IDUG
Leading the DB2 User Community since 1988

IDUG DB2 North American Tech Conference
Anaheim, California | April 30 - May 4, 2017

#IDUGDB2

DB2 Backup and Recovery Best Practices

Dale McInnis
IBM Canada Ltd.

Session Code: XD17
Alternate Session | Platform: Linux, UNIX and Windows

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Objectives

Objective 1: Review recovery features and functions

Objective 2: How can some of the lesser know features help you?

Objective 3: What is available to help isolate performance challenges in the area of recovery?

Objective 4: Are traditional backups still required?

Objective 5: What are real live customers going?

Why am I backing up?

- Hardware / Software failure protection
  - Much better methods available
    - H/W Clusters, HADR, Q Repl, CDC, …
- Moving to a new platform (H/W or OS)
  - If the same OS then consider HADR
  - If different OS then consider logical replication
- Populate a QA/Dev system
  - If the entire DB is not required consider Transportable Schemas or the REBUILD option on restore
- Logical Protection
  - Someone messed up the data and you have to roll it back
  - If error is limited in scope them perhaps you can use DB2 Tooling
    - Recovery Export
    - High Performance Unload

What types of backup should I use?

Types of backups

- Traditional DB2 Backup
  - Full, incremental or delta
- Snapshot backups
- Offloaded Snapshot backups
- Non-traditional
  - Q Repl
  - HADR
  - …
- My rule of thumb
  - If the elapsed time for the backup exceeds 6-8 hours (after tuning) then strongly consider an snapshot backup

DB2 Backup/Restore Supported Devices

Random Access Devices, e.g. HDD, SSD, local or network attached

Sequential Access Devices, e.g. Tape Library

Storage Manager, e.g. TSM/Spectrum Protect

Storage array snapshots, e.g. IBM Flashcopy

DB2 Backup versus File System Level Copy

	DB2 Backup	File System Copy
Online Support	Yes	No – data will be inconsistent
PIT Recovery	Yes	DB offline during copy and use db2rfpen after restoring all files
Subset recovery	Yes	No – meta data must be synced up with user data during recovery
Incremental Backup	Yes – page level	Yes – file level
Online recovery	Yes	No – unable to take table spaces offline or sync up with the meta data
Redirected Recovery	Yes	No – DB2 meta data contains pointers to physical file locations
Recording of event	Yes – DB2 history file Used by automatic restore	No history accessible by DB2

Agenda

- Overview
- Technology Review
  - What's new
  - What could help you become a super star
- Performance Tuning
- Recommendations

Autonomics

Both backup and restore are self tuning

- If not specified the following settings will be computed
  - Parallelism
  - Buffer size – capped to 4097 x 4K pages
  - # of buffers
- All setting are dependent largely on the following settings:
  - UTIL_HEAP_SZ
  - # of CPUs
  - # of table spaces
  - Extent size
  - Page size
- Autonomics do handle all possible options
  - Compression
  - Data deduplication
  - …

How can I make backup run faster?

- **Distribute your data evenly across the tablespaces**
- **Let the DB tune the backup parameters**
  - Ensure you heap size is large enough (util_heap_sz 50000)
- **If backing up to a file system create multiple objects**
  - If the target device is a raid 5 file system creating multiple backup objects will speed up the process
  - Backup db sample to /mydb/backup, /mydb/backup, /mydb/backup, …
  - For TSM – open multiple sessions

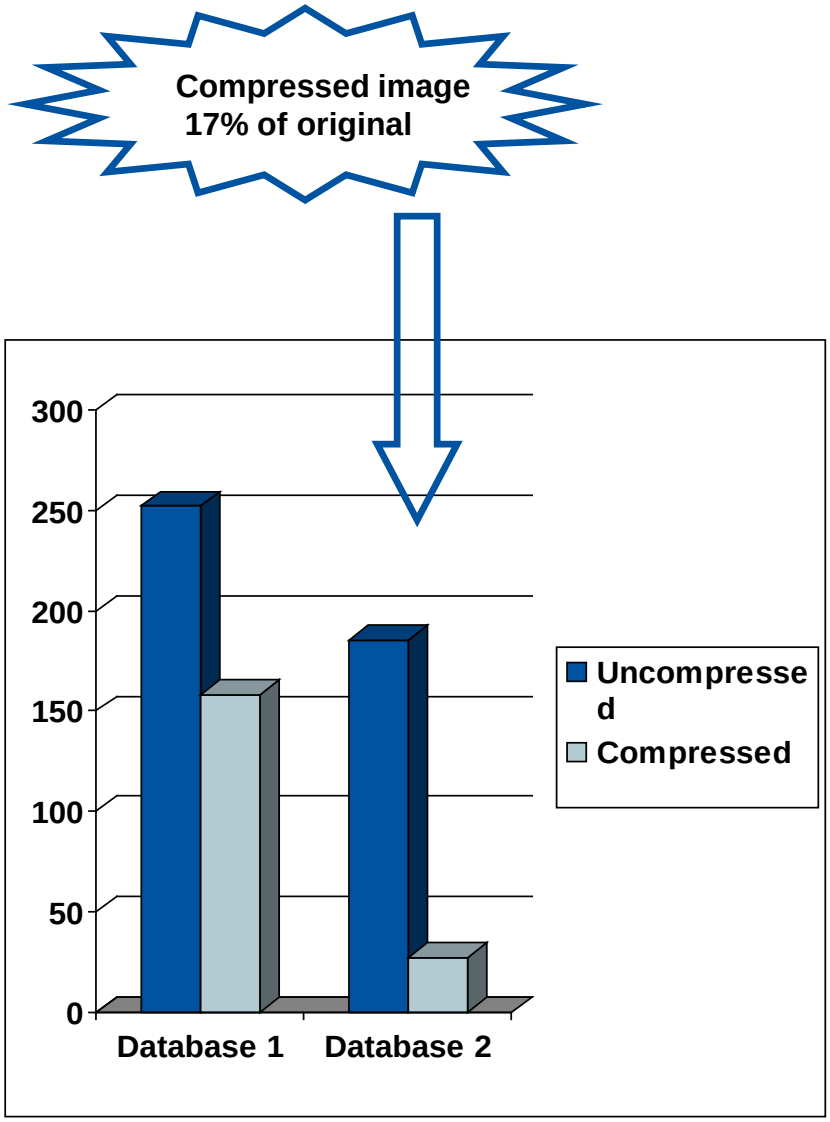
What about backup compression?

- **Compression can be used in 4 different areas**
  1. Table compression
     - Row level compression in the database
     - 10.1 introduced adaptive compression
  2. DB2 Backup compression
     - compressed while backing up
     - Requires additional CPU resources
     - Avoid if backup target is a data deduplication device
  3. TSM Software compression (if you have TSM)
  4. Storage (ie. hardware) level compression
     - Depends on your storage devices
- **Recommendation:**
  - Start with the following:
    - If you have table compression, you may not see a huge benefit with using db2 backup compression as well.
    - If you have storage level compression, then you do may not need to enable to TSM software compression.
- Test the combinations of compression types, to see what is the best fit, it will depend on the resource usage in your environment.

Backup Compression

- DB2 backups can now be automatically compressed
  - Using built-in LZ compression or an (optional) user-supplied compression library
  - Can significantly reduce backup storage costs

- Performance characteristics
  - CPU costs typically increased (due to compression computation)
  - Media I/O time typically decreased (due to decreased image size)
  - Overall backup/restore performance can increase or decrease
    - Depending on whether CPU or media I/O is a bottleneck



11.1

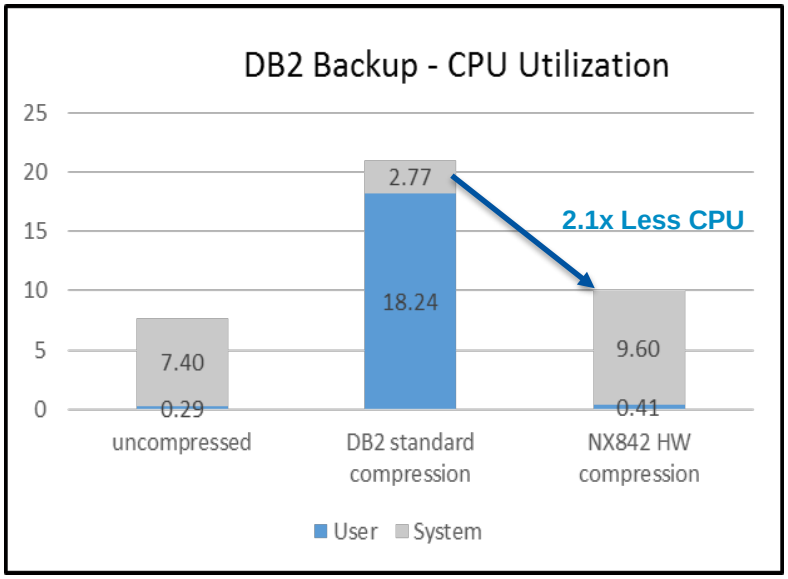
DB2 Support for the NX842 Accelerator

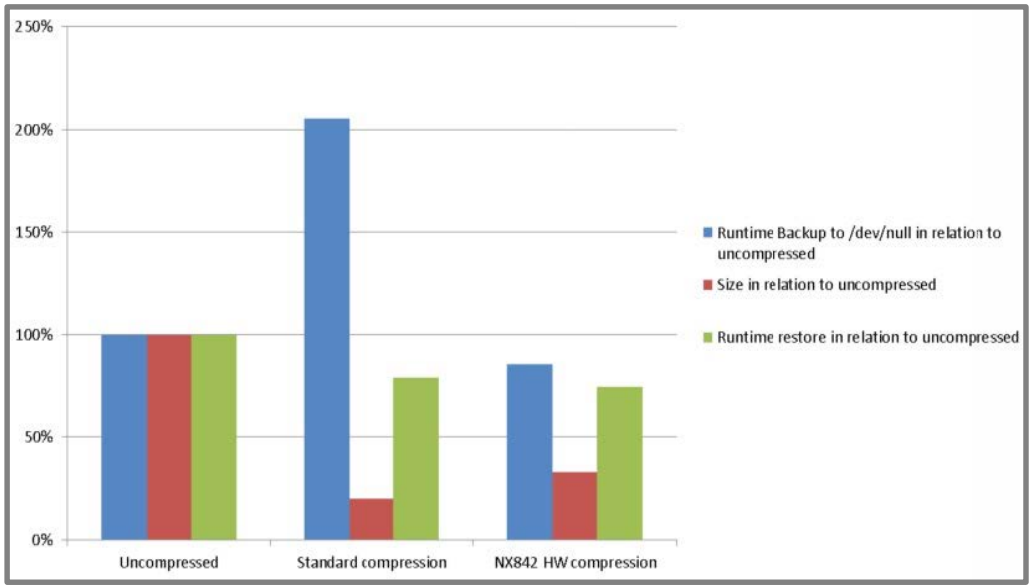
- DB2 backup and log archive compression now support the NX842 hardware accelerator on POWER 7+ and POWER 8 processors
- DB2 BACKUPs require the use of a specific NX842 library
  - `backup database <dbname> compress comprlib libdb2nx842.a`
- Backups can be compressed by default with NX842
  - Registry variable **DB2_BCKP_COMPRESSION** has to be set to **NX842**
  - Use the following backup command format:
  - `backup database <dbname> compress`
- Log archive compression is also supported
  - Update the database configuration parameter **LOGARCHCOMPR1** or **LOGARCHCOMPR2** to **NX842**
  - `update database configuration for <dbname> using LOGARCHCOMPR1 NX842`
  - Note: These two parameters can still take different values

11.1

DB2 Backup Compression Performance Results

- Preliminary results from early system testing
  - I/O bottleneck is usually the limiting factor in backup time
- About 50% DB2 backup size reduction compared to uncompressed
- Factor 2x less CPU consumption compared to DB2 standard compression





Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

The backup history file

- Each database has a backup history file that contains information about database recovery operations
  - BACKUP
  - RESTORE
  - ROLLFORWARD
  - Log file archive
- The history file is pruned periodically to keep its size manageable
  - After a full database backup
  - Using PRUNE HISTORY
- Automatic pruning is managed by two database config parameters
  - NUM_DB_BACKUPS
  - REC_HIS_RETENTN

How to manage the life cycle of backup assets

- The database configuration parameter that controls whether the underlying logs, backups and other associated objects are deleted when the history file is pruned

- AUTO_DEL_REC_OBJ
  - OFF (default, existing behaviour)
  - ON

- When does this automatic deletion occur ?
  - After a successful backup
  - On an explicit PRUNE HISTORY AND DELETE command

- What is deleted ?
  - Any full database backup images which exceed both NUM_DB_BACKUPS and REC_HIS_RETENTN db cfg parameters will be deleted
  - Any associated incremental backup images, load copy images, table space backup images or log files will be deleted

NO TABLESPACE Option for Backup Database

- DB2 History File
  - Contains information about log file archive location, log file chain etc.
  - If you use snapshot backups you want to have current information about log file location to perform point in time recovery (RECOVER command)
  - For RECOVER you need a current version of the history file
  - NO TABLESPACE backup allows you to create a current and consistent backup of the history file in a convenient way
  - If you use COMPRESS option of the BACKUP command, the created backup image is small
- BACKUP with NO TABLESPACE option
  - A NO TABLESPACE backup does not contain tablespaces
  - A no tablespace backup is used to restore the history file by using the HISTORY FILE option with the RESTORE DATABASE command
  - HISTORY FILE keyword is specified to restore only the history file from the backup image

Remote Storage Option for Utilities

- Remote storage is now accessible from:
  - INGEST, LOAD, BACKUP, and RESTORE
  - Accessed through the use of storage access aliases
- Supported Storage
  - IBM® SoftLayer® Object Storage
  - Amazon Simple Storage Service (S3)

Remote Storage Option for Utilities

Catalog storage access first

- CATALOG STORAGE ACCESS ALIAS <alias> VENDOR (SOFTLAYER | S3) SERVER (DEFAULT | <endpoint>) USER <storage-user-ID> PASSWORD <storage-password> [CONTAINER <container-or-bucket>] [OBJECT <object>] [DBGROUP <group-ID> | DBUSER <user-ID>]

Backup to Swift

- db2 backup db testdb to db2remote://Alias//<storage-path>

DB2 Snapshot backups,
AKA - Advanced Copy Services (ACS)
PureScale support was added to
DB2 V 10.5 FP4

Terminology

- **Flashcopy = Hardware snapshot feature name**
- **ACS = Advanced Copy Services, previous name for DB2's integrated flashcopy support**
- **Snapshot = Generic term to reflect logical flashcopy, current feature name for DB2's Flashcopy support**

Current product name	Old Product name
Spectrum Protect	TSM
Spectrum Protect Snapshot	Flash Copy Manager (FCM)
Spectrum Copy Services Manager	TPC for Replication (TPC-R)
Spectrum Copy Data Manager	New (Catalogic)

Integrated Snapshot Copy Backup

- Flashcopy backup/restore just like any other DB2 backup
- 1
  2
  3. Establish the flashcopy pair(s)
  4. Issue DB2 SUSPEND I/O command to tell DB2 to suspend write I/Os
  5. Issue storage commands necessary to do the actual flash copy
  6. Issue DB2 RESUME I/O command to return DB2 to normal

DB2 BACKUP DB sample USE SNAPSHOT

- 1
  2. Issue DB2INIDB command to initialize the database for rollforward recovery
  3. Issue DB2 ROLL FORWARD command

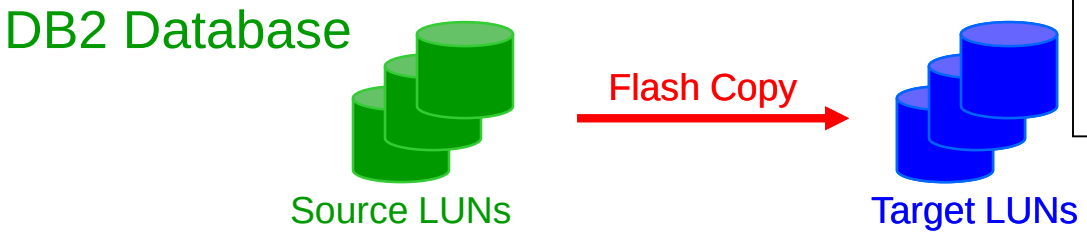
DB2 RESTORE DB sample USE SNAPSHOT

DB2 ROLLFORWARD …

✓ History file record

✓ Simple !

✗ Wide (but not exhaustive) storage support



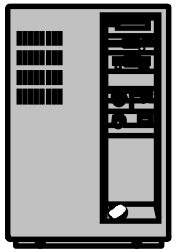
ACS Differences between V 9.7 and V 10.1+

Feature	V 9.7	V 10.1+
AIX	Y(5.3 & 6.1)	Y (6.1 & 7.1)
Linux	Y RHEL 5 & SLES 10 (nSeries/NetApp only)	Y RHEL 5&6 SLES 10&11 (all storage devices)
HPUX IA64	**N**	**Y**
Solaris SPARC	**N**	**Y**
DS8K	Y	Y
XIV	Y (Gen 2)	Y (Gen 2 & 3)
SVC	Y (2.1 – 4.3.1)	Y (4.3.0 – 6.2)
ESS800 (shark)	**Y**	**N**
DS6K	**Y**	**N**
Storwize V7000	**N**	**Y**
Storwize V5000	**N**	**Y (10.5.0.5/10.1.0.4)**
IBM N-Series	**Y**	**Y (10.5.0.5/10.1.0.4)**
NetApp	**Y**	**Y (10.5.0.5/10.1.0.4)**

Mapping of DB2 databases to storage subsystem volumes for using snapshot backups

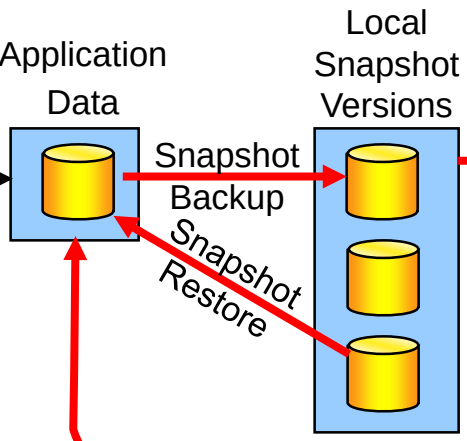
- To use SNAPSHOT backup and restore functions DB2 databases need to be carefully configured on storage system volumes:
  - **Location** of database path and table spaces
    The database path as well as all system, user and temporary table spaces need to be assigned to a set of storage system volumes that do not overlap any other DB2 database or even another database partition of the same database.
  - **Location** of active database logs
    The active and mirror (mirrorlogpath) database logs need to be defined on a separate volume group. This ensures that the recovery logs are not overwritten during a snapshot restore.
  - **Location** of archive logs
    Log files archived to local disk need to be defined on a separate volume group to avoid being overlaid by a snapshot restore

Application System



ORACLE
DB2
SAP
Microsoft SQL Server
Microsoft Exchange Server
Custom Apps
File Systems
VMware

FlashCopy Manager



With Optional TSM Backup Integration

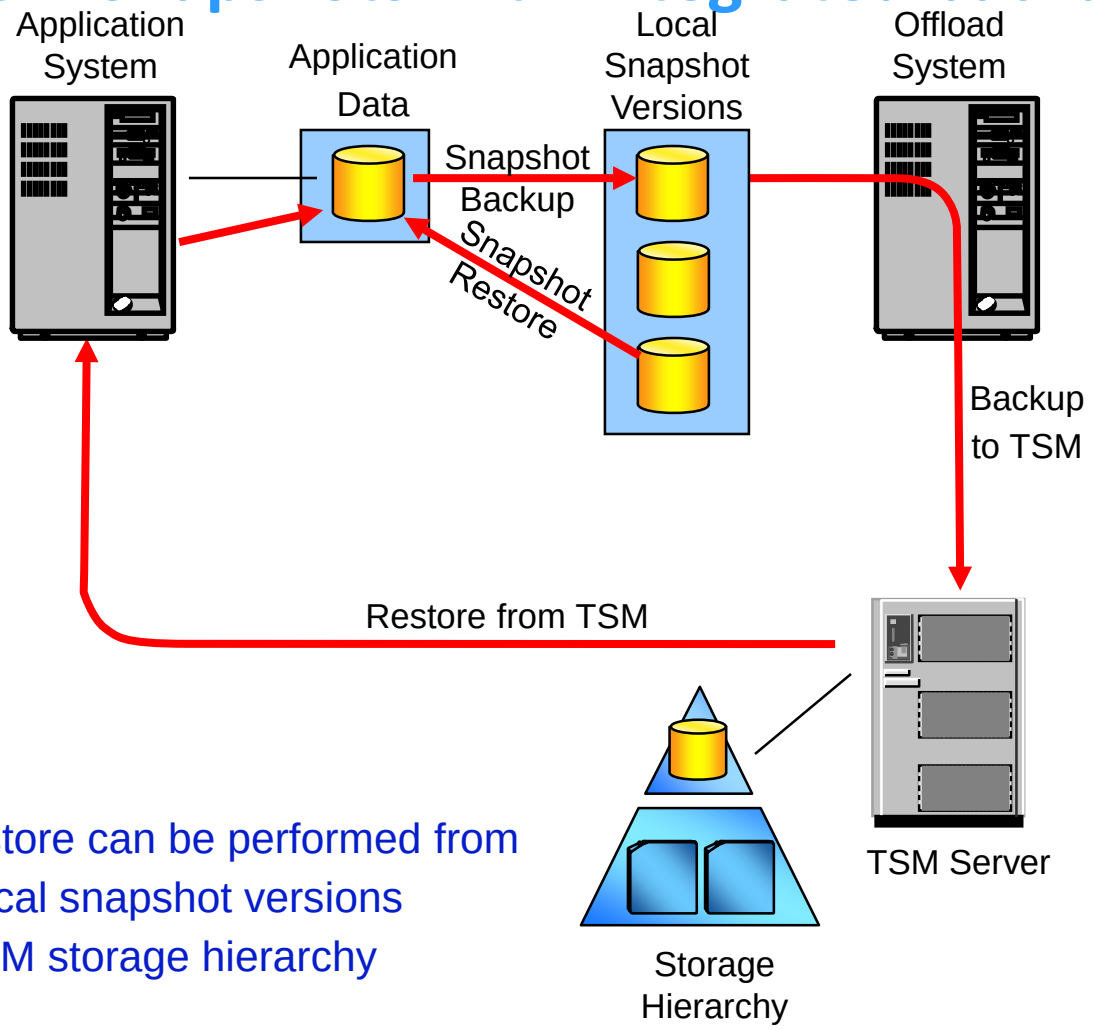
For Various Storage

- ✓SVC
- ✓V7000
- ✓V5000
- ✓V3700
- ✓XIV
- ✓DS8000
- ✓N-Series
- ✓NetApp
- ✓**EMC***
- ✓**HDS***
- ✓Other**

* Via Rocket Adapter
** VSS Integration

- ✓ Online, near instant snapshot backups with minimal performance impact
- ✓ High performance, near instant restore capability
- ✓ Integrated with Storage Hardware snapshots
- ✓ Simplified deployment
- ✓ Database Cloning

FCM snapshots with Integrated backup to TSM



Restore can be performed from
- Local snapshot versions
- TSM storage hierarchy

Snapshot backup to TSM server
- Transfer outboard of application server to minimize impact to application
- Copies on TSM server provide long-term retention and disaster recovery
- Very fast restore from the snapshot

Support for multiple, persistent snapshots
- Persistent snapshots retained locally

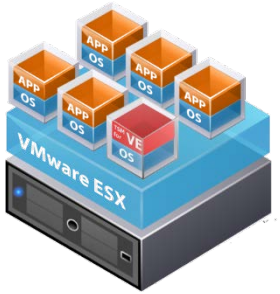
Policy-based management of local, persistent snapshots
- Retention policies may be different for local snapshots and copies on TSM server
- Automatic reuse of local snapshot storage as older snapshot versions expire

Custom Apps

DB2

ORACLE

SAP



- AIX
- Linux x86_64
- Solaris

File sys
db2acs
acsora
backint
VADP

App Agent

FlashCopy Manager

Device agent Interface

Device agent A
Dev agent B
Dev agent C
Dev agent D
Rocket Adapter

Snapshot Providers

XIV
SVC or V7000
DS8000
NetApp
3RD Party Devices EMC. …

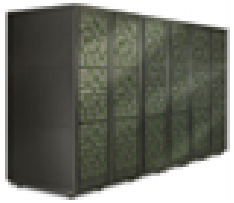
Rocket Device Adapters Extend FCM Value Prop to 3rd Party Devices

Rocket Device Adapters for FCM Summary


EMC
VMAX & DMX



EMC VNX


HDS Storage
Systems

NetApp CDOT

- Q4 2013 release supports EMC VMAX & DMX
  - TimeFinder Snap, VP Snap, and Clone
  - backup, restore, persistent and non-persistent R/W mounts to backup systems
  - Backup types: Snapshot, NoCopy, Differential Copy and Full Copy
  - RAID and thin-provisioned target volumes
  - Dynamically allocated target volumes
- April 2014
  - EMC VNX Snapshot & SnapView Snaps & Clones
    - Block storage
  - EMC VMAX/DMX SRDF integration
- July 2014 - HDS VSP & HUS VM
- October 2014 - NetApp Cluster Mode
- January 2015 - HDS USP & HUS
- March 2015 - HP 3PAR
- April 2015 – HDS G1000
- Installed on the server where FCM is running
  - Simple InstallAnywhere™ installer
- Works like a driver, no user interface
  - Supports AIX, Linux (RedHat & SUSE) and Solaris
- Separately Orderable Product
  - IBM Product ID 5725-P59
  - Front-end capacity based pricing model like FCM
  - Download from the Rocket Customer Portal
  - Support provided by Rocket Software

Rocket Device Adapter Key Requirements

- VMAX & DMX
  - FCM 4.1 or later (4.1.0.1 recommended)
  - EMC Symmetrix Command Line Interface (SYMCLI) v7.5 and later
    - Delivered in EMC Solutions Enabler v7.5
  - VMAX and DMX devices as supported by SYMCLI software
    - Details at **https://elabnavigator.emc.com/**
  - Application and OS platform levels as supported by FCM 4.1
- VNX
  - FCM 4.1.0.2 or later
  - EMC NaviSecCLI software version 7.33
  - VNX devices as supported by NaviSecCLI software
  - Application and OS platform levels as supported by FCM 4.1
- HDS
  - FCM 4.1.0.2 or later
  - HDS Device Manager software version 7.6 or later
  - HDS CCI software version 1-30-03 or later
  - VSP and HUS devices as supported by the HDS Device Manager
    - Details at https://portal.hds.com/
- NetApp CDOT
  - ONTAP 8.1 & later
  - SnapMirror integration
- HP 3PAR
  - FCM 4.1.0.2 or higher.
  - HP 3PAR Remote CLI v3.2 software.

Scripted Interface for Snapshot Copy Backup

- Flashcopy backup/restore just like any other DB2 backup
- Backup
  1. Identify LUN(s) associated with the database
  5. Issue storage commands necessary to do the actual flash copy
  6. Issue DB2 RESUME I/O command to return DB2 to normal

10.5

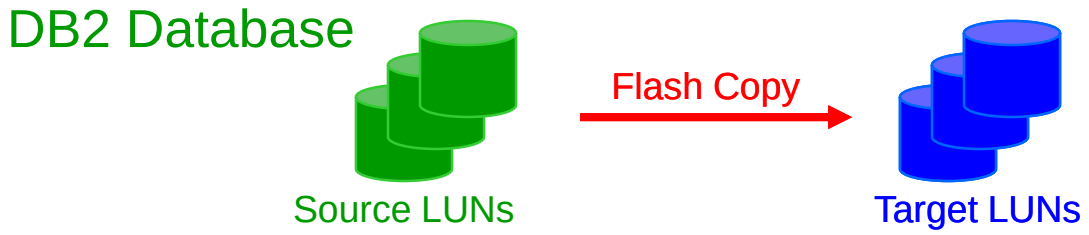
DB2 BACKUP DB sample USE SNAPSHOT SCRIPT

'/myscript.sh'

- Restore

DB2 RESTORE DB sample USE SNAPSHOT SCRIPT
'/myscript.sh' TAKEN AT <timestamp>

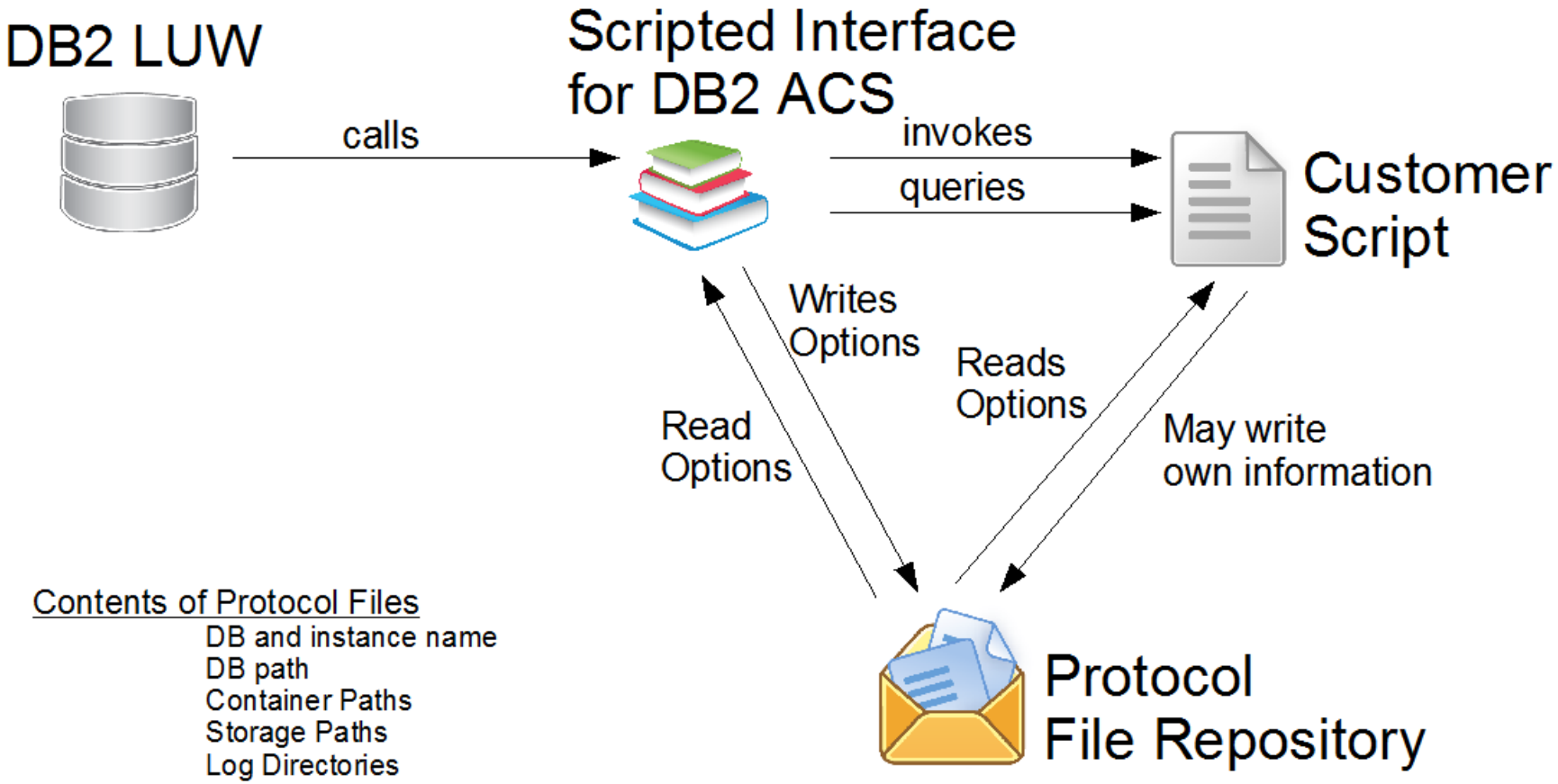
DB2 ROLLFORWARD …



- ✓ History file record
- ✓ Simple to use !
- ✓ Wider storage support enabled

Scripted Interface for Flash Copy Backup

Backup Command using the solution:
BACKUP DATABASE SAMPLE ONLINE
USE SNAPSHOT SCRIPT /script/libacssc.sh OPTIONS '/repository/'



Contents of Protocol Files

- DB and instance name
- DB path
- Container Paths
- Storage Paths
- Log Directories
- Options like include / exclude logs

Scripted interface for snapshot backup

- Available in DB2 10.5, DB2 9.7 FP 9 and DB2 10.1 FP3

- White Papers published
  - Introduction to the Scripted Interface
  - Implementation of the Scripted Interface for DB2 snapshot backup Using Linux LVM
  - Implementation of the Scripted Interface for DB2 snapshot backup Using IBM GPFS
  - Implementation of the Scripted Interface for DB2 snapshot backup using IBM DS4000 midrange storage system

10.5

SET WRITE SUSPEND

New option added in DB2 V 10.5

INCLUDE LOGS (default)

- Specifies that writes to the log files are not allowed when the database is in a write-suspended state. This is the default.

EXCLUDE LOGS

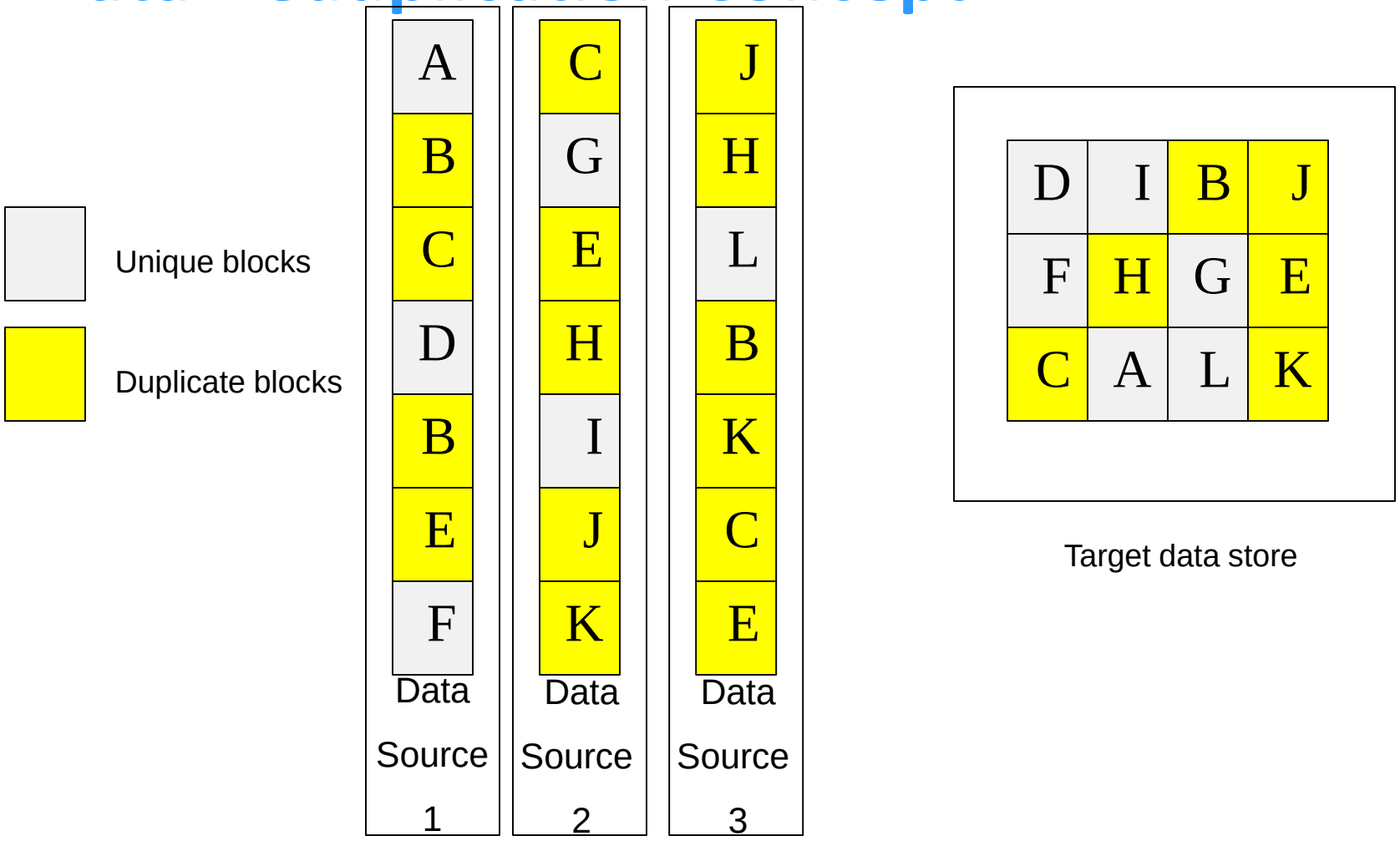
- Specifies that writes to the log files (but not to log file header and mirror log file header files) can occur when the database is in a write-suspended state. This provides a window during which update transactions running against the database can still complete. This can help to reduce the impact on the workload that would normally occur while the database is write suspended. Any copies of the database that are taken while it is write suspended and the EXCLUDE LOGS option is specified must not include log files in the copy.

Note: There are some situations in which logged operations can still be blocked from proceeding. This can happen, for example, if the current active log file is full

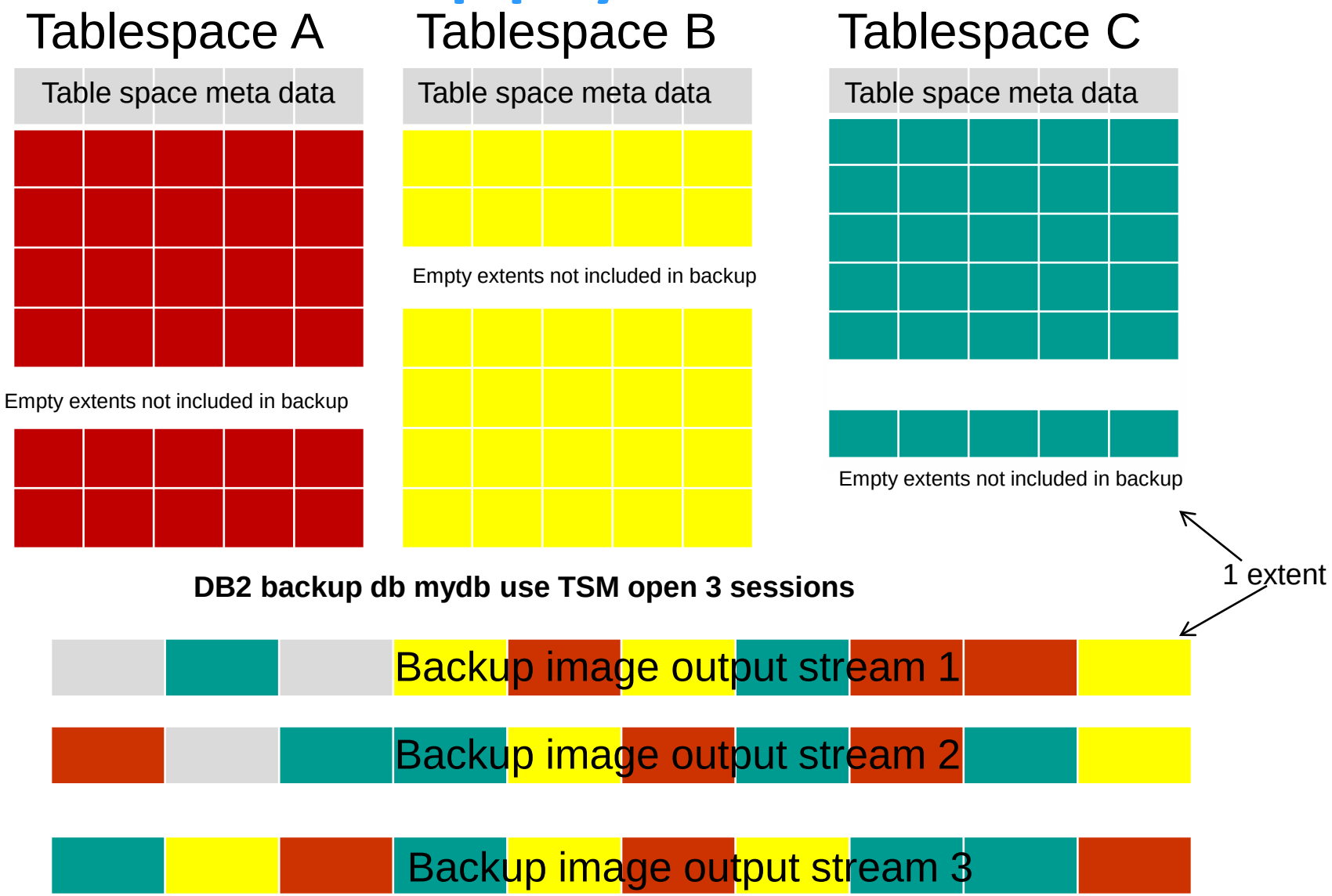
DB2's native support for Data Deduplication

- Native support for all deduplication devices
- Alters the format the backup image to be dedup friendly
- Not recommended for Archived Logs
- Required specialized H/W or S/W
  - IBM ProtecTIER
  - TSM 6.1 or above
  - Data Domain devices
- See White paper: *http://www.ibm.com/developerworks/data/library/techarticle/dm-1302db2deduplication/*

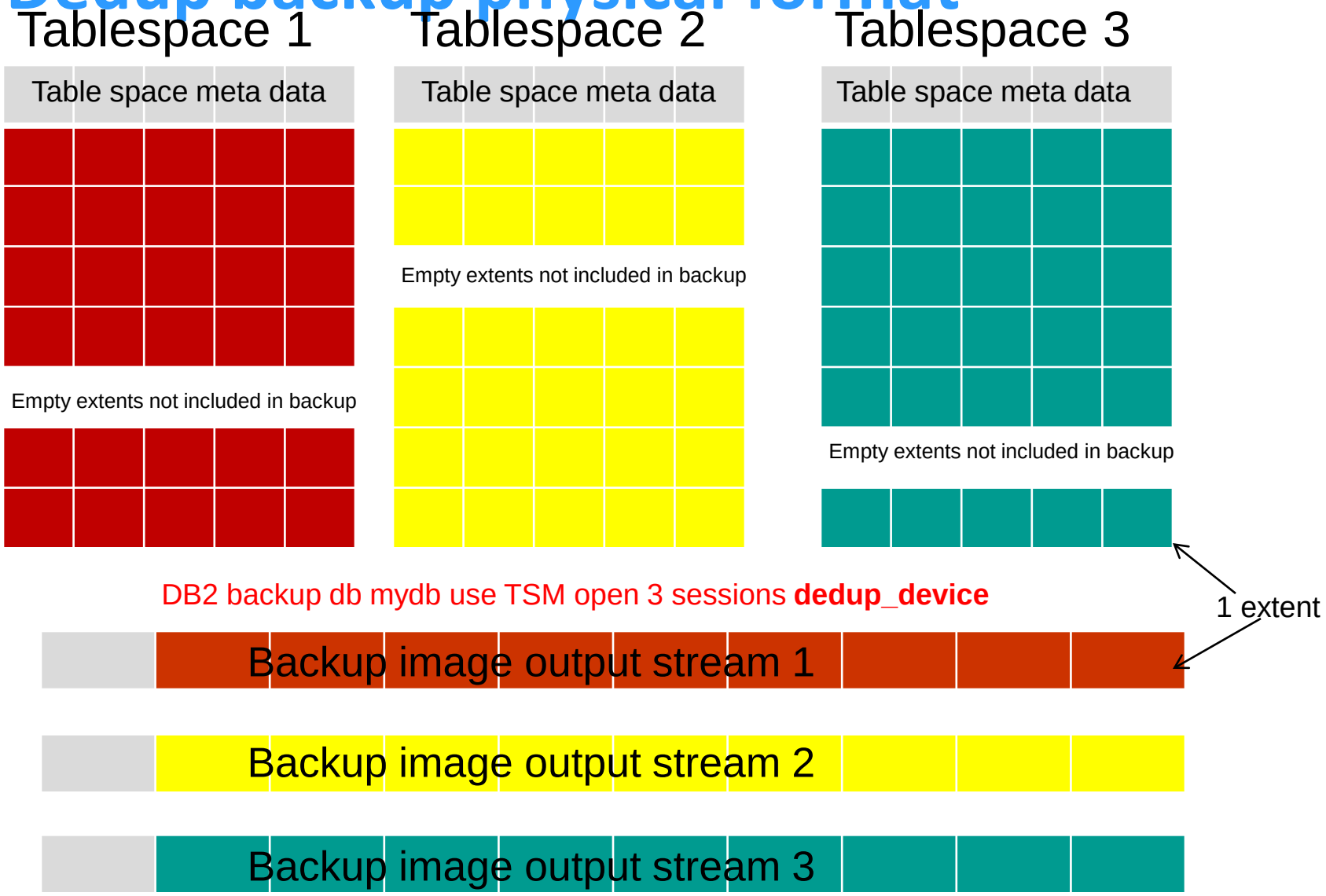
Data Deduplication Concept



Default backup physical format



Dedup backup physical format



TSM Client Deduplication Results

Command:

DB2 backup db PERF use TSM open 8 sessions dedup_device

Settings:

Util_heap_sz = 300000

Autonomic performance settings:

Backup Buffer Size = 16384

Backup # of buffers = 18

Backup Parallelism = 8

Start	End	Elapsed Time	Actual DB Size in bytes	Dedup Size sent to server in bytes	% Savings in bandwidth
10:22:00 PM	1:53:47 AM	3:31	1.36499E+11	1.35729E+11	0.56%
1:53:55 AM	2:22:30 AM	0:28	1.36499E+11	1248387123	99.09%
2:22:26 AM	2:49:39 AM	0:27	1.36499E+11	352043759	99.74%
2:49:44 AM	3:17:12 AM	0:27	1.36499E+11	175945343	99.87%
3:17:18 AM	3:45:32 AM	0:28	1.36499E+11	129529760	99.91%

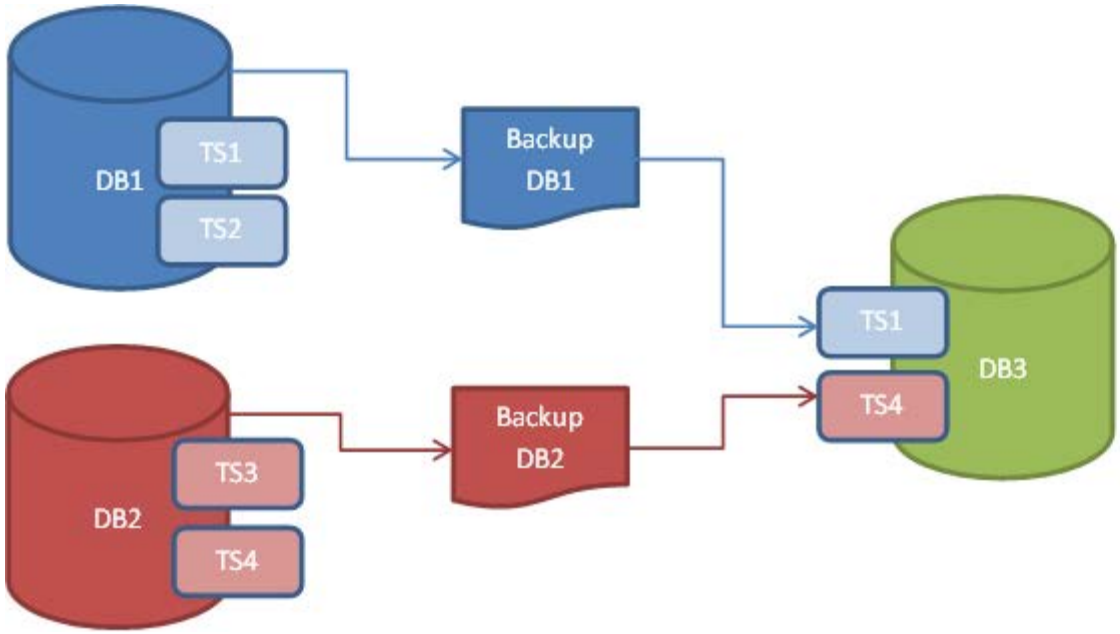
TSM Client Deduplication Results

Command: *DB2 backup db PERF use TSM open 8 sessions* ***dedup_device*** *with 18 buffers buffer* ***16384*** *parallelism 8* ***compress***

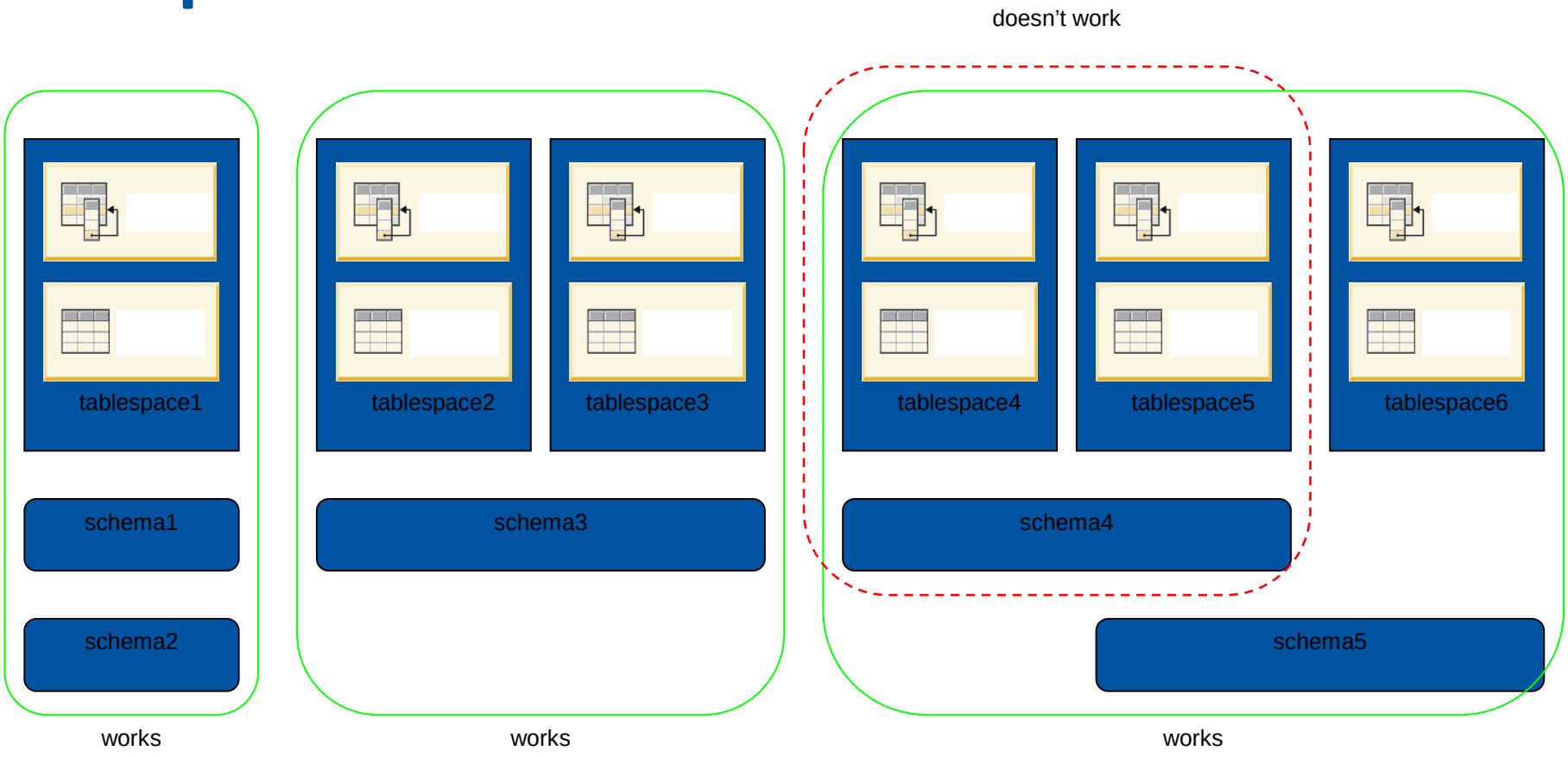
Start	End	Elapsed Time HH:MM:SS	Actual DB Size in bytes	Dedup Size sent to server in bytes	% Savings in bandwidth
5:32:19 PM	8:53:02 PM	3:20:43	103683256320	102876574054	
8:53:02 PM	11:54:50 PM	3:01:48	103683256320	9268429	99.9%
11:54:50 PM	3:13:01 AM	3:18:11	103683256320	2875036	99.9%
3:13:01 AM	6:26:13 AM	3:13:12	103683256320	3566182	99.9%
6:26:13 AM	9:22:23 PM	2:56:10	103683256320	3189526	99.9%

Transportable Schema

Ability to restore a tablespace from a database into a completely separate database



Transport Sets



What will be transported

- Tables / CGTTs / MQTs / 'normal + stats' Views
- Generated columns
  - Expression
  - Identity
  - Row change timestamp
  - Row change token
- UDFs / UDTs + generated functions
- Constraints
  - Check
  - Foreign key
  - Unique / primary
  - Functional dependency
- Indexes
- Triggers
- Sequences
- Procedure – not external routine executable
- Object authorizations / privileges / Security / Access control / Audit
- Table Statistics + profiles / hints (plan lockdown)
- Packages

What will NOT be transported

- Created Global variables
- Aliases
- Jobs
- Index extensions
- Hierarchical tables, typed tables, typed views
- Nicknames
- Structured types
- methods
- Servers
- Wrappers
- Functional mappings & templates
- OLE DB External functions
- sourced Procedures
- XML - sysxmlstrings and sysxmlpaths collisions an issue

DPF and pureScale environments are not be supported

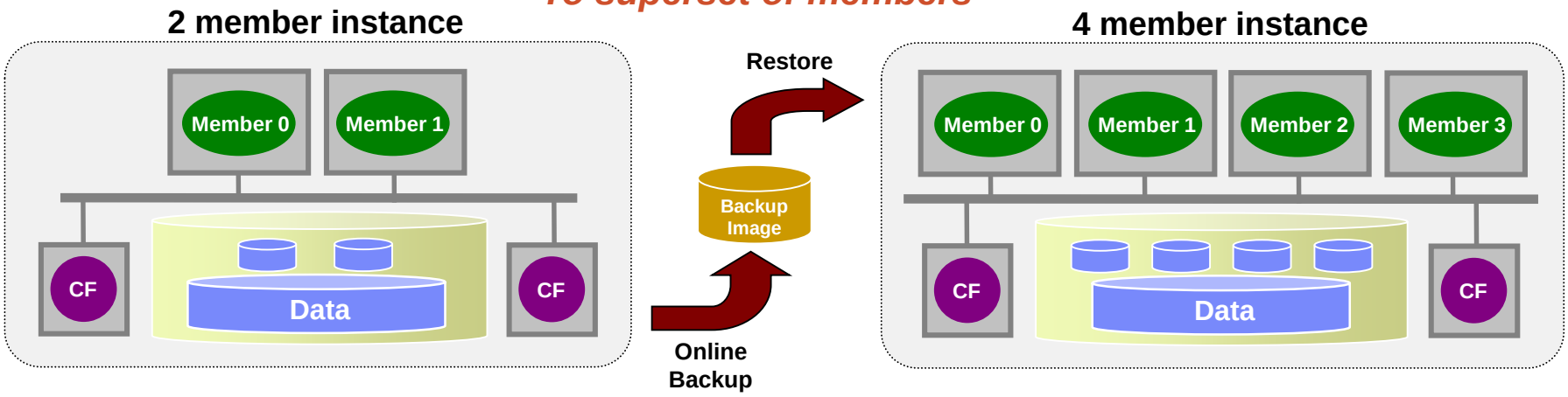
pureScale On-line Management – Backup Restore

10.5

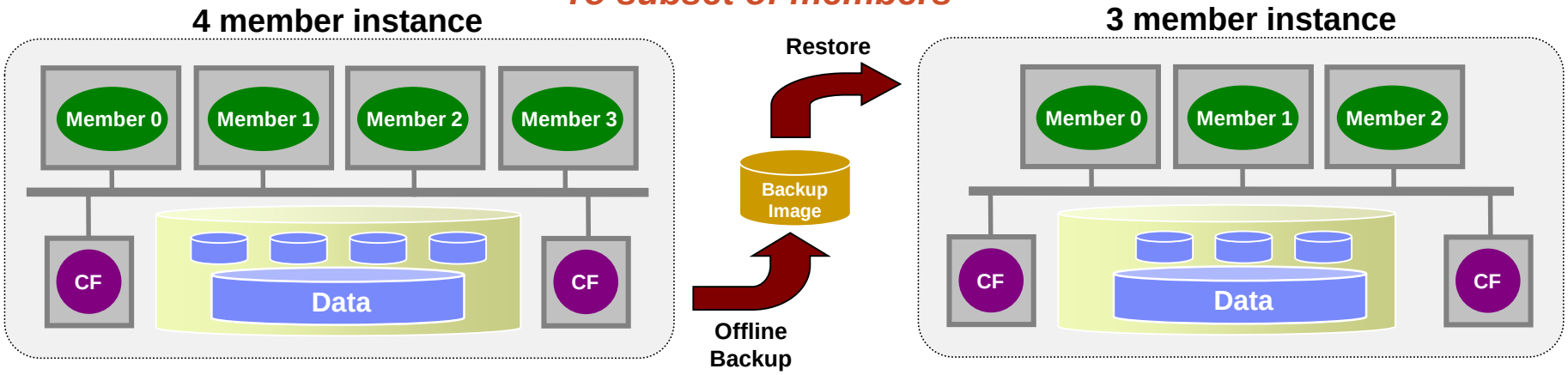
Topology-Changing B/R

- Backup and restore between topologies with differing numbers of members

To superset of members



To subset of members



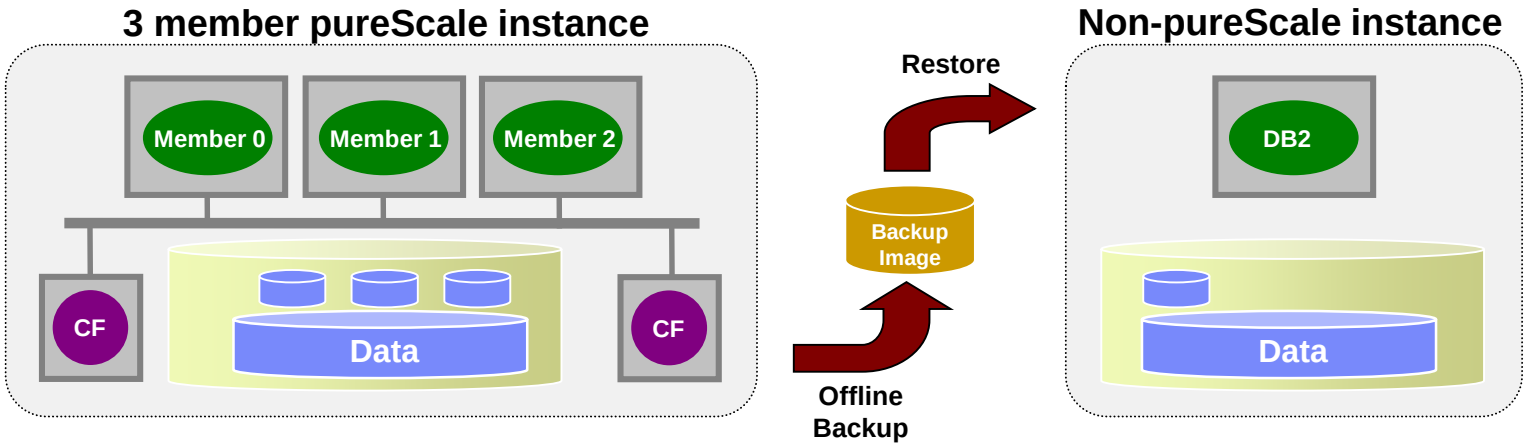
pureScale Management – Backup Restore

10.5

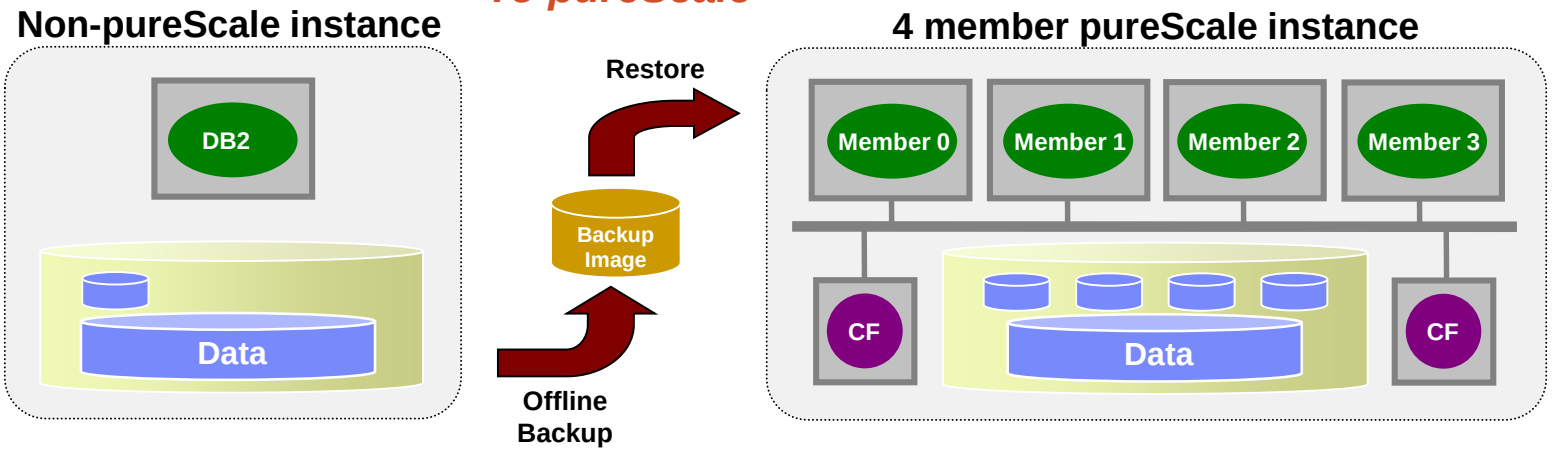
Topology-Changing B/R

- Backup and restore from pureScale to non-pureScale (and vice-versa)

To non-pureScale



To pureScale



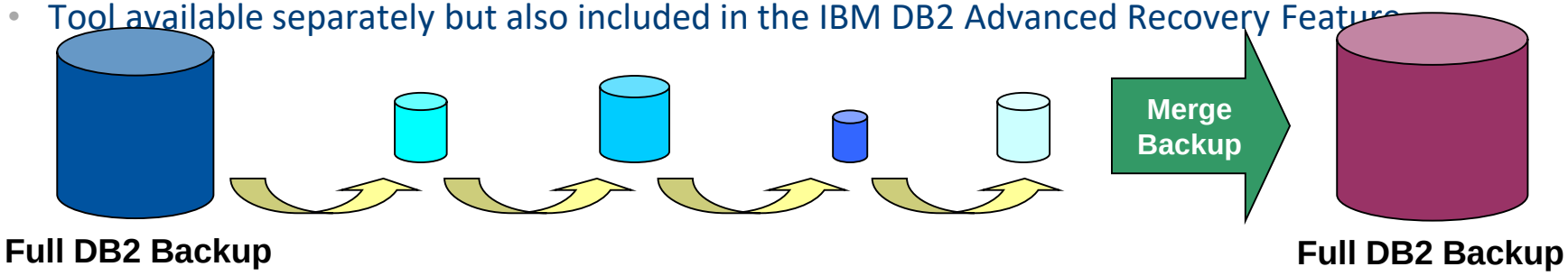
Incremental Backup/Restore and DB2 Merge Backup

10.5

- Incremental backups now supported for pureScale
  - Allows for smaller backup images, as unchanged data not backed up
  - Applicable to database-level or table space-level backups
  - Enabled via TRACKMOD database configuration parameter

- Two types of backups
  - Incremental: Copy of all data that has changed since the most recent, successful, full backup operation (also know as cumulative backup)
  - Delta: Copy of all data that has changed since the last successful backup of any type (full, incremental, or delta) (also known as a differential backup)

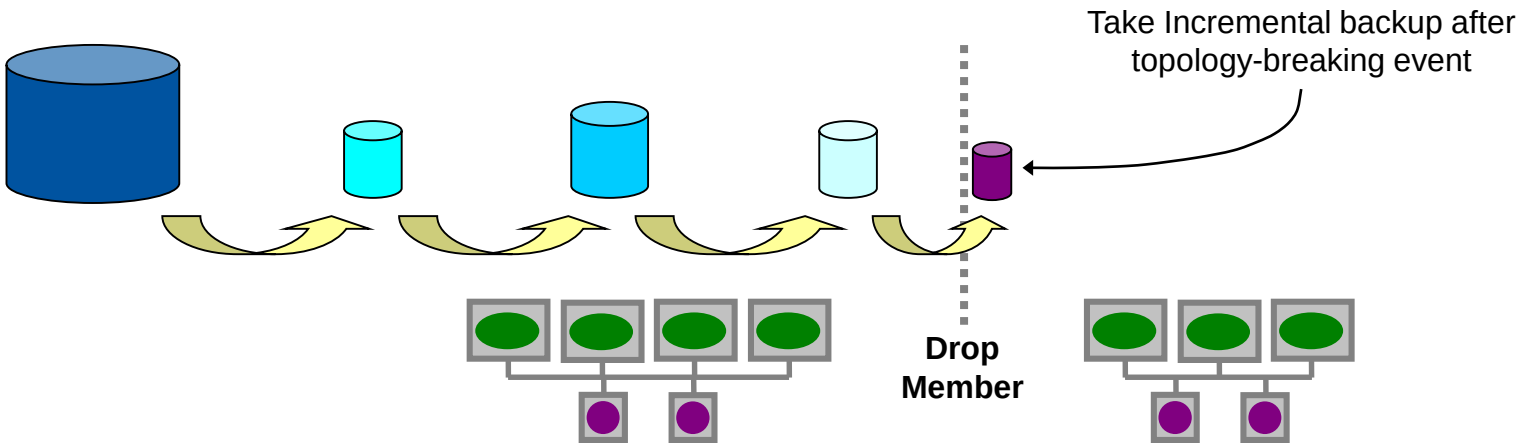
- Support for pureScale in DB2 Merge Backup V2.1 FP1 (to be shipped in parallel with DB2 Cancun Release)
  - The Merge Backup utility combines an older full backup with subsequent incremental and delta backups to create a new full backup image
  - Tool available separately but also included in the IBM DB2 Advanced Recovery Feature



Database Topology Changes with Incremental Backup

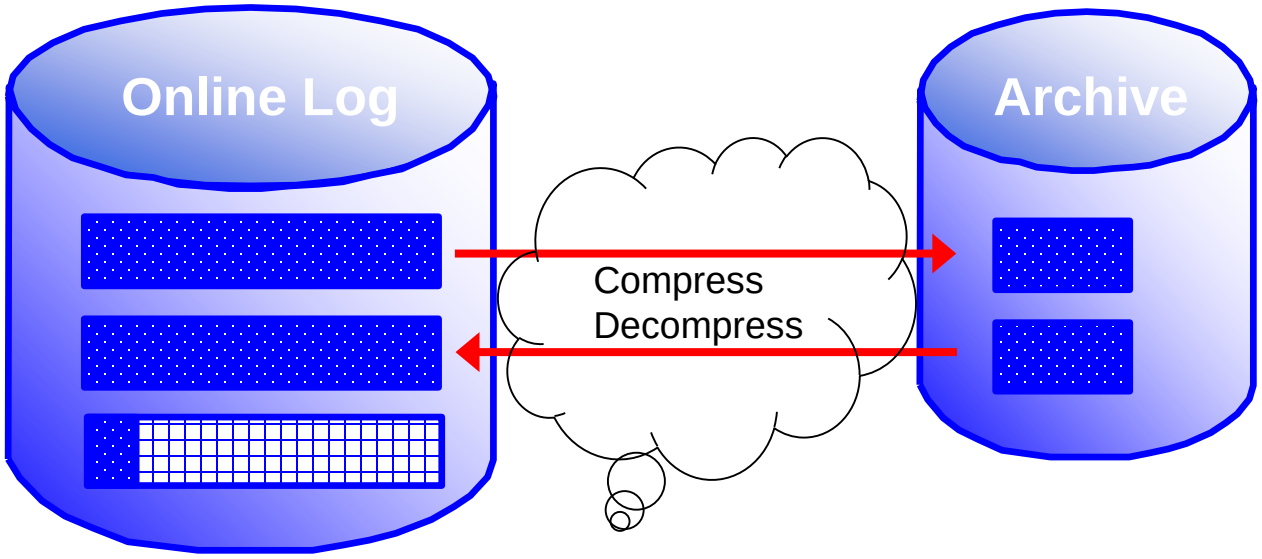
10.5

- Certain operations are considered "topology-breaking" for a database
- Examples include
  - Drop member from cluster
  - Restore database backup to a cluster with a subset of the members
  - Restore non-pureScale database backup into pureScale instance
  - Restore pureScale database backup into non-pureScale instance
- Previously, a full offline database backup was required following these events to provide a new recovery starting point for the database
- Now, an incremental offline database backup can be performed instead
  - Likely to be faster and resulting backup image will be smaller



Archival Log Compression

- Row and index compression helps reduce log record size
- However, each log record also contains a significant amount of housekeeping information (previous log record pointer, transaction ID, etc, …)
- New optional database configuration parameter
- Simply turn it on and DB2 does the compression for you
  - logarchcompr1 database configuration parameter set to ON
- Retrieval automatically determines if decompression is needed



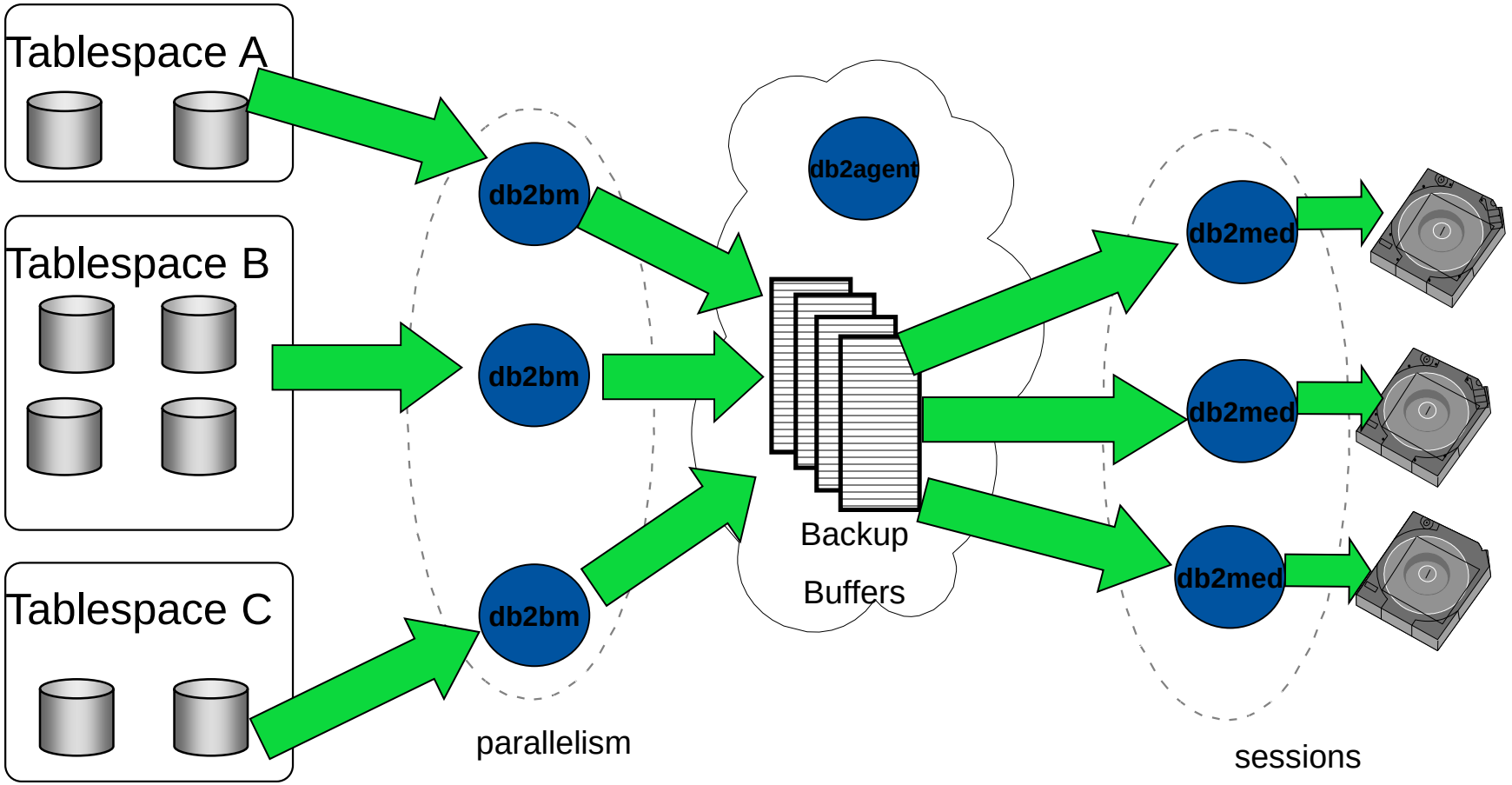
Agenda

- Overview
- Technology Review
  - What's new
  - What could help you become a super star
- Performance Tuning
- Recommendations

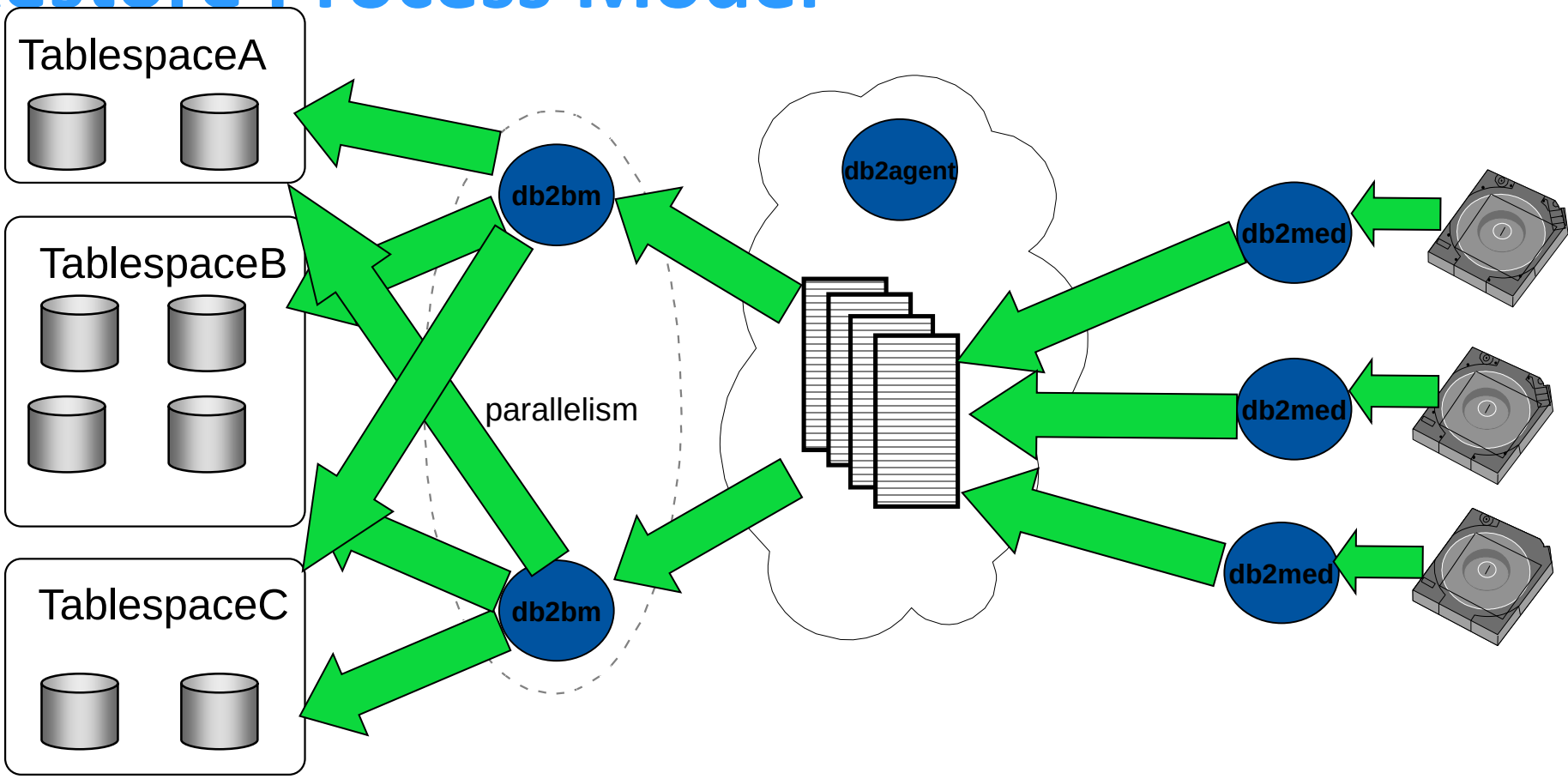
DB2_BAR_STATS

- New db2diag.log entry added at the end of each backup and restore operation
- Contains stats detailing where each BAR thread spent its time
- One entry per db2BM and per db2MED threads
- It was introduced in 9.7 under reg var control, and is enabled by default as of 10.1 FP2 (the reg var is no longer required).

Backup Process Model



Restore Process Model



Sample Output

```
2013-07-19-04.13.57.496158+000 E7374506A2396        LEVEL: Info
PID     : 23658576                 TID  : 4113          PROC : db2sysc 0
INSTANCE: db2pb1                   NODE : 000           DB   : PB1
APPHDL  : 0-7                      APPID: *LOCAL.db2pb1.130717232706
AUTHID  : DB2PB1
EDUID   : 4113                     EDUNAME: db2agent (PB1) 0
FUNCTION: DB2 UDB, database utilities, sqluxLogDataStats, probe:281
MESSAGE : Performance statistics
DATA #1 : String, 1935 bytes

Number of buffers = 30
Buffer size       = 16781312 (4097 4K pages)

BM#     Total         I/O         MsgQ        WaitQ       Buffers    GBytes
---  ---------   ---------   ---------   ---------   ---------   ---------
000  103510.00    34318.08    68661.04        5.49      129892        2029
001  103509.99    12716.83    61922.75    28477.03       85713        1339
002  103509.99    15396.27    71039.63    16605.96      107371        1677
003  103509.99    12022.06    63610.40    27480.43       86771        1355
004  103509.99    14991.83    59660.31    28477.52       83021        1297
005  103509.99    15170.57    59541.68    28421.50       82117        1283
006  103509.99    13501.96    61204.07    28402.28       87714        1370
007  103509.99    15359.61    59459.05    28312.37       82607        1290
008  103509.99    16304.06    58362.13    28476.03       80317        1254
009  103509.99    11367.38    63259.74    28476.81       88328        1380
---  ---------   ---------   ---------   ---------   ---------   ---------
TOT  1035100.00   161148.70   626720.86   243135.47      913851       14278

MC#     Total         I/O         MsgQ        WaitQ       Buffers    GBytes
---  ---------   ---------   ---------   ---------   ---------   ---------
000  103512.33    47805.32     3395.74        0.03      129893        2030
001  75049.26     31252.53      197.54        8.20       85714        1339
002  86913.43     31695.56      329.32        8.20      107372        1678
003  76041.04     29464.31      213.91        8.21       86772        1356
004  75050.68     31783.52      193.83        8.20       83022        1297
005  75099.25     31211.08      199.07        8.21       82118        1283
006  75118.45     41718.12      143.80        8.21       87715        1370
007  75207.78     33519.80      182.74        8.21       82608        1291
008  75046.60     28422.91      208.62        8.20       80318        1255
009  75046.06     30891.54      178.83        8.21       88329        1380
---  ---------   ---------   ---------   ---------   ---------   ---------
TOT  792084.92    337764.72     5243.45       73.90      913861       14282
```

Explanation

BM# - the number we assigned to an individual Buffer Manipulator. BM's READ data from the databases tablespace during a backup and place them into buffers.

MC# - the number assigned to an individual Media Controller. MC's WRITE buffers out to the target location.

Total - The total amount of time spent by the process in seconds.

I/O - The amount of time spent either reading or writing data. For the BM's this represents time reading data from tablespace, and filling the buffer. For MC it's time spent reading from buffer and sending it to the target destination.

MsgQ - This is the amount of time we spend waiting to get a buffer. For BM's it's how long is spent waiting to get an empty buffer for filling. For MC's it's time spent waiting to get a full buffer in order to write out.

Wait Q - Amount of time spent waiting on directives from the agent overseeing the whole backup.

Buffers - The number of Buffers Processed by a particular BM or MC. A BM filled X number of buffers. An MC wrote out X number of buffers.

GBytes - The amount of data handled by a particular BM or MC in Gbytes.

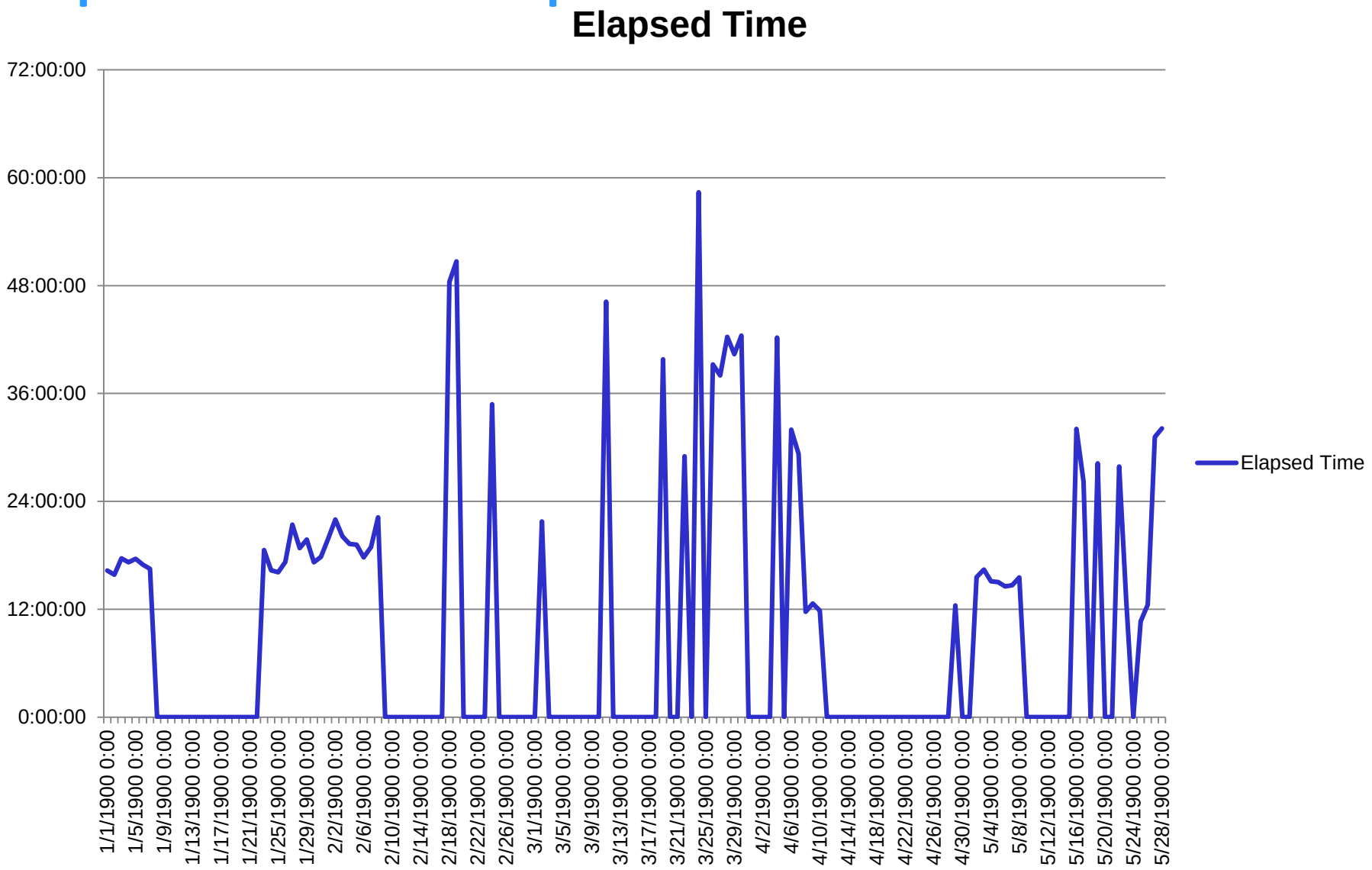
Customer XYZ DB2 Backup Analysis

Why are my backups slow?

Background

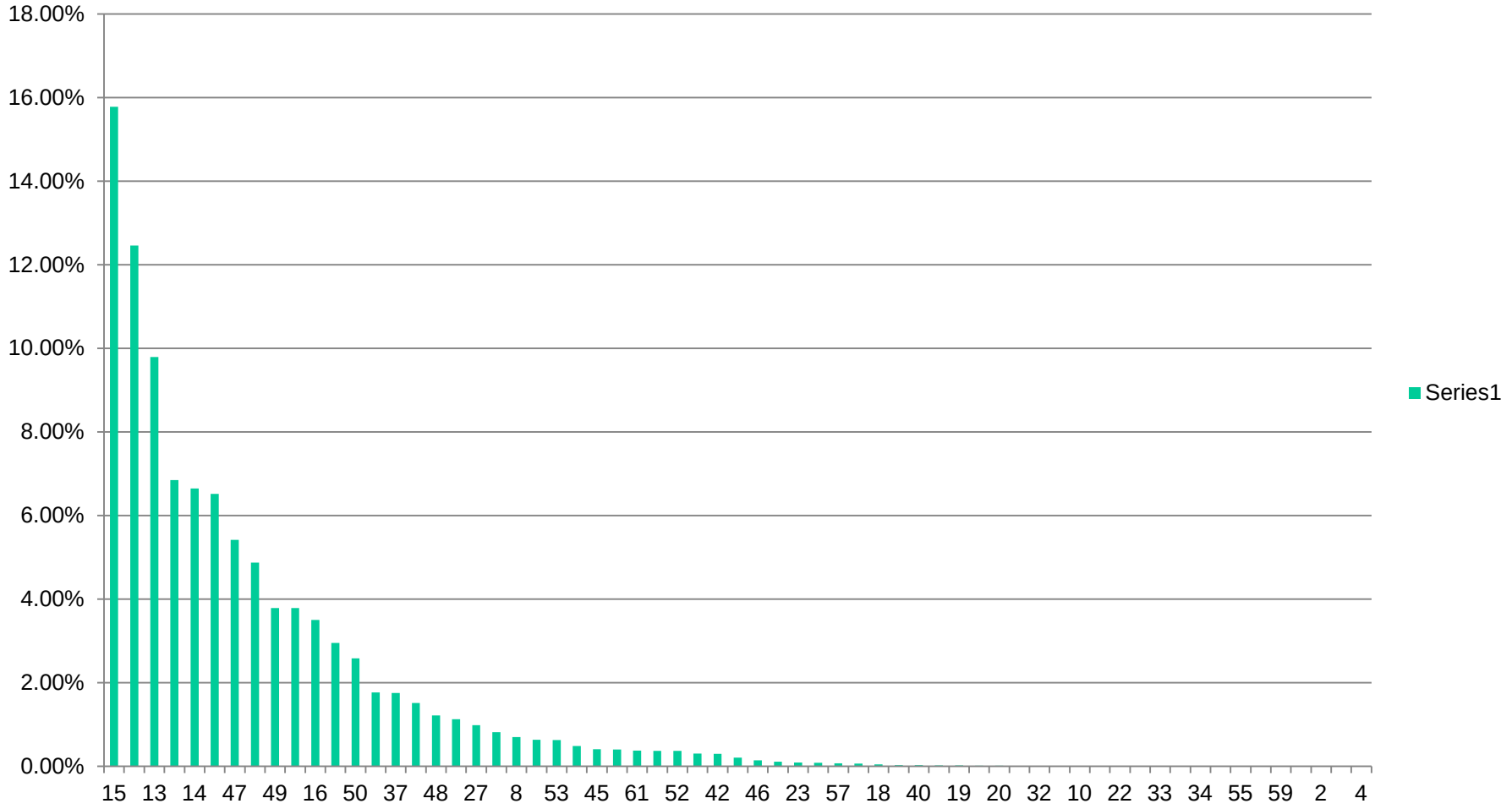
- SAP R3 Customer
- Backing up to TSM which has an IBM ProtecTIER VTL
- Database size is growing by 150 GB per month
- Enabling the data deduplication parameter for DB2 has elongated the overall elapsed time of the backup beyond the 24 hours window
- Database Backup elapsed time has been erratic

Elapsed time for backups



Data distribution

- The data is not evenly distributed across all of the table spaces



Backup Detailed analysis

- Options used: DEDUP_DEVICE, 30 sessions, 60 buffers, buffersize 8192
- Started on July 8, 2013 at 23:29:11
- Completed on July 10, 2013 at 10:10:48
- **Elapsed time: 34:41:37**
- Total Backup size = 14.01 TB
- Largest table space = 2.21 TB
- Time to backup largest single table space was 34:41:11
- Through-put 18MB/second over a single stream

db2_bar_stat analysis – Buffer Manipulators

BM#	Total	I/O	MsgQ	WaitQ	Buffers	GBytes	% Time on I/O	% time waiting for buffers	% time waiting for other threads
---	--------	--------	--------	--------	--------	--------			
0	124885.7	31713.7	92632.03	11.98	70748	2209	25.39%	74.17%	0.01%
1	124885.6	15774	90321.45	18361	55240	1725	12.63%	72.32%	14.70%
2	124885.6	12565.6	78445.55	33513.7	44460	1388	10.06%	62.81%	26.84%
3	124885.6	11674.2	63675.79	49269.3	31386	980	9.35%	50.99%	39.45%
4	124885.6	11759.8	61663.76	51205.5	30191	943	9.42%	49.38%	41.00%
5	124885.6	11025.8	61377.62	52229	29637	925	8.83%	49.15%	41.82%
6	124885.6	10744	57109.16	56814.8	24969	779	8.60%	45.73%	45.49%
7	124885.6	9355.42	52296.05	63039.5	22473	701	7.49%	41.88%	50.48%
8	124885.6	6080.38	48480.21	70168.3	17485	546	4.87%	38.82%	56.19%
9	124885.6	6459.55	47083.99	71185.2	17435	544	5.17%	37.70%	57.00%
10	124885.6	10038.2	42897.55	71810	15646	488	8.04%	34.35%	57.50%
11	124885.6	5868.69	42343.08	76553.8	13615	425	4.70%	33.91%	61.30%
12	124885.6	7038.35	38784.47	78960.5	11895	371	5.64%	31.06%	63.23%
13	124885.6	5033.29	34859.28	84922.3	8208	256	4.03%	27.91%	68.00%
14	124885.6	5270.65	33577.73	85969.6	7792	243	4.22%	26.89%	68.84%
15	124885.6	2954.88	33676.33	88194.4	6970	217	2.37%	26.97%	70.62%
16	124885.6	3053.31	31709.32	90072.4	5902	184	2.44%	25.39%	72.12%
17	124885.6	2017.88	31215.53	91608.1	5181	161	1.62%	25.00%	73.35%
18	124885.6	1756.7	28654	94437.6	4501	140	1.41%	22.94%	75.62%
19	124885.6	2193.6	27125.59	95535.2	3733	116	1.76%	21.72%	76.50%
20	124885.6	2523.08	25863.92	96471.6	3232	100	2.02%	20.71%	77.25%
21	124885.6	1369.08	26293.93	97195	3310	103	1.10%	21.05%	77.83%
22	124885.6	1264.64	26314.23	97280.2	3218	100	1.01%	21.07%	77.90%
23	124885.6	1458.65	26120.73	97279.9	3171	98	1.17%	20.92%	77.90%
24	124885.6	2279.59	25301.51	97279.8	2939	91	1.83%	20.26%	77.90%
25	124885.6	2438.22	25131.44	97291.1	2960	92	1.95%	20.12%	77.90%
26	124885.6	1220.39	26358.59	97280.3	3161	98	0.98%	21.11%	77.90%
27	124885.6	2121.5	25528.18	97209.1	3211	100	1.70%	20.44%	77.84%
28	124885.6	1044.03	26533.28	97281.3	3216	100	0.84%	21.25%	77.90%
29	124885.6	1984.92	25592.93	97280.9	3203	99	1.59%	20.49%	77.90%
---	--------	--------	--------	--------	--------	--------			
TOT	3746569	190082	1256967	2295712	459088	14322	5.07%	33.55%	61.28%

db2_bar_stat analysis – Media Controllers

MC#	Total	I/O	MsgQ	WaitQ	Buffers	GBytes	% Time on I/O	% time waiting for buffers	% time waiting for other threads
0	124888.9	76825.5	842.67	0.03	70749	2210	61.52%	0.67%	0.00%
1	106543.5	37585.2	543.94	9.87	55241	1726	35.28%	0.51%	0.03%
2	91390.44	35590.4	289.39	9.87	44461	1389	38.94%	0.32%	0.03%
3	75634.08	29064.1	153.2	9.87	31387	980	38.43%	0.20%	0.03%
4	73700.61	36517.5	72.37	9.87	30192	943	49.55%	0.10%	0.03%
5	72675.22	26042.5	157.67	9.87	29638	926	35.83%	0.22%	0.04%
6	68093.13	21956	152.8	9.87	24970	780	32.24%	0.22%	0.04%
7	61863.59	24598.9	72.07	9.87	22474	702	39.76%	0.12%	0.04%
8	54734.77	13687.5	88.64	9.87	17486	546	25.01%	0.16%	0.07%
9	53717.47	12710.5	88.35	9.87	17436	544	23.66%	0.16%	0.08%
10	53095.89	10927	100.67	9.87	15647	488	20.58%	0.19%	0.09%
11	48348.48	10626.5	70.37	9.87	13616	425	21.98%	0.15%	0.09%
12	45944.12	11687.7	55.88	9.87	11896	371	25.44%	0.12%	0.08%
13	39980.77	5671.74	52.12	9.87	8209	256	14.19%	0.13%	0.17%
14	38933.72	6738.3	50.5	9.87	7793	243	17.31%	0.13%	0.15%
15	36711.24	5515.8	44.08	9.87	6971	217	15.02%	0.12%	0.18%
16	34830.15	4265.31	41.27	9.87	5903	184	12.25%	0.12%	0.23%
17	33296.2	3433.6	35.27	9.87	5182	161	10.31%	0.11%	0.29%
18	30469.33	2692.67	35.07	9.87	4502	140	8.84%	0.12%	0.37%
19	29370.44	2867.86	27.1	9.87	3734	116	9.76%	0.09%	0.34%
20	28437.13	2091.62	27.27	9.87	3233	101	7.36%	0.10%	0.47%
21	27707.99	2397.45	25.55	9.87	3311	103	8.65%	0.09%	0.41%
22	27629.07	2221.45	26.64	9.87	3219	100	8.04%	0.10%	0.44%
23	27626.2	1401.62	28.24	9.87	3172	99	5.07%	0.10%	0.70%
24	27634.34	2110.49	26.77	9.87	2940	91	7.64%	0.10%	0.47%
25	27614.46	1695.52	27.51	9.87	2961	92	6.14%	0.10%	0.58%
26	27626.69	1778.02	27.7	9.87	3162	98	6.44%	0.10%	0.56%
27	27693.92	1572.03	28.13	9.87	3212	100	5.68%	0.10%	0.63%
28	27621.05	1790.85	25.49	9.87	3217	100	6.48%	0.09%	0.55%
29	27626.77	2153.48	26.16	9.87	3204	100	7.79%	0.09%	0.46%
TOT	1451440	398217	3242.89	286.26	459118	14331	20.17%	0.16%	0.26%

Recommendations

- Analyze TSM Server / PT environment to determine why TSM is responding slowly – Max of 18.5 MB/second over a single session
- Implement DB2 row level compression on all large tables
- Customer should consider implementing an archive and/or purge process to reduce the overall size of the database
- Distribute the data more evenly within the database across the table spaces
- Decrease the number of sessions to 10, this will reduce the amount of time the db2bms are spending waiting
- Decrease the size of the buffer to so that TSM can respond quicker, use a buffer size of 4097
- Ensure there are sufficient buffers, with 10 sessions you should be using 30 buffers
- Consider increasing the NUM_IOSERVERS db cfg parameter to 10*14 = 140 (this may affect runtime performance)

Changes Implemented Immediately

- Decreased the number of sessions from 30 to 10
  - Most sessions were idle
- Decreased the size of the buffer to 8192 to 4097
  - Allow quicker response from TSM
- Descreased number of buffers from 60 to 30
  - Ensure there are sufficient buffers for all threads
- Changed NUM_IOSERVERS db cfg parameter to 10*14 = 140
  - Ensure there are enough prefetchers for the backup utility

Post Changes Backup Analysis

- Options used: DEDUP_DEVICE, 10 sessions, 30 buffers, buffersize 4097
- Started on July 17, 2013 at 23:27:07
- Completed on July 19, 2013 at 04:13:57
- **Elapsed time: 28:46:50**
- Total Backup size = 13.95 TB
- Largest table space = 1.98 TB
- Time to backup largest single table space was 28:45:03
- Through-put 17MB/second over a single stream

Post changes analysis of DB2_BAR_STATS output

Number of buffers = 30									
Buffer size	1678	1312 (4097 4K pages)							
BM#	Total	I/O	MsgQ	WaitQ	Buffers	GBytes	% Time on I/O	% time waiting for buffers	% time waiting for other threads
0	103510	34318.08	68661.04	5.49	129892	2029	33.15%	66.33%	0.01%
1	103509.99	12716.83	61922.75	28477.03	85713	1339	12.29%	59.82%	27.51%
2	103509.99	15396.27	71039.63	16605.96	107371	1677	14.87%	68.63%	16.04%
3	103509.99	12022.06	63610.4	27480.43	86771	1355	11.61%	61.45%	26.55%
4	103509.99	14991.83	59660.31	28477.52	83021	1297	14.48%	57.64%	27.51%
5	103509.99	15170.57	59541.68	28421.5	82117	1283	14.66%	57.52%	27.46%
6	103509.99	13501.96	61204.07	28402.28	87714	1370	13.04%	59.13%	27.44%
7	103509.99	15359.61	59459.05	28312.37	82607	1290	14.84%	57.44%	27.35%
8	103509.99	16304.06	58362.13	28476.03	80317	1254	15.75%	56.38%	27.51%
9	103509.99	11367.38	63259.74	28476.81	88328	1380	10.98%	61.11%	27.51%
TOT	1035099.9	161148.7	626720.8	243135.42	913851	14274	15.57%	60.55%	23.49%

MC#	Total	I/O	MsgQ	WaitQ	Buffers	GBytes	% time on I/O	% time waiting for buffers	% time waiting for agent
0	103512.33	47805.32	3395.74	0.03	129893	2030	46.18%	3.28%	0.00%
1	75049.26	31252.53	197.54	8.2	85714	1339	41.64%	0.26%	0.01%
2	86913.43	31695.56	329.32	8.2	107372	1678	36.47%	0.38%	0.01%
3	76041.04	29464.31	213.91	8.21	86772	1356	38.75%	0.28%	0.01%
4	75050.68	31783.52	193.83	8.2	83022	1297	42.35%	0.26%	0.01%
5	75099.25	31211.08	199.07	8.21	82118	1283	41.56%	0.27%	0.01%
6	75118.45	41718.12	143.8	8.21	87715	1370	55.54%	0.19%	0.01%
7	75207.78	33519.8	182.74	8.21	82608	1291	44.57%	0.24%	0.01%
8	75046.6	28422.91	208.62	8.2	80318	1255	37.87%	0.28%	0.01%
9	75046.06	30891.54	178.83	8.21	88329	1380	41.16%	0.24%	0.01%
TOT	792084.88	337764.7	5243.4	73.88	913861	14279	42.61%	0.57%	0.01%

Observations

- **Improved overall elapsed time by 6+ hours**
- Observed increase % of wait time for TSM per db2MED thread, however each thread must now process more data. **Overall totals decreased by 15%**
- Observed increased time waiting for buffers, this is expected due to observation #2. **Overall totals decreased by 50%**
- Observed increase % of time waiting for I/O. **Overall totals decreased by 15%.**

March 2015 Latest Update

- Distribute the data more evenly across the table spaces using the admin_table_move procedure
- Implement adaptive compression to reduce overall DB size on most of the largest tables
  - DB size 10.2 TB

- RESULTS:
  - Average backup time 15:14:48
  - Compared to 34:41:37 when we started

March 2015 DB2_BAR_STATS output

Number of buffers = 30
Buffer size = 16781312 (4097 4kB pages)

BM#	Total	I/O	MsgQ	WaitQ	Buffers	GBytes	% Time on I/O	% time waiting for buffers	% time waiting for other threads
0.00	57173.28	2517.47	54184.42	251.75	67941	1061	4.40%	94.77%	0.44%
1.00	57173.27	2264.38	54488.35	198.62	69391	1084	3.96%	95.30%	0.35%
2.00	57173.27	2988.10	53717.40	255.06	66566	1039	5.23%	93.96%	0.45%
3.00	57173.27	3180.01	53526.62	255.13	66131	1033	5.56%	93.62%	0.45%
4.00	57173.27	2770.56	53986.24	201.48	66689	1041	4.85%	94.43%	0.35%
5.00	57173.27	6955.30	49765.47	254.92	61101	954	12.17%	87.04%	0.45%
6.00	57173.27	5982.53	50977.12	8.85	63771	996	10.46%	89.16%	0.02%
7.00	57173.27	6640.54	50155.10	176.72	61725	964	11.61%	87.72%	0.31%
8.00	57173.27	5125.88	51583.75	254.97	64750	1011	8.97%	90.22%	0.45%
9.00	57173.27	4012.76	52682.19	255.07	69715	1089	7.02%	92.14%	0.45%
TOT	571732.76	42437.58	525066.71	2112.61	657780	10275	7.42%	91.84%	0.37%

MC#	Total	I/O	MsgQ	WaitQ	Buffers	GBytes			
0.00	56932.20	26766.03	178.94	0.03	67941	1061	47.01%	0.31%	0.00%
1.00	56987.95	26433.48	178.16	10.91	69392	1084	46.38%	0.31%	0.02%
2.00	56934.07	26269.79	177.43	10.94	66567	1040	46.14%	0.31%	0.02%
3.00	56933.48	22438.73	211.08	10.89	66132	1033	39.41%	0.37%	0.02%
4.00	56991.76	20771.30	232.07	10.94	66690	1042	36.45%	0.41%	0.02%
5.00	56931.78	19318.64	240.55	10.94	61102	954	33.93%	0.42%	0.02%
6.00	57189.49	20111.70	236.53	10.90	63773	996	35.17%	0.41%	0.02%
7.00	57012.08	20191.93	255.09	10.90	61726	964	35.42%	0.45%	0.02%
8.00	56933.05	19882.94	293.40	10.94	64751	1011	34.92%	0.52%	0.02%
9.00	56930.52	26750.77	255.70	10.94	69716	1089	46.99%	0.45%	0.02%
TOT	569776.43	228935.36	2258.99	98.39	657790	10280	40.18%	0.40%	0.02%

Feb 2016 Latest Update

- Distribute the data more evenly across the table spaces using the admin_table_move procedure
- Implement adaptive compression to reduce overall DB size on most of the largest tables
  - DB size now 8.8 TB
- Modified storage arrays for Protectier.

- RESULTS:
  - Average backup time 10:00:47
  - Compared to 34:41:37 when we started

Latest DB2_BAR_STATS output

BM#	Total	I/O	MsgQ	WaitQ	Buffers	MBytes	% Time on I/O	% time waiting for buffers	% time waiting for other threads
0	37901.74	3799.84	33484.94	411.45	61203	978976	10.03%	88.35%	1.09%
1	37901.72	3043.69	33487.78	1170.83	59229	947504	8.03%	88.35%	3.09%
2	37901.72	3429.41	33203.28	1070.14	59186	946845	9.05%	87.60%	2.82%
3	37901.72	4172.74	32386.12	1155.59	55588	889173	11.01%	85.45%	3.05%
4	37901.72	3666.92	32870.93	1170.95	57297	916521	9.67%	86.73%	3.09%
5	37901.72	5953.33	30613.38	1166.43	49865	797548	15.71%	80.77%	3.08%
6	37901.72	8814.05	28925.98	6.74	46140	737863	23.26%	76.32%	0.02%
7	37901.72	9333.21	28380.74	37.54	44845	717032	24.62%	74.88%	0.10%
8	37901.72	3160.15	33371.79	1170.93	59147	946193	8.34%	88.05%	3.09%
9	37901.72	4123.42	32585.37	998.82	57583	921167	10.88%	85.97%	2.64%
TOT	379017.26	49496.8	319310.36	8359.46	550083	8798826	13.06%	84.25%	2.21%

MC#	Total	I/O	MsgQ	WaitQ	Buffers	MBytes	% time on I/O	% time waiting for buffers	% time waiting for agent
0	37502.95	12033.87	144.55	0	61203	979471	32.09%	0.39%	0.00%
1	36742.79	12997.36	130.11	8.49	59230	947895	35.37%	0.35%	0.02%
2	36856.42	12210.2	136.67	8.49	59187	947207	33.13%	0.37%	0.02%
3	36754.93	12063.34	136.53	8.49	55589	889625	32.82%	0.37%	0.02%
4	36740.75	12810.88	128.1	8.49	57298	916975	34.87%	0.35%	0.02%
5	36747.11	10476.66	149.39	8.49	49866	798034	28.51%	0.41%	0.02%
6	37904.77	9703.62	409.98	8.49	46142	738420	25.60%	1.08%	0.02%
7	37875.67	8781.63	380.08	8.49	44846	717695	23.19%	1.00%	0.02%
8	36740.47	12057.28	172.42	8.49	59148	946583	32.82%	0.47%	0.02%
9	36912.79	11040.93	187.35	8.49	57584	921552	29.91%	0.51%	0.02%
TOT	370778.69	114175.79	1975.24	76.46	550093	8803460	30.83%	0.53%	0.02%

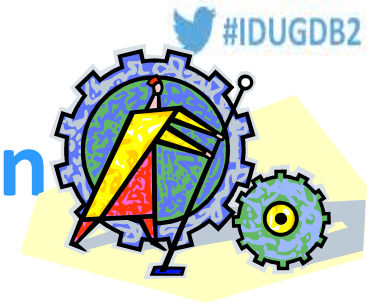
Further work items

- Distribute the data more evenly across the table spaces
- Examine layout of protecTIER meta data layout, currently residing on old storage – Gen 2 XIV storage
- Work with PT, TSM and Network administrators to determine why lan-free through-put is low
- Implement data purging process to reduce overall DB size
- Implement row / adaptive compression to reduce overall DB size

Agenda

- Overview
- Technology Review
  - What's new
  - What could help you become a super star
- Performance Tuning
- Recommendations

Best Practices – Your database design



- Leverage intelligent database design practices!
  - All referential tables are in the same table space, or set of table spaces.
  - All staging tables (for load) are in a separate table space
  - Active vs. In-active data – separated by table space
    - Range partitioning – further identify active vs. inactive by designing ranges
  - Local vs. Global Indexes – if you have global indexes (ie. Prior to v9.7), then keep those in a table space that you back up with it's matching data table space. Rebuilding your indexes can be time consuming!
- Goal: you want to be set up to restore only what you need.

Best Practices: Backup recommendations

- Backup Images:
  - Online database backups
  - Online table space backups
  - Incremental backups
  - Compression is good *
- Use the Single System View option. Let DB2 manage the backups and timestamps for you.
- Always use lan-free option to write to tape drives directly via SAN
  - DO NOT archive directly to tape!
  - configure FAILARCHPATH in case primary-log-destination becomes unavailable
- Include the log records in each backup image – it's easier that way!
- Operating System files backed up twice per week

Best Practices: Backup recommendations cont'd

- Example:
  - Full online table space backup of hot* and warm table spaces twice a week, include logs
  - Full online database backup of catalog partition daily, logs included
  - Full online database backup of database partitions on quarterly or monthly basis
  - Full table space backup after adding a new table space

Best Practices - Logging

- Use archive logging – not circular logs for your production environment
- Goal: ease of maintenance
  - Include logs in your backup image
  - Use TSM to handle your archiving for you
- Small transaction logs:
  - Reduce risk since logs contain fewer transactions
  - Increases the number of logs required, since large transactions may span multiple log files, plus more overhead in system resources.
- Large transaction logs:
  - Increase risk since logs contain more transactions
  - Benefit: reduces the amount of system resources used, as less log switching takes place
- Recommendation: Start with 50 primary logs, of about 50MB each.

Best Practices - Recovery recommendations

- For table space recovery, use the restore command with the REBUILD option
  - Use the REBUILD option (v9.5) to get the subset of the system you need up and running immediately
- Use the log manager, not user exits
- Use db2adutl to retrieve log files from tape to disk – keep ahead of the log files you need .

Backup and Recovery

- **If a DB2 online backup is not feasible due to conflicts with other utilities then consider SNAPSHOTS**

OR

- **IBM Tivoli Storage Manager for Advanced Copy Services For DB2**
  - http://www.ibm.com/developerworks/tivoli/library/t-acsdb2_1/index.html
- **When using TSM** always **use Lan-Free option**
- **Review SAP 20 TB BI Benchmark Results**
  - Full backup in < 7 hours
  - http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101012

- **Best Practices for backup and recovery in ISAS**
  - http://www.ibm.com/developerworks/data/bestpractices/isasrecovery/index.html

More Information

- DB2 LUW Best Practices: https://www.ibm.com/developerworks/data/bestpractices
- Best Practices: Building a Recovery Strategy for an IBM Smart Analytics System Database http://www.ibm.com/developerworks/data/bestpractices/isasrecovery/index.html
- Best Practices: Multi-Temperature Data Management http://www.ibm.com/developerworks/data/bestpractices/multitemperature/index.html
- Article: DB2 instance recovery for IBM Smart Analytics System http://www.ibm.com/developerworks/data/library/techarticle/dm-1010db2instancerecovery/index.html?ca=drs-


Thank
You

Dale McInnis
IBM Canada Ltd.
dmcinