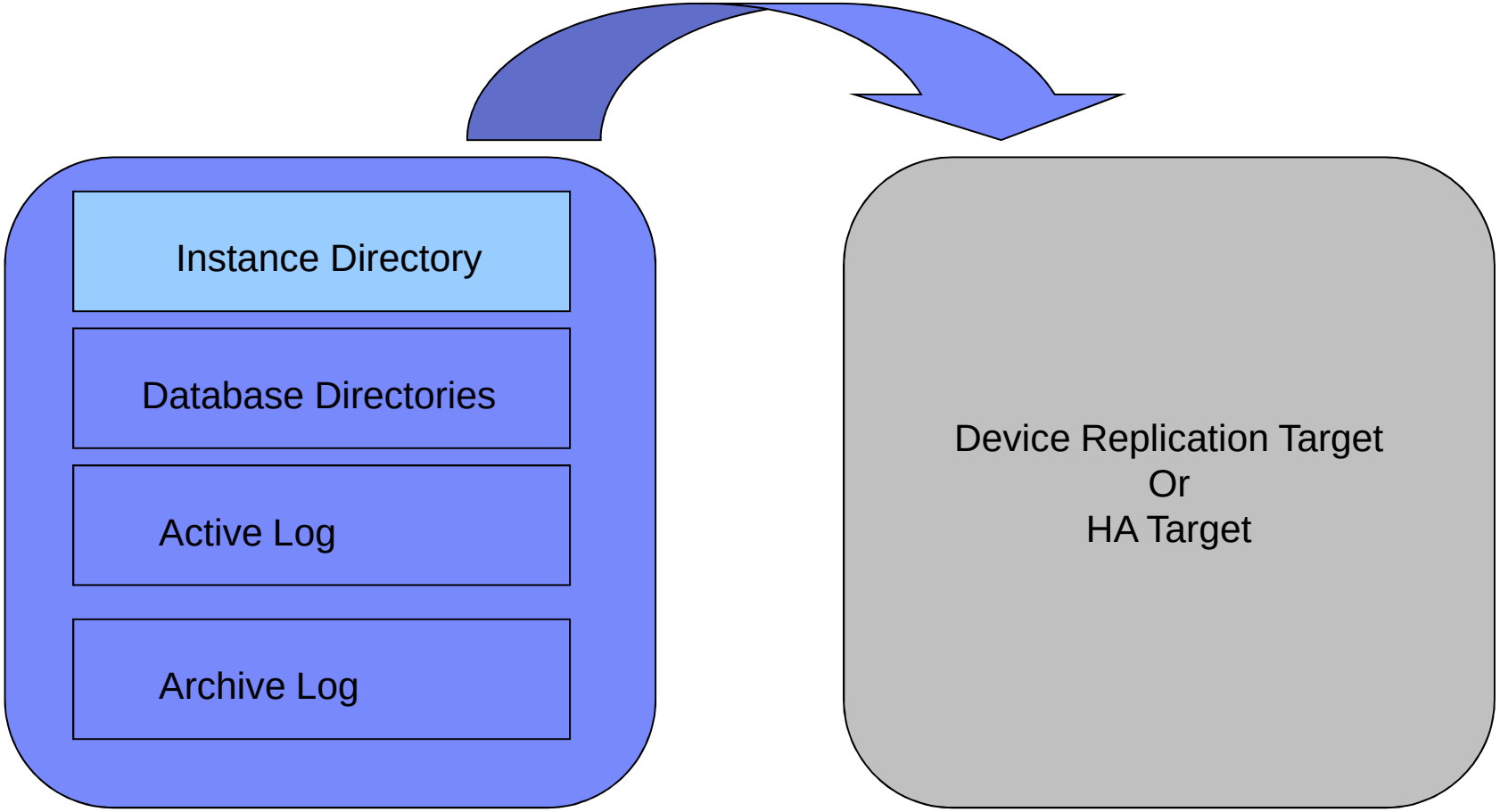
DB2 Assets:

- Connections done using DB2 CLI client.
- DB2 server (processes) to represent corresponding DB2 instance and database.

INSTANCE: Server instance directory



INSTANCE: Server instance directory and replication or high availability



If using disk based replication or a high-availability solution, be sure to include the instance directory as part of the consistency group or shared disk that is managed for this purpose.

Omitting the instance directory can result in an unusable server at the target site.

INSTANCE: Other instance directory considerations

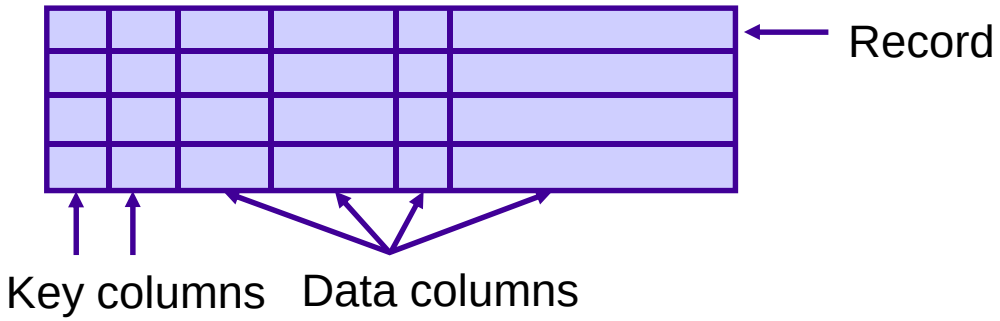
- For a crash or dump:
  - If DB2 crashed, the core or dump file will be in the db2dump directory.
  - If TSM crashed, the core or dump file will be in the instance directory.
  - Be sure to delete these once they are no longer needed.
  - Keep in mind that core/dump files can be large as these represent the application and memory used by that application.
- The file system for the instance directory should be monitored for space.
  - If the available space is exhausted, this can result in:
    - Runtime or operational failures if the server is up and running at the time that this occurs.
    - Inability to restart the server.
  - Updates are made to files in the instance directory so space is necessary in this directory in order to allow the server to run and operate properly.
- Planning Considerations:
  - Provision space for the instance directory like you would the database and active log.
    - Simply adding this to /home or a C: without consideration for the required space can result in issues either immediately or else in the future.
    - TSM InformationCenter suggests ~2 GB for the server home directory.

      In practice, this should be larger to account for space in this directory being consumed.

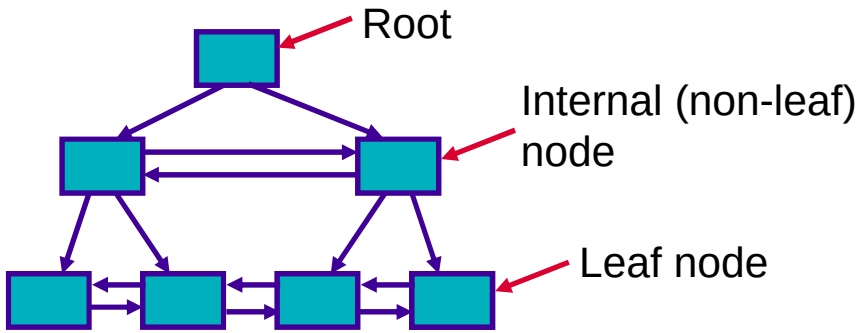
      And if core/dump files are not being directed to a different location than this will need to be larger.

DB: Database table objects in V5 and earlier

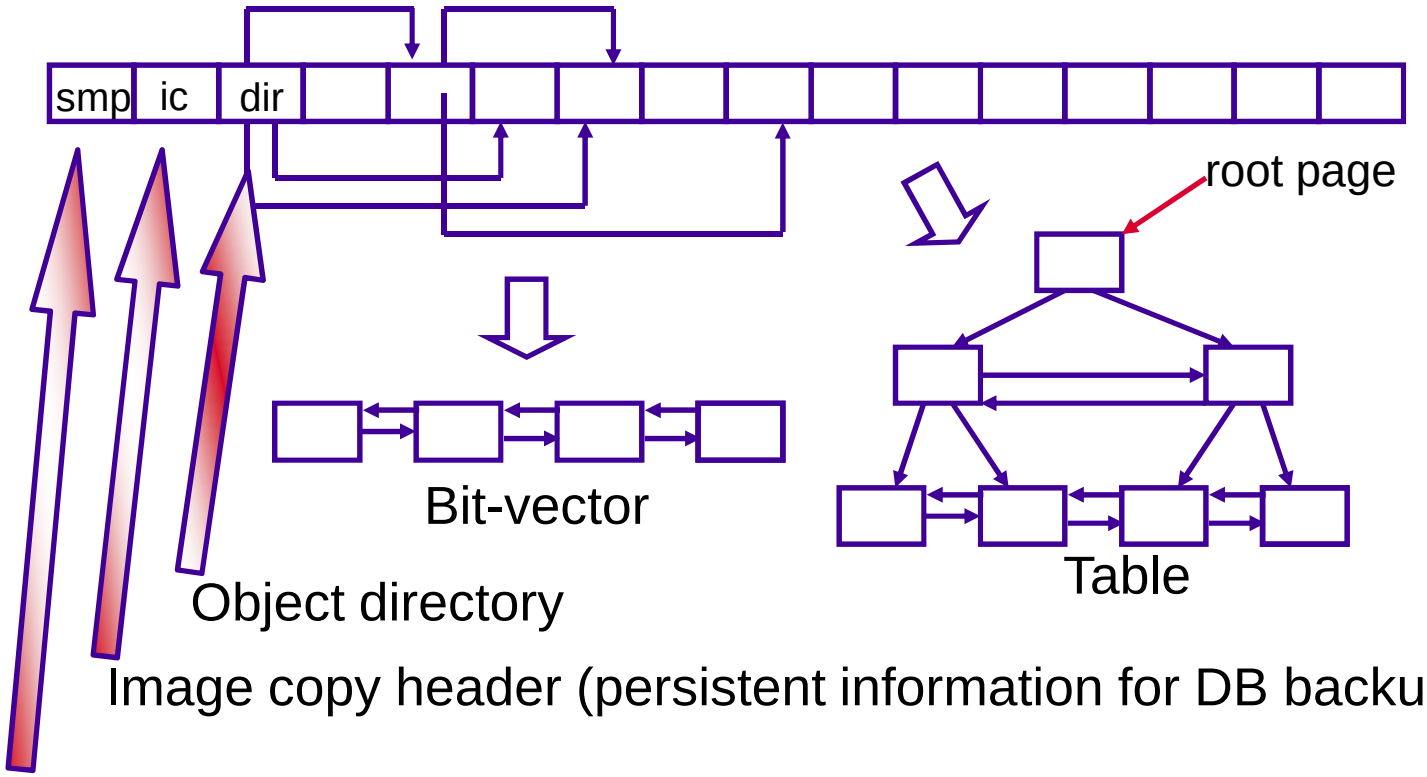
From the perspective of a TSM component that uses the database, table schema describes key and data columns for table



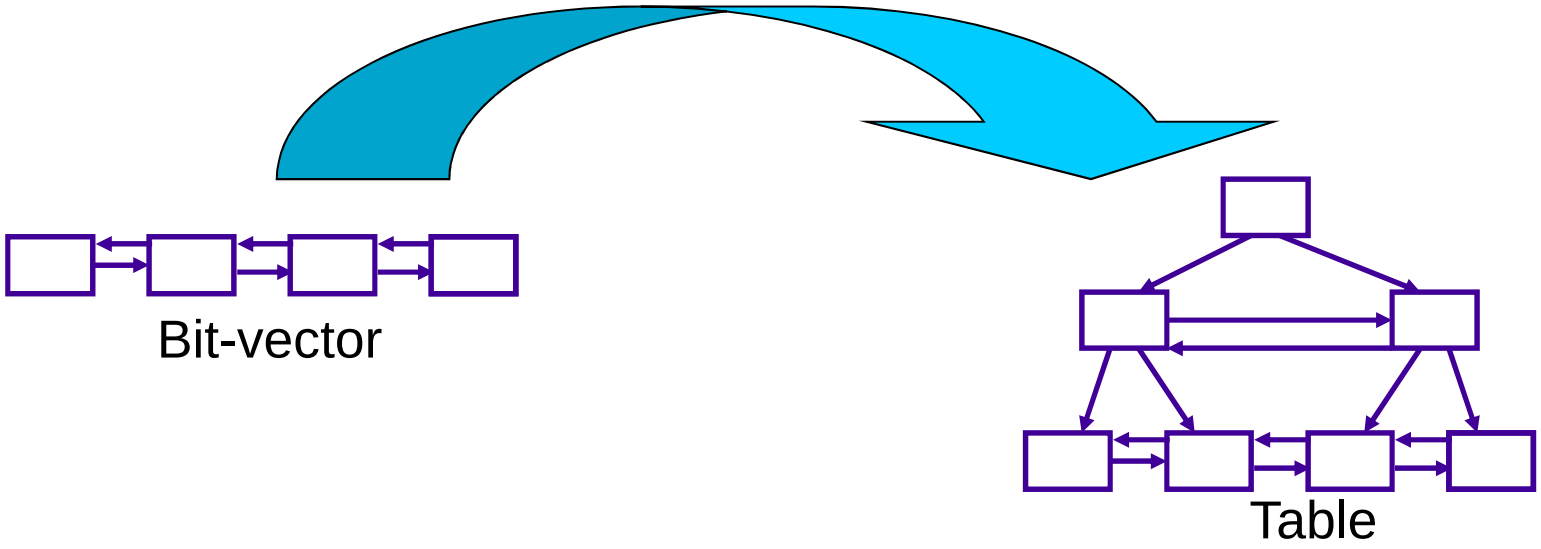
Within the database, table structure is a balanced tree (B^+-tree), whose nodes are stored as database pages



DB: Database structure in V5 and earlier



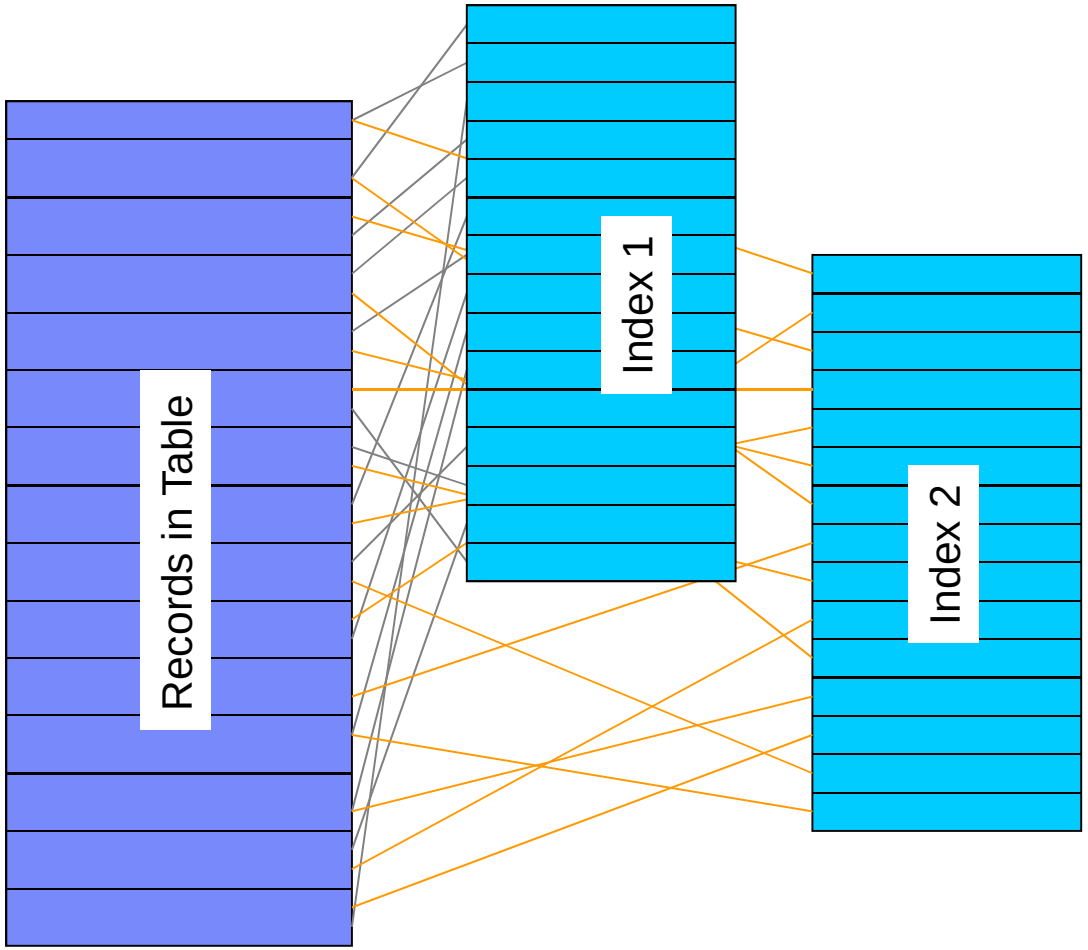
DB: Support for DISK in V6



- "Bit-vector" objects were special database objects used to track allocation for DISK storage pools.
- V6.x still supports DISK storage pools, but bit-vectors are eliminated.
  - Allocation and management is entirely managed from the database tables themselves.
  - No more "special" bit-vector database object needed or used.
- Bit-vectors first eliminated in 6.2.
  - Backfit to 6.1 in fix-pack 6.1.5.0.

DB: Table objects in V6

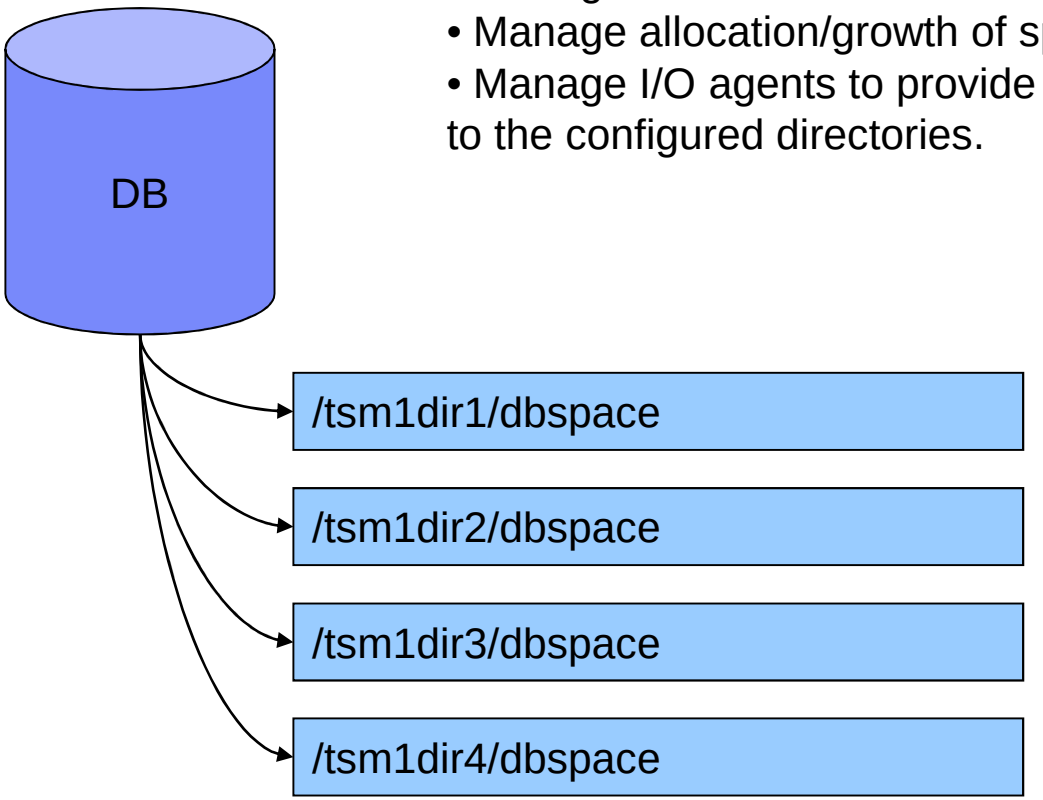
- Records in table are stored as needed and space available.
- Indices are used to manage ordering and access.
- Indices can cause duplication of data into a given index.
- Reorganization of tables versus indices are separate operations.
- Tables require at least one index as “PRIMARY”.
- Many tables have multiple indices implemented in order to support different access patterns and ultimately performance.



DB2 has a process called “runstats” that is run for tables. This analyzes the table (size, cardinality, dispersion) and the available indices for the table. This information is used during processing to build an “access plan” that is optimized based on this information. Runstats is managed automatically and not generally something that needs to be considered for management.

DB: Database space in V6

Database comprised of directories. The DB itself will:
- Manage data distribution across the directories.
- Manage allocation/growth of space within a given directory.
- Manage I/O agents to provide parallel read/write capabilities to the configured directories.

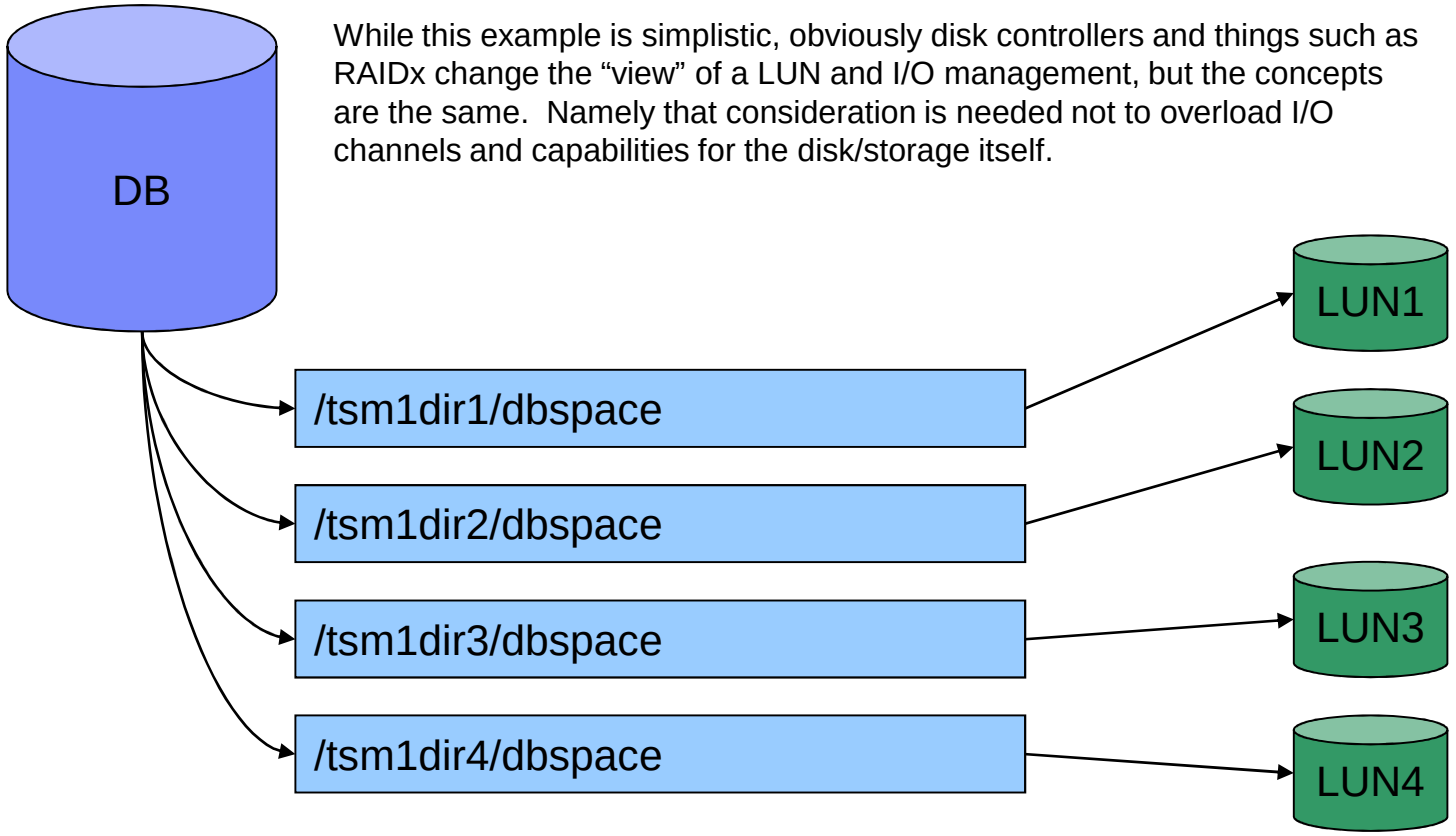


Performance tuning and recommendations similar to those for V5.x servers.

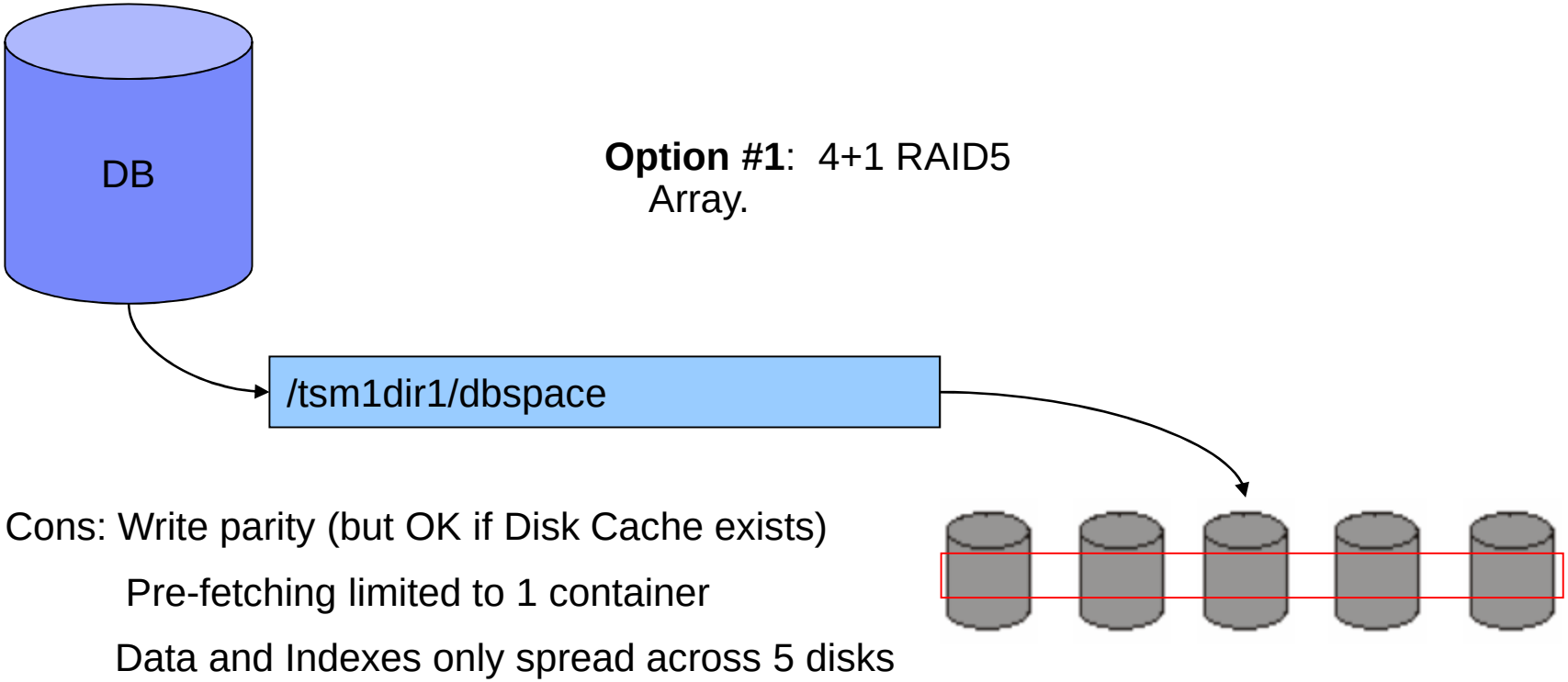
DB: Preferred LUN assignments

The following illustrates a preferred or optimal mapping of database directories to the underlying ‘physical’ disk that is being used. The key here being that I/O to a given directory has dedicated or appropriate I/O bandwidth to the underlying device to support the operations.

While this example is simplistic, obviously disk controllers and things such as RAIDx change the “view” of a LUN and I/O management, but the concepts are the same. Namely that consideration is needed not to overload I/O channels and capabilities for the disk/storage itself.



DB: A “Good” layout



Option #1: 4+1 RAID5 Array.

Cons: Write parity (but OK if Disk Cache exists)

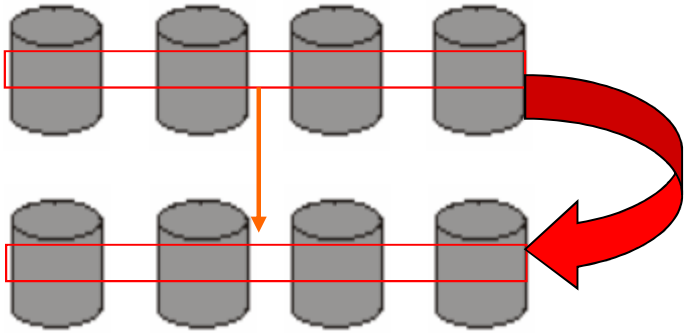
Pre-fetching limited to 1 container

Data and Indexes only spread across 5 disks

Recommended stripe size is 512 KB. Not all disk subsystems and vendors allow configuration of this value. When a disk subsystem allows this to be configured, consider using this value but also check with the disk vendor for any additional recommendations.

DB: A “Better” layout

Option #2: 4+4 RAID10 Array.



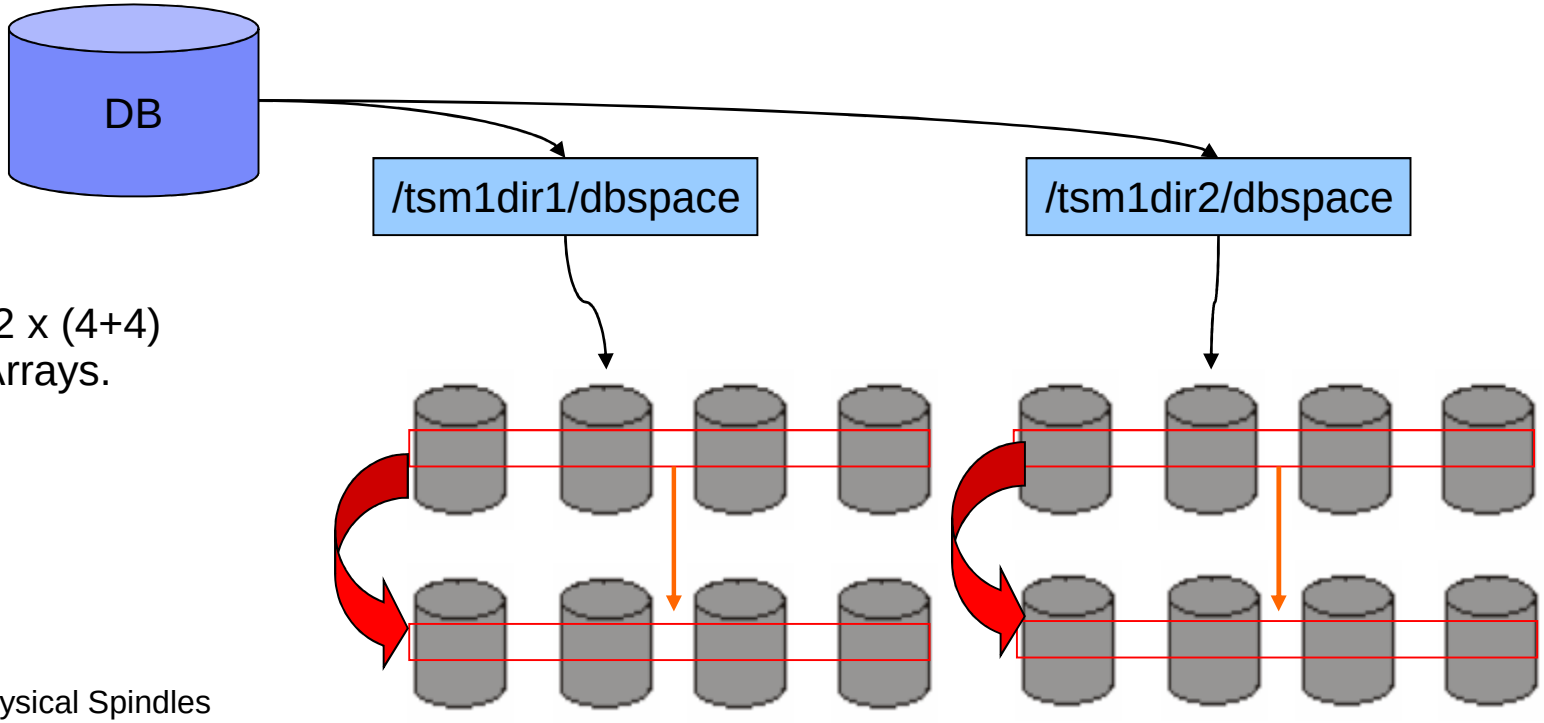
Pros: No parity write overhead

Faster reads (can read from either set of disks)

Cons: More Expensive

Recommended stripe size is 512 KB. Not all disk subsystems and vendors allow configuration of this value. When a disk subsystem allows this to be configured, consider using this value but also check with the disk vendor for any additional recommendations.

DB: An even “Better” layout



Option #3: 2 x (4+4) RAID10 Arrays.

Pros: Lots more physical Spindles

More Containers (more data is pre-fetched) (maybe could even add more than 2)

No parity write overhead

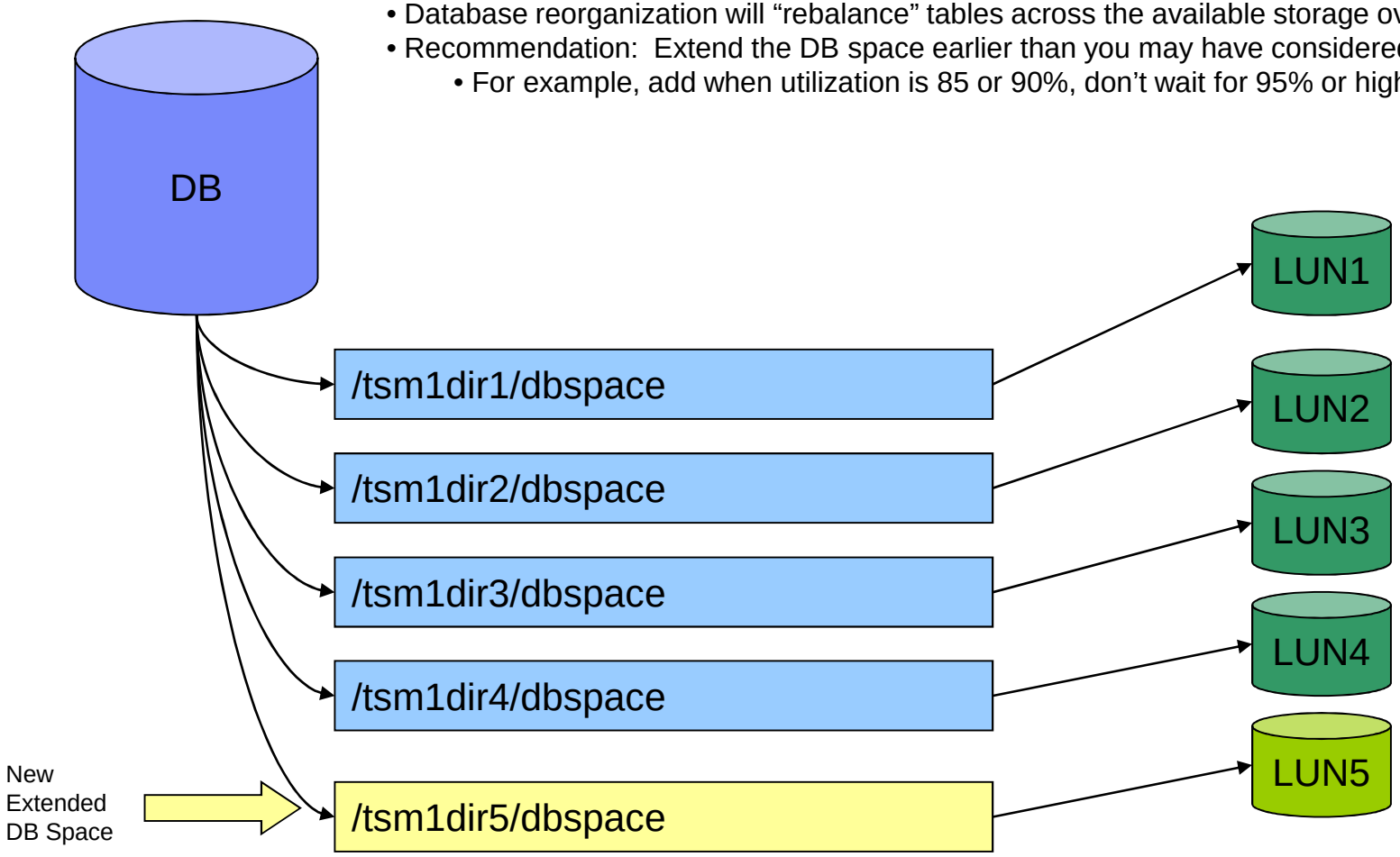
Faster reads, data more balanced across spindles

Cons: Much more Expensive

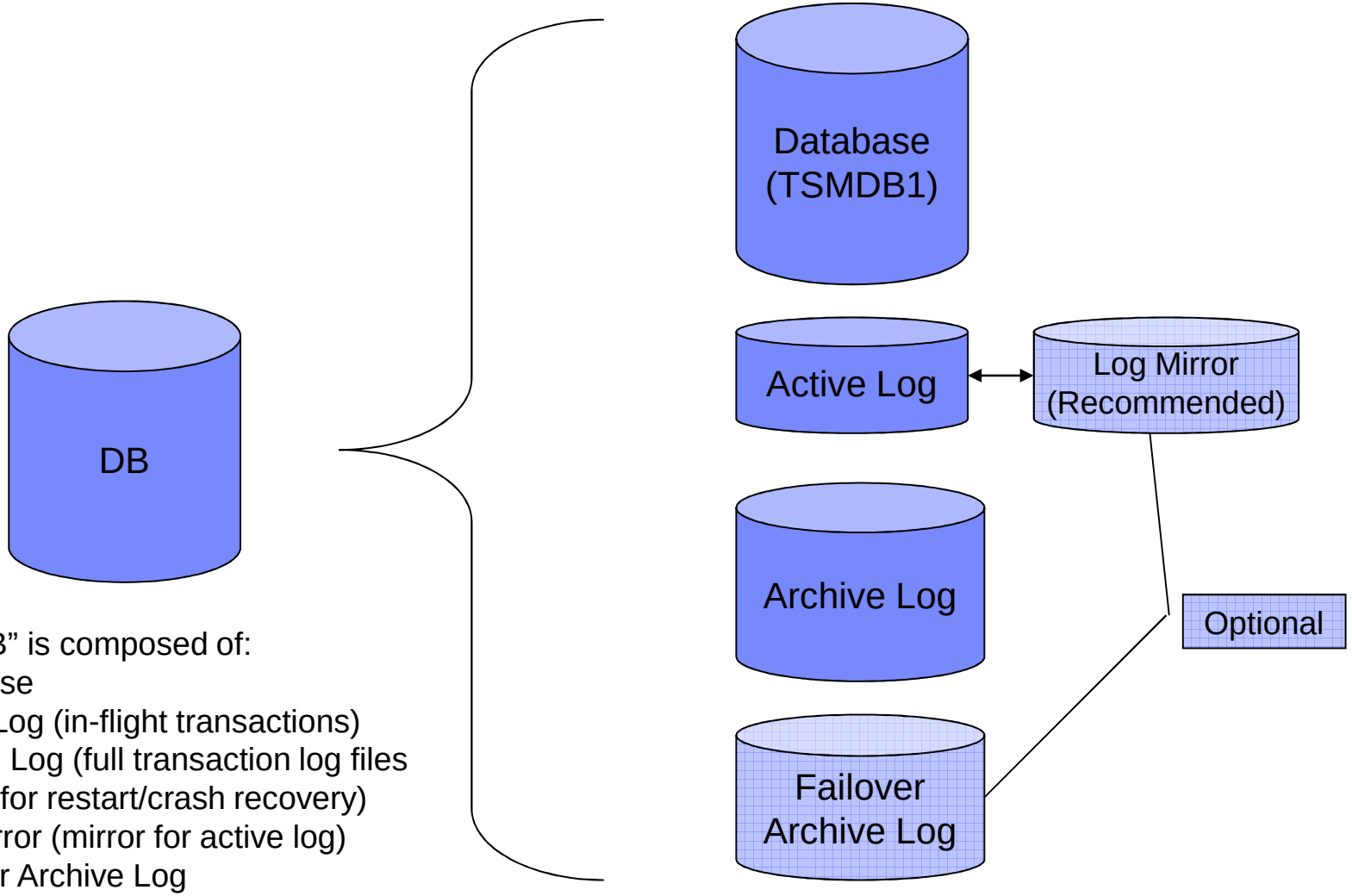
Recommended stripe size is 512 KB. Not all disk subsystems and vendors allow configuration of this value. When a disk subsystem allows this to be configured, consider using this value but also check with the disk vendor for any additional recommendations.

DB: Adding space and when is it used?

- Space may not be usable immediately.
  - Temporary space is anchored to existing table spaces and storage containers, it may not be able to utilize new space until system activity quiesces or the next server restart.
- Database reorganization will “rebalance” tables across the available storage over time.
- Recommendation: Extend the DB space earlier than you may have considered with V5.
  - For example, add when utilization is 85 or 90%, don’t wait for 95% or higher.



LOG: Composition



TSM “DB” is composed of:
- Database
- Active Log (in-flight transactions)
- Archive Log (full transaction log files retained for restart/crash recovery)
- Log Mirror (mirror for active log)
- Failover Archive Log

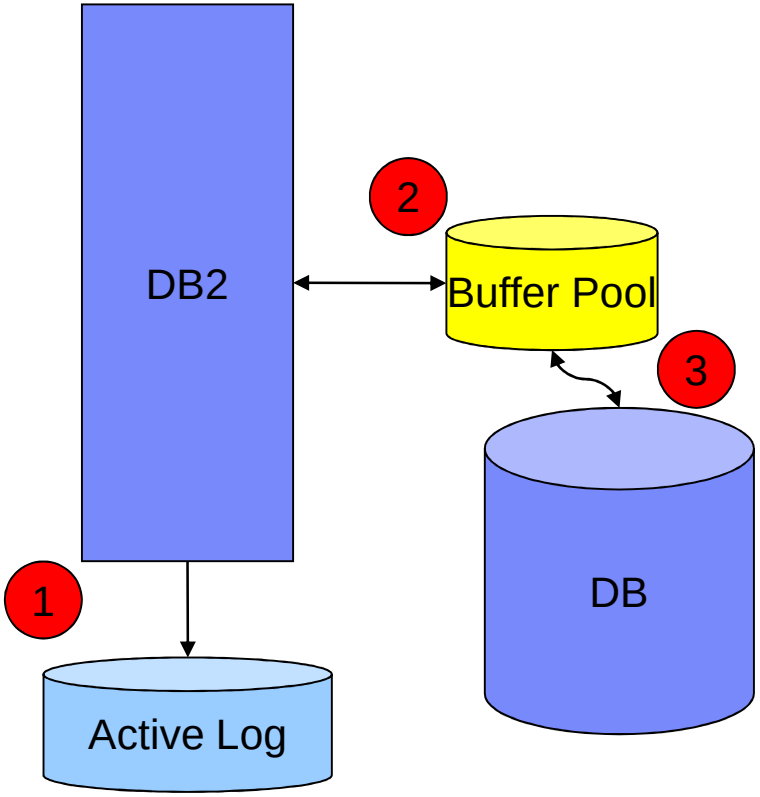
LOG: Active and archive log size minimum recommendations

Estimated Database Size	Server Deduplication Enabled?	Active Log Size (min)	Archive Log Size (min)
<= 1 TB	YES	32	96
<= 1 TB	NO	16	48
> 1 TB and <= 2 TB	YES	64	192
> 1 TB and <= 2 TB	NO	32	96

The recommendations shown are minimum or “starting” values.

- The active log is directly affected by “concurrent” workload such as peak current number of sessions performing a backup.
  - A peak concurrent workload of 250 sessions will not require as much active log space as a peak workload of 500 concurrent sessions.
  - Increasing the TXNGROUPMAX can require an increase in active log size to support the concurrent in-flight transactions.
- The archive log is affected by the total workload over time.
- Note that the maximum active log size is 128 GB.
  - To support large concurrent workloads, there is room for growth (higher active log values) compared to what is shown here for “minimum” values to consider.

LOG: Sequence of operations with the log



1. As changes are made to the database, these are first reflected in the active log. This provides “transaction consistency” to the database by allowing transactions to be replayed against the database during crash/restart recovery when needed.

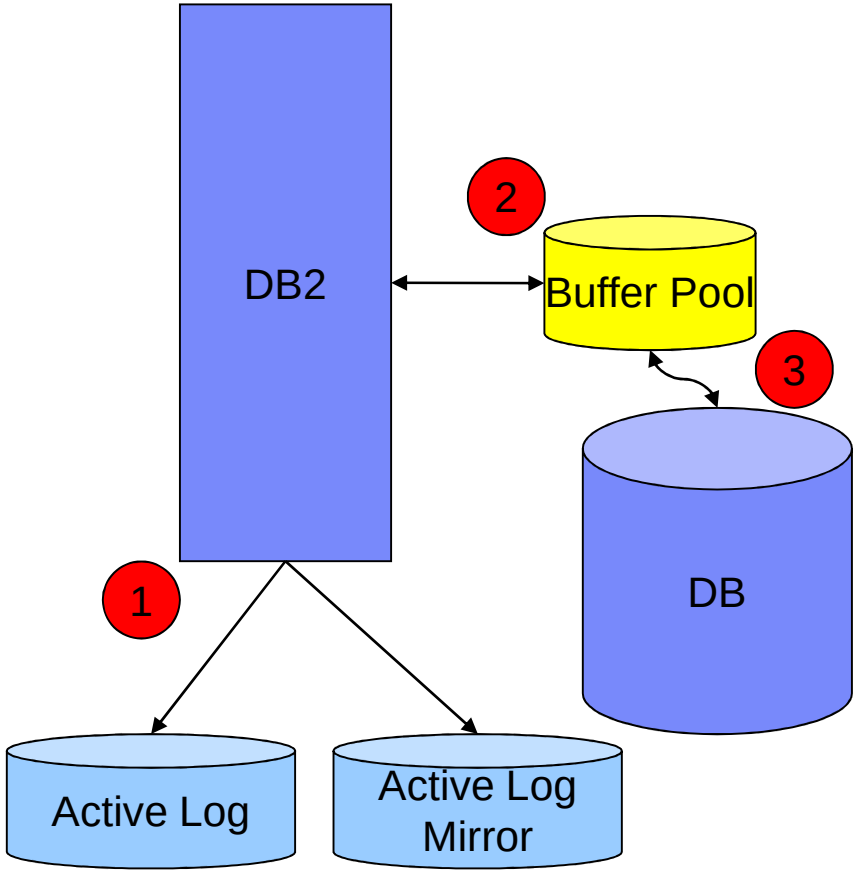
2. New, changed, and deleted database pages are managed through the bufferpool(s). This represents a “cache” in order to not couple database activity directly to I/O to-from the database itself.

   As such, the buffer pool relies on the active log data being written to stable storage as a “protection” for the database transactional consistency.

3. As changes are made to the database, these are first reflected in the active log. This provides “transaction consistency” to the database by allowing transactions to be replayed against the database during crash/restart recovery when needed.

Note: *For those who have been to previous Symposium and seen discussions on the V5 database and log, TSM used a “Write-Ahead” log. TSM V6, with DB2, also uses a write-ahead log the difference being that DB2 is now in control of it on TSM’s behalf.*

LOG: Why is mirroring of the active log important?

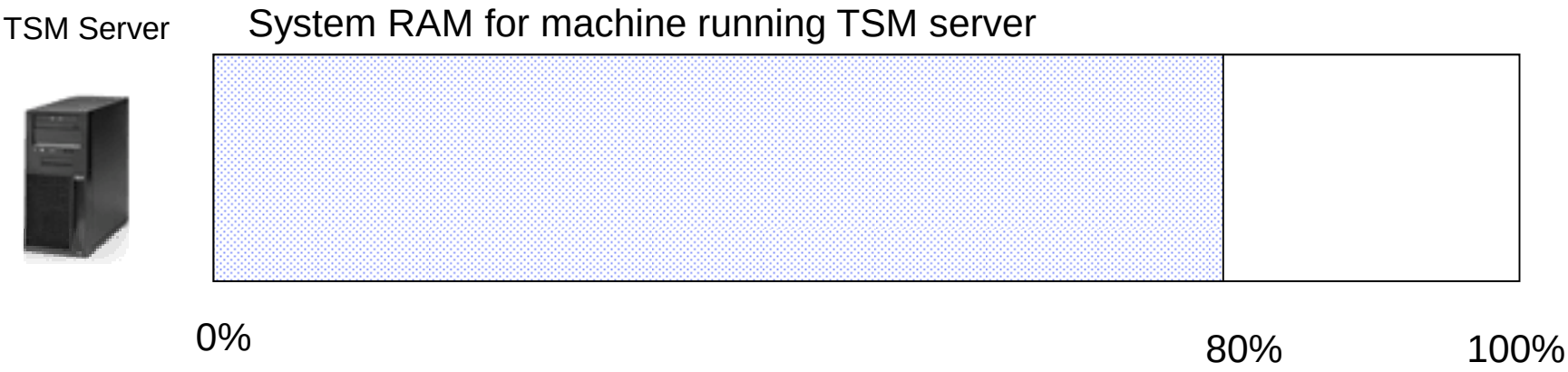


We have seen hardware issues where the write order of database pages to the actual hardware can be affected.

As a result of this, if the active log were affected (such as a partial write to the storage), this represents a single point of failure where the log data necessary to "reconcile" the database for transaction consistency is not available.

Having a mirror for the active log provides a higher degree of protection. The protection provided is making it less likely that the active log data is also affected by a hardware write issue.

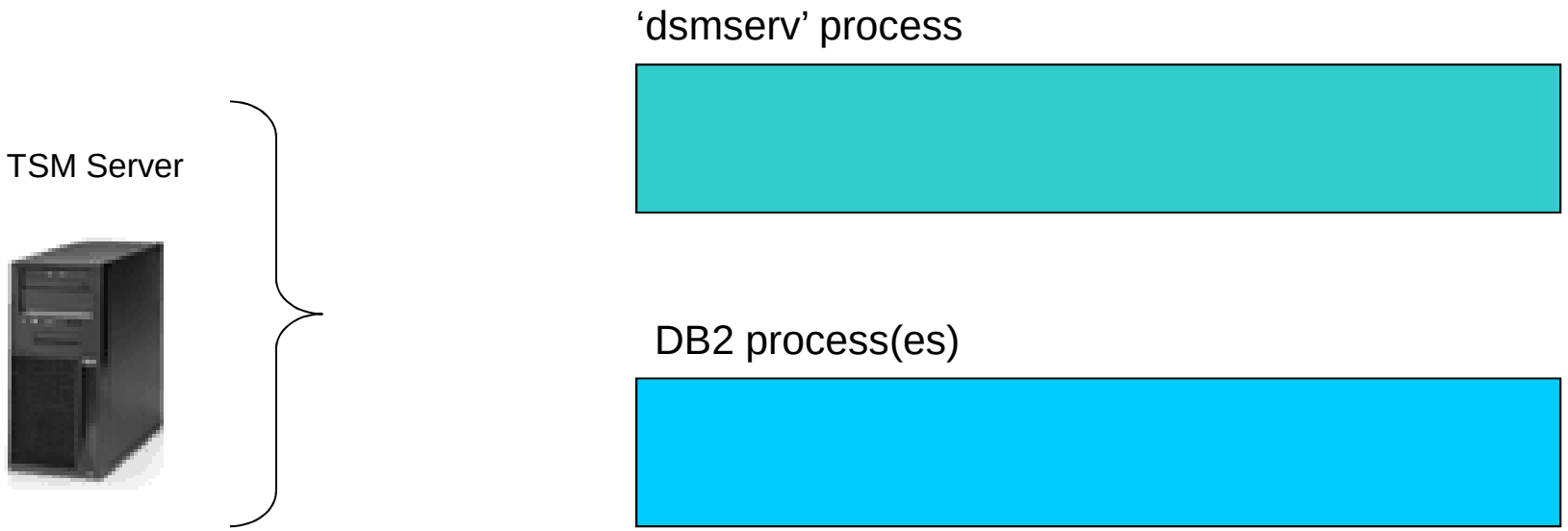
MEMORY: Managing memory used by a TSM server



The server option “DBMEMPERCENT” controls how much memory TSM will configure for use by DB2. The default is 80%. This means that:

- DB2 for this TSM server will use up to 80% of available RAM on the machine.
- Remaining 20% of RAM is left to the operating system and other tasks.
  - Note that the TSM server itself will compete for memory and the remaining 20% should consider that this is memory that the server itself may need.

MEMORY: View of memory for TSM V6.x server



A single "TSM" server decomposes to be the dsmserv process representing the TSM server and the corresponding DB2 processes. The primary DB2 process is db2sysc but there are others present for various purposes as well.

MEMORY: View of Memory for TSM V6.x Server

TSM Server

‘dsmserv’ process

Server Memory is used for:

- Data transfer buffers
- Session management
- Database communication and management.
- Normal application stuff like threads, control descriptors, transactions, lock lists, serialization constructs and such.

MEMORY: View of Memory for TSM V6.x Server

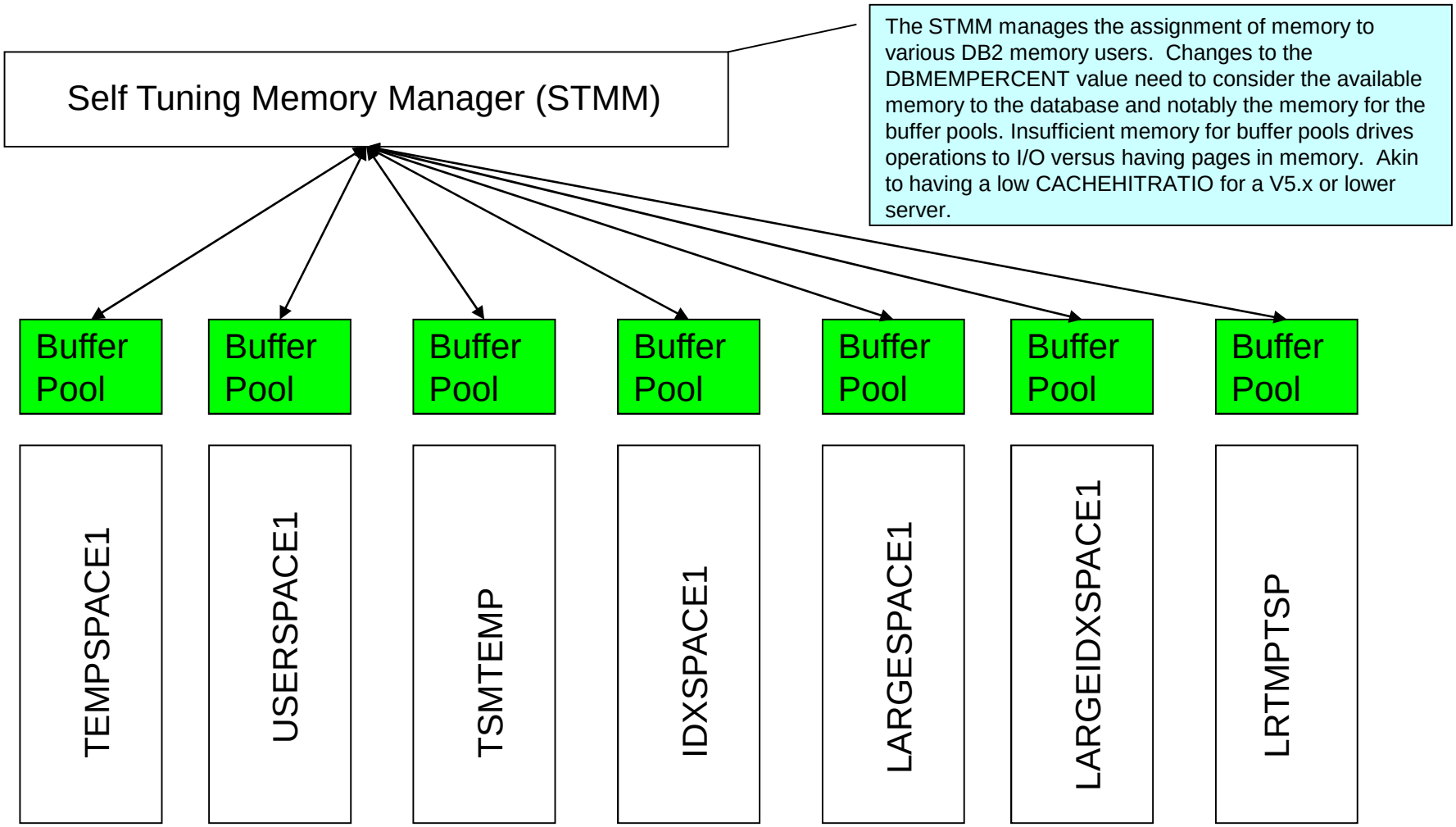
TSM Server

DB2 Process(es)

Database memory is used for:

- Buffer pools.
- Communication to/from applications (TSM).
- Transactions/Units of Work.
- Lock Lists.
- Application Heap (support for connections and activities).
- Sort Heap (support for in-memory sort operations for SQL statements).
- Statement Cache.
- Access plans.
- Other process “stuff” such as threads, control descriptors, and such.

MEMORY: Database Primary memory consumer(s)



MEMORY: Memory recommendations corresponding to DB size

Estimated Database Size	Server Deduplication Enabled?	RAM Available to TSM
<= 500 GB	YES	24
<= 500 GB	NO	16
<= 1 TB	YES	48
<= 1 TB	NO	32
<= 1.5 TB	YES	72
<= 1.5 TB	NO	48
<= 2 TB	YES	96
<= 2 TB	NO	64

The values shown here are larger than the “minimum” and “recommended” on the documented product requirements web page and those documented in the TSM Information Center.

The primary consideration is to plan for the memory requirements based on where your server is going and not where it has been.

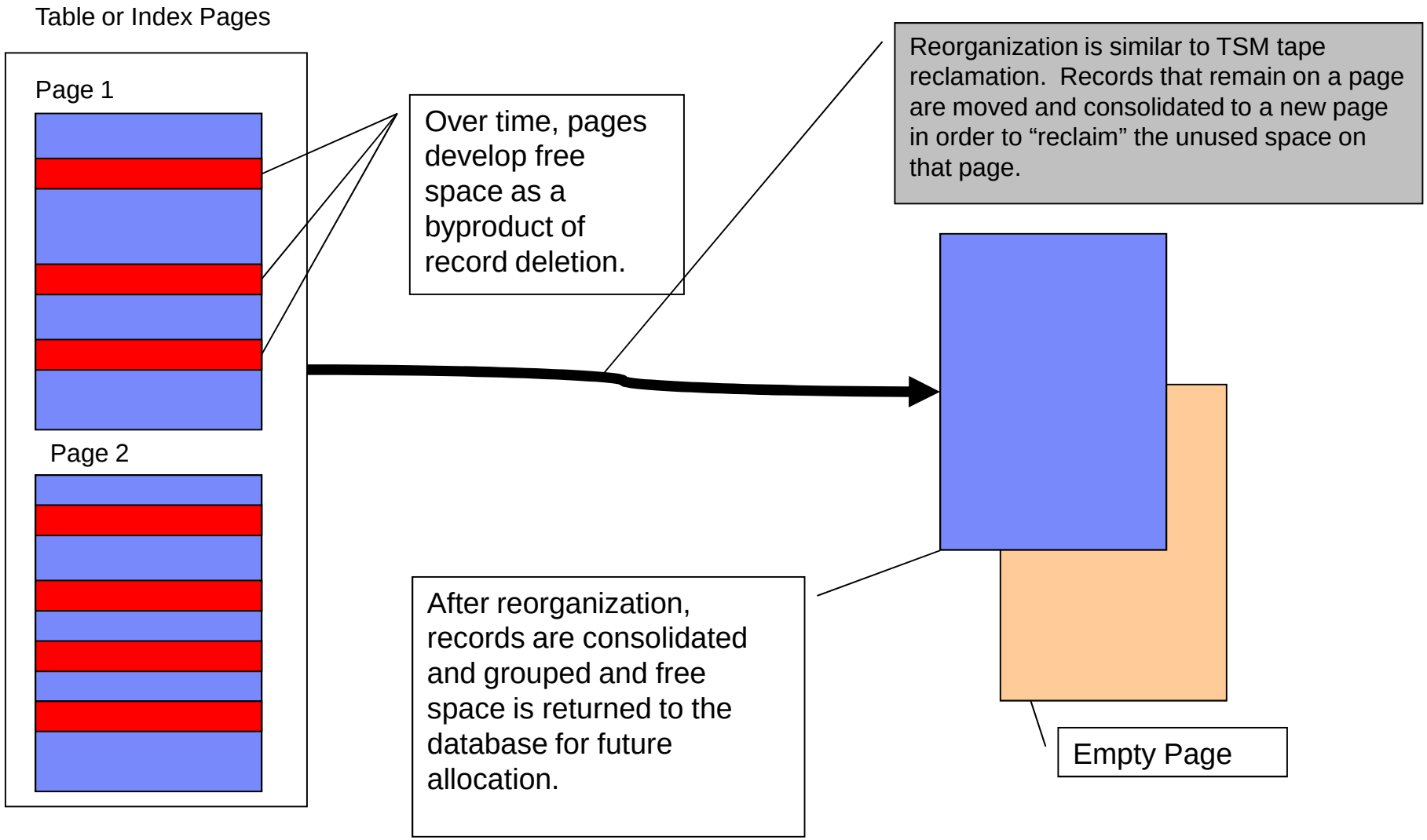
• Consider where your server will be in 6, 12, 18 months and plan to provision more memory to the machine in the same way that you are provisioning additional disk space.

• Memory directly affects performance and contention within the system, so growing memory as the server grows will help to position it to stay healthy.

MEMORY: Configuration Considerations

Machine	Number of TSM servers	Recommendation for DBMEMPERCENT
Single Physical Host	1	“default” – no change needed
Single Physical Host	>1	80 / number of servers For example, a 3 server environment would set 26. This results in nearly 80% (78) of RAM split between the three server instances with ~20% remaining to support the OS and other processes.
Vritualized Server such as multiple LPAR/WPAR on Single Physical Host	1 Server / LPAR	Consideration needed for: ▪ How many WPARS? ▪ How many TSM server instances running? ▪ Other memory intensive applications? For example, 4 WPARS each with a TSM server running could set a DBMEMPERCENT value in the range of 10-15. (assuming that each WPAR was assigned up to 100% of the available RAM). The key to consider is that the memory available to DB2 needs to also consider the other memory requirements of the “virtualized” system that it runs in. And how memory or resources are available and limited to the virtualized system.

REORGANIZATION: What does reorganization do?



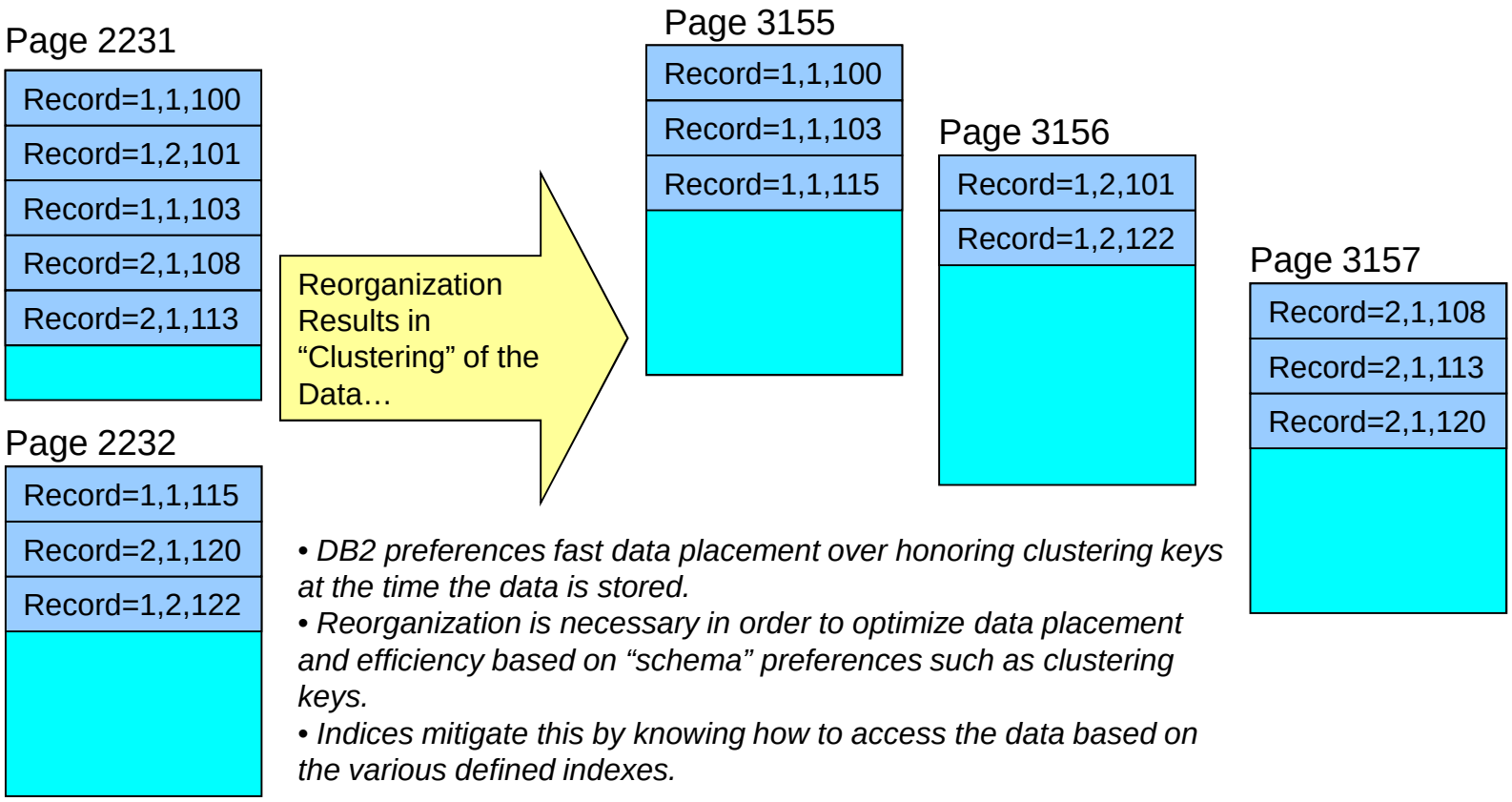
REORGANIZATION: Advantages of reorganization

- Allowing reorganization to run is beneficial to the TSM server:
  - Optimize data placement
    - Query efficiency with less I/O to DB.
    - Higher “hit ratio” for data in buffer pool.
  - Optimize space by returning partially used database pages as empty and available for re-allocation.

REORGANIZATION: Data Placement

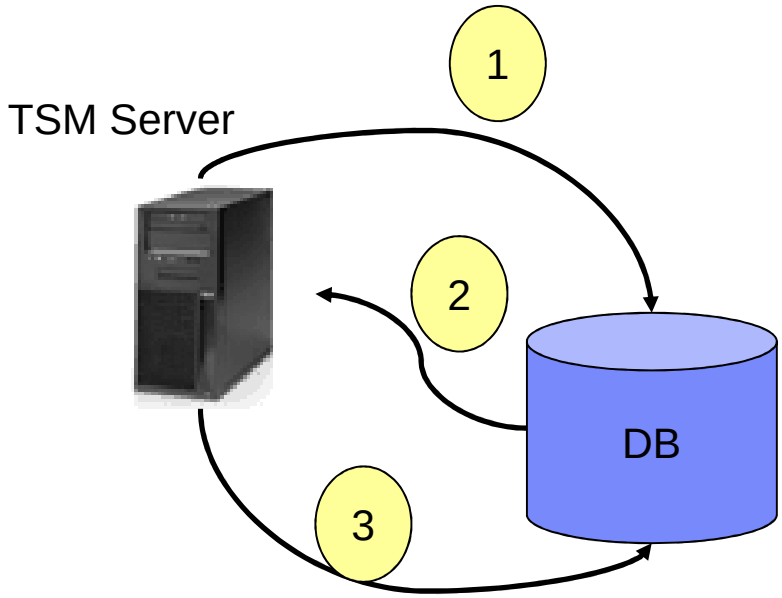
For this illustration, a database row is composed of X, Y, ZZZ where X is the first key for this table, Y is the second key, and ZZZ is an identifier. Those familiar with TSM database layout can think of these in terms of nodeId, filespaceId, objectId.

Also consider a “clustering key” for this table to be X and Y. Meaning the preference is to group this data first based on the value of column X and second based on the value of column Y.



- *DB2 preferences fast data placement over honoring clustering keys at the time the data is stored.*
- *Reorganization is necessary in order to optimize data placement and efficiency based on “schema” preferences such as clustering keys.*
- *Indices mitigate this by knowing how to access the data based on the various defined indexes.*

REORGANIZATION: How does TSM do this?



1 Server requests information from DB2 about tables that are eligible for reorganization.

2 DB2 responds with information about which tables are eligible for reorganization.

3 Server initiates reorganization for one of the candidate tables.

A few observations about reorganization:

- *TSM only runs one table reorganization at a time.*
- *Table reorganization operations may be paused by either the server or DB2 for various reasons.*
  - *DB2 may elect to pause based on workload.*
  - *TSM may elect to pause based on operations, such as running DB Backup.*
- *If TSM sees a paused reorganization, we will restart that prior to trying to start reorganization for a new/different table.*

REORGANIZATION: When does it run?

< 6.1.5.0 < 6.2.3.0	Table reorganization a candidate to be run ANY time. (Demand based over 24 hour window)
>=6.1.5.0 >=6.2.3.0	Configurable to run in a “window”. Default window is 24 hours in order to match prior behavior.

REORGANIZATION: Managing the reorganization window

- Reorganization of tables can be turned ON or OFF by using:
  - Set the server option: ALLOWREORGTABLE
    - YES – allow table reorganization to occur. (DEFAULT)
    - NO – do not allow table reorganization to occur.
- With reorganization enabled (ALLOWREORGTABLE=YES), the window is managed by:
  - REORGBEGINTIME
    - This is specified as a time of day in the format hh:mm.

      "hh" is hours and may be specified in the range of 0 to 23.

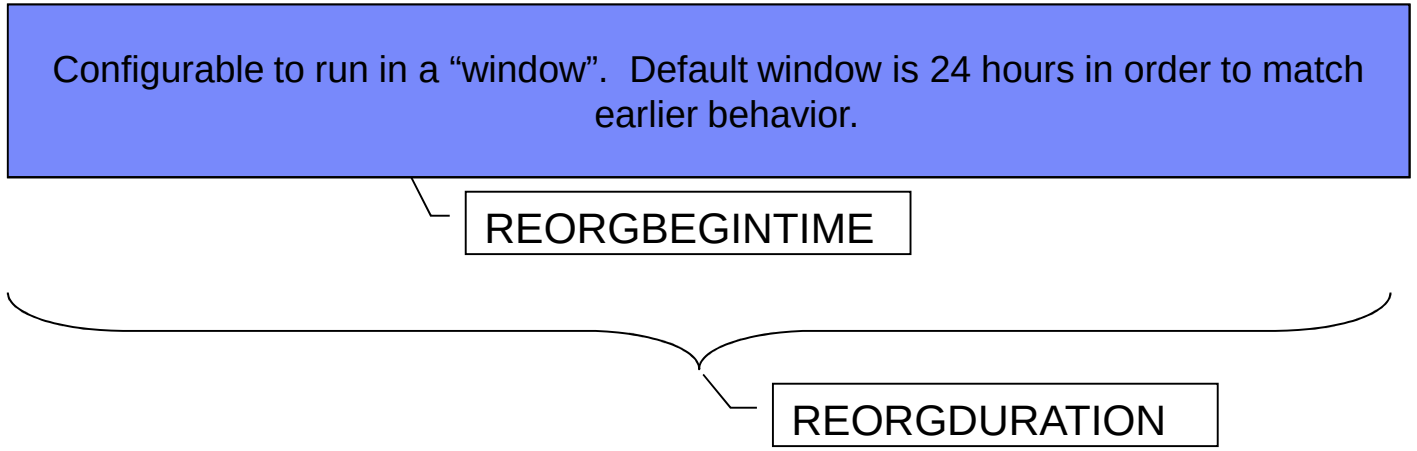
      "mm" is minutes and may be specified in the range of 0 to 59.

      For example, 06:30 denotes 6:30 in the morning (AM).
  - REORGDURATION
    - This is the duration, in hours, from the REORGBEGINTIME that reorganization is allowed to run.
    - The default for this is 24, this mimics the pre-existing behavior of allowing reorganization to run throughout the entire day.
  - If there are periods of contention or workload peaks that exhaust the available resources for the server, consider changing the REORGBEGINTIME and REORGDURATION in order to prevent reorganization from running during this time.
  - If a table reorganization is still running once the REORGDURATION has elapsed, the reorganization will be paused and resumed during the next "window".
    - Note that an index reorganization will run until completion. It will not pause at the end of the REORGDURATION window.

REORGANIZATION: Reorganization Window

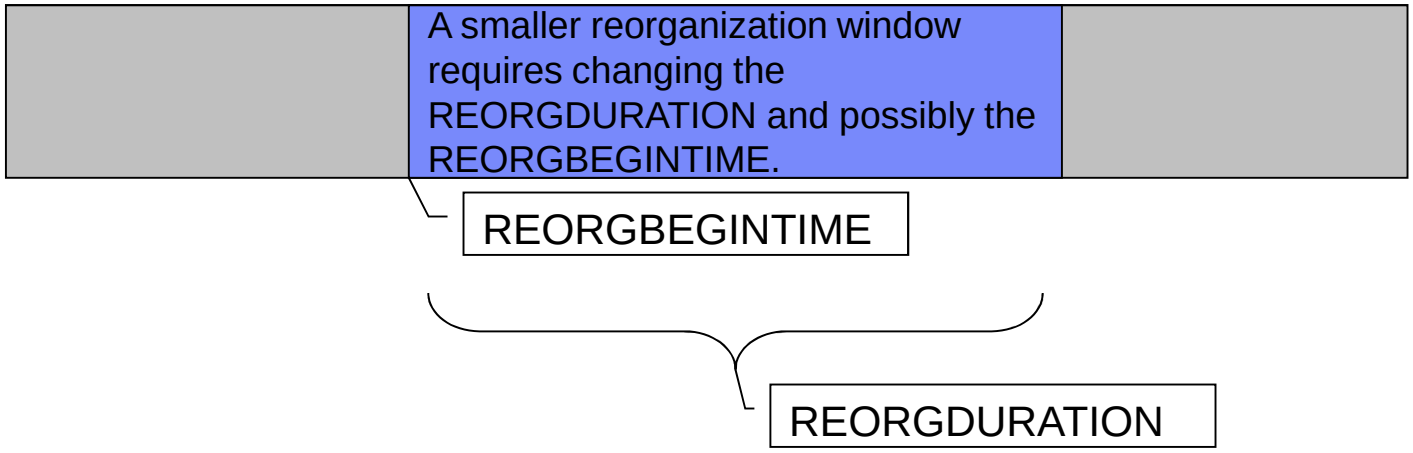
Default:

>=6.1.5.0
>=6.2.3.0



Using an explicit window:

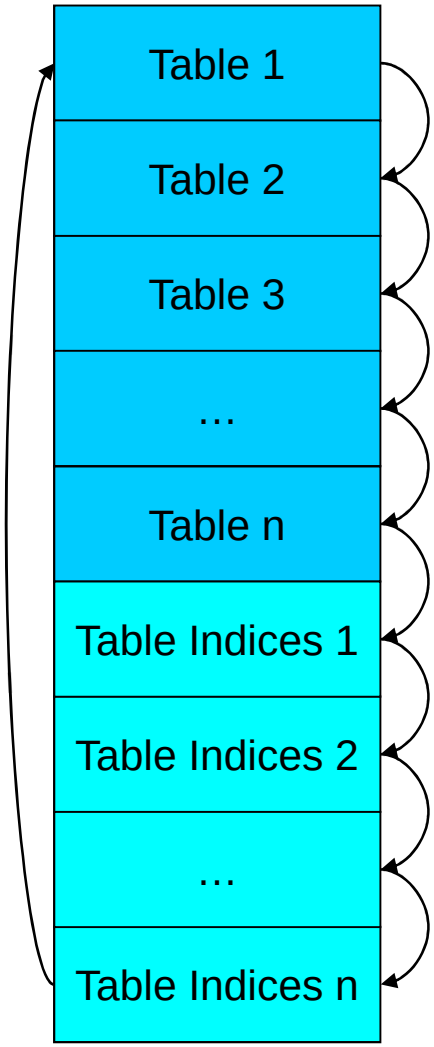
>=6.1.5.0
>=6.2.3.0



REORGANIZATION: Reorganization for Indices

- Reorganization of indices is managed differently than tables.
- Index reorganization requires the following option to be enabled:
  - ALLOWREORGINDEX
    - YES – allow reorganization of indices.
    - NO – do not allow reorganization of indices. (DEFAULT)
- Why is this managed separately from table reorganization?
  - Index reorganization does not have the flexibility that table reorganization provides.
    - Can not be paused.
    - All indices for a given table will be done together.
  - Index reorganization has been seen to be most beneficial for servers that are configured to use TSM deduplication.
  - Index reorganization requires that the server option **DB_DB2_KEEPTABLELOCK** is set to “NO”.
    - The default for this is YES.
    - KEEPTABLELOCK set to YES works appropriately for table reorganization but for index reorganization this may cause problems. (IC77773)

REORGANIZATION: Table and index reorganization sequencing



- Only a single reorganization (table or index) will be run at a given time.
- Table reorganizations are considered before index reorganization.
- Once a table or index is reorganized, it is disqualified from a subsequent reorganization for 20 days. (Avoid high-activity tables from “monopolizing” all the reorganization processing time.)
- Once we finishing the list (last index), we move back to the top and start again with the tables.

REORGANIZATION: Recommendations for index reorganization

- Index reorganization should be done periodically.
  - Enable it by setting the server options:
    - ALLOWREORGINDEX YES
    - DB_DB2_KEEPTABLELOCK NO
  - Allow index reorganizations to be accomplished over a few days or week depending upon the time and amount of reorganization that is needed.
  - Once index reorganization is completed, disable index reorganization by setting the server options:
    - ALLOWREORGINDEX NO
    - DB_DB2_KEEPTABLELOCK YES
- How frequently should this be done:
  - If using TSM deduplication:
    - Consider index reorganization monthly.
  - If not using TSM deduplication:
    - Consider index reorganization once or twice for a year.
- See the “Useful Links” at the end of this presentation for a link to the latest information on reorganization.
  - Technote updated periodically to reflect:
    - New or changing best practice recommendations.
    - Details on fixes and known issues affecting reorganization.

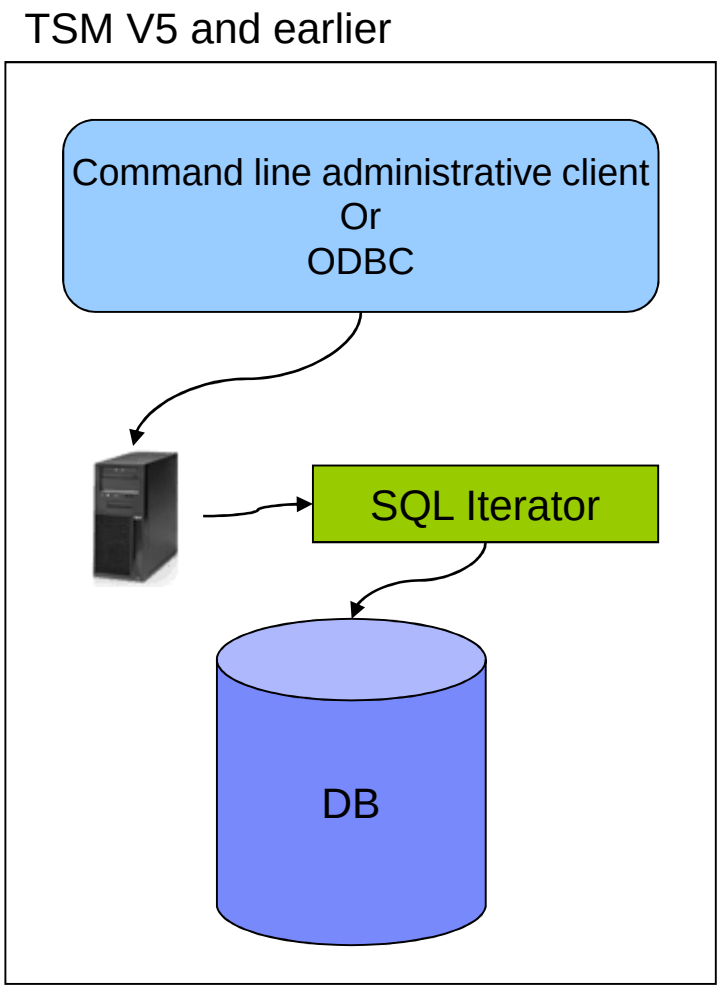
SQL: Administrative interfaces to the server

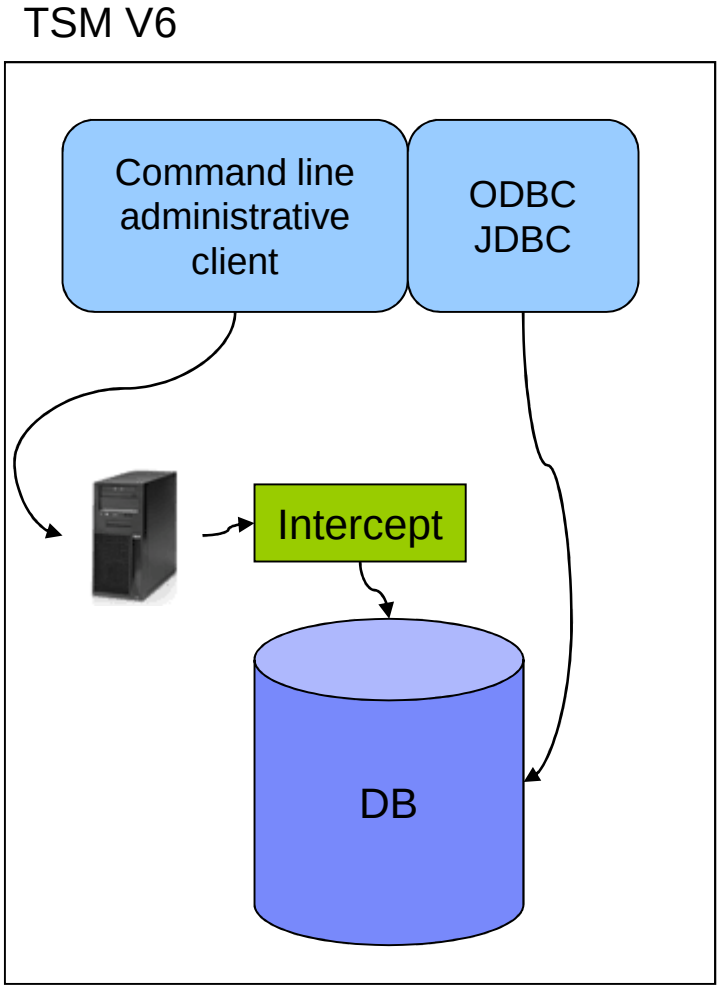
TSM Server

Command Based:
- Server console
- Command line administrative client
- Web based through Administration Center

SQL Based:
- SQL commands through command line administrative client (read only)
- SQL through ODBC/JDBC connector

SQL: Overview of SQL engine in TSM server





SQL: V5 iterator versus V6 intercept

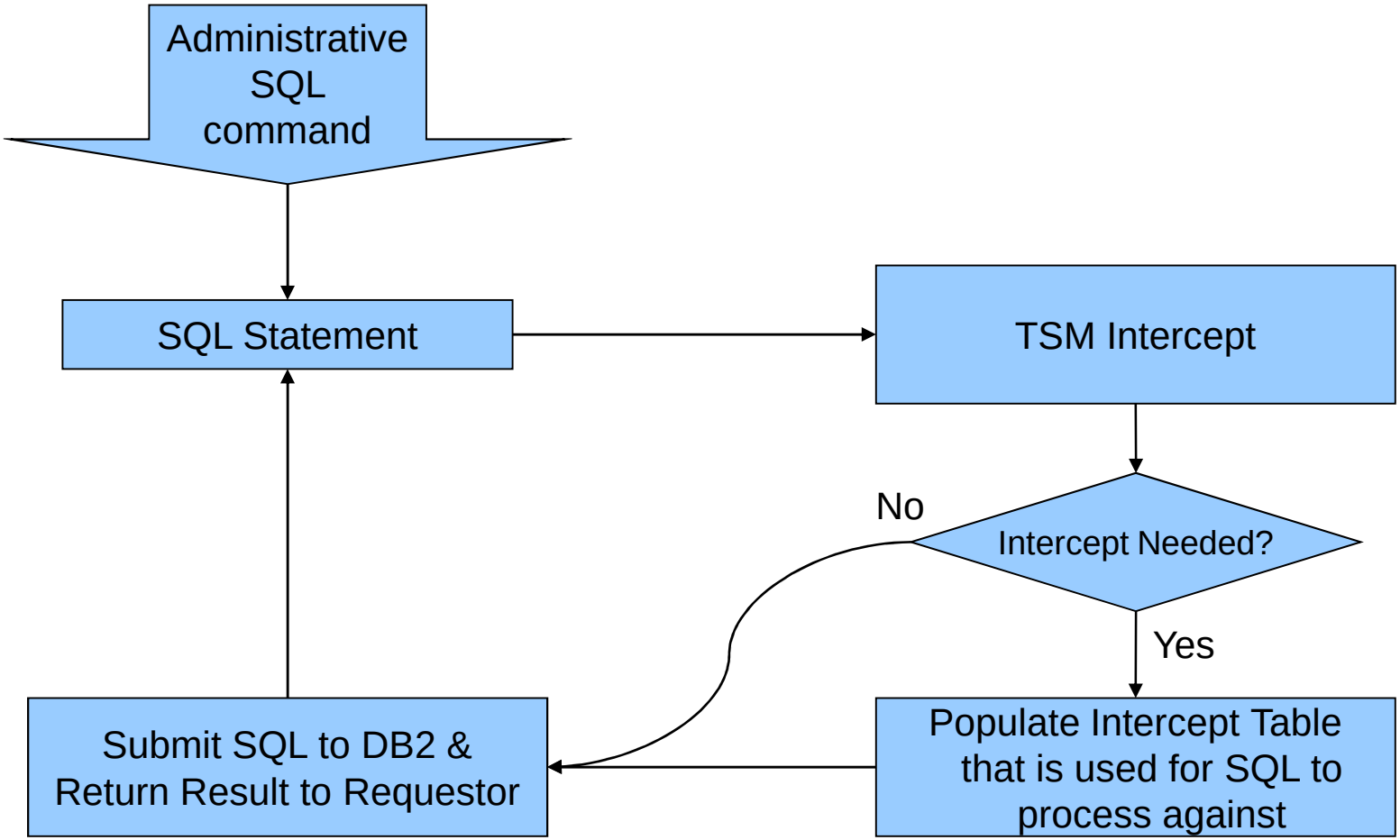
- The “SQL Iterator” in V5 and earlier provided:
  - DB tables implemented as B+ trees, so no true relational and SQL semantics.
  - Iterator “virtualized” B+ tree tables and in-memory constructs (in some cases) to make them available through limited capability SQL interface.
  - Data in some cases was “transformed” from what was actually in the database to human readable form.

- SQL Intercept in V6 server provides SQL table name translation
  - Some of the SQL table names collide with the actual database table name.
    - Not able to access the table in the database directly because “format” of the data does not match what customers expect.
    - Table virtualized through V5 SQL may not have “matched” column for column.
  - Some of the SQL tables are staged to transitory database tables.

SQL: V6 SQL interpretation

- In V5, SQL statements were parsed and syntax enforced by TSM.
  - Limited SQL support.
  - “Nuances” compared to industry standard SQL capabilities.
- In V6, SQL statements are:
  - Minimally processed on the TSM server.
    - Cases where we have to translate table names and such to accommodate mappings of the administrative SQL table names to actual table names.
  - SQL parsed and syntax enforced by DB2.
    - Provides complete and robust SQL capability.
    - SQL capabilities in-line with industry standards and expectations.
  - SQL through the administrative command interface is always enforced to be read-only.
    - Transactions (unit of work) is always rolled back.
    - Submit only SELECT statements.

SQL: V6 Intercept



SQL: V6 Intercepted SQL tables

- Admins
- Client_schedules
- Dbspace
- Devclasses
- Drives
- Events
- Filespaces
- Libraries
- Libvolumes
- Licenses
- Media
- Nodes
- Options
- Scripts
- Server_group
- Sessions
- Stgpools
- Volumes
- Status
- Profiles
- Subscriptions
- Clientopts
- Mgmtclasses
- Domains
- Restores
- Processes
- Drmedia
- Paths
- Cloptsets
- Drmsrpf
- Drmtrpf
- Drmstanza
- Vfsmappings
- San
- Shredstatus
- Script_names
- License_details
- Auditocc

Intercepted tables represent:
- *Complex data transformations where what is in the database does not readily match "human readable" form.*
- *Complex table relationships where data comes from multiple underlying tables.*
- *In memory constructs that do not exist in the database such as sessions, processes, and others.*

SQL: ODBC/JDBC

- ODBC provides access to the entire database.
  - Tables not previously accessible through legacy interface are now visible.
  - Care should be taken to consider the format and content of the columns.
    - TSM does not document internal data representation as items are stored in the database.
    - Many fields have embedded structures, bit-flag structures and such.
    - The “usability” outboard from the TSM application itself may be limited.
  - SQL processing against the “intercept” staging tables will only be current as of the last time an administrative SQL statement was executed against that intercept table.
    - Execute directly through administrative interface if needing the “most up to date” view of this data is necessary.
    - Schedule administrative client to periodically issue select against the intercept table in question in order to re-populate it with some frequency.
- Any ODBC/JDBC needs to be considered and implemented as read-only.
  - Do not ALTER, UPDATE, CREATE, DELETE, or perform any other “destructive” commands through an ODBC/JDBC interface direct to the server.

UPGRADE: What are the options?

- TSM Upgrade Utility
  - SUMMARY: Two phase utility capable of using media (tape) or going direct from V5 server to V6 server using TCP/IP. The database entries from the V5 server are extracted using DSMUPGRD EXTRACTDB utility. And entries are inserted into V6 server using DSMSERV INSERTDB utility.
  - PRO:
    - V5 server is upgraded to V6.
  - CON:
    - Server downtime required while the EXTRACT and INSERT are being performed.
- TSM Upgrade Utility Hybrid
  - SUMMARY: A procedure has been documented for using EXPORT/IMPORT in conjunction with the upgrade utility.
  - PRO:
    - Significantly reduced downtime.
  - CON:
    - Still a little downtime corresponding to when the EXTRACTDB is performed.
    - More complicated as managing the EXPORT/IMPORT is now needed.
- Crossover workload
  - SUMMARY: Deploy new V6 servers and leave the existing V5 servers in place. Generally in this model we see client workloads “switched” to the V6 servers. And also some EXPORT/IMPORT to manage things like HSM or archive data.
  - PRO:
    - No downtime.
    - Allows for server consolidation. For example 2 or 3 V5.x servers consolidated into a single V6 server.
  - CON:
    - In the short-term, more servers to manage.
    - HSM/Archive data need to be evaluated and handled using EXPORT/IMPORT.
- Other Possibilities:
  - IBM services
  - Butterfly Storage Migration, a business partner offering capability of managing the transition of data from TSM V5 to TSM V6.
  - PRO:
    - More options.
    - Expertise and experience in the area of upgrading TSM.
  - CON:
    - Additional cost for services and products.

What have we seen customers do?
- *TSM Upgrade Utility*
- *Crossover workload – consolidate 1 or more V5 servers to a new V6 server.*

Conclusion

Thank You

A few useful links…

- V6 Deployment best practices:
  http://www-01.ibm.com/support/docview.wss?uid=swg21421060
- Database Reorganization:
  http://www-01.ibm.com/support/docview.wss?uid=swg21452146
- Memory requirements for V6:
  http://www-01.ibm.com/support/docview.wss?uid=swg21450229
- TSM configured for HADR:
  http://www.ibm.com/developerworks/wikis/display/tivolistoragemanager/Electronic+vaulting+using+deduplicated+remote+copy+storage+pools
- TSM Administrative SQL:
  http://www-01.ibm.com/support/docview.wss?uid=swg21380830
- Tivoli Storage Manager Version 6 Hybrid Upgrade Migration Method
  https://www.ibm.com/developerworks/wikis/display/tivolistoragemanager/Tivoli+Storage+Manager+Version+6+Hybrid+Upgrade+Migration+Method
- Migrating from zOS V5 to AIX, Windows, HP-UX, Linux, or Solaris V6
  https://w3.tap.ibm.com/w3ki07/download/attachments/600000058691/TSMmigration_white_paper_061009.pdf?version=1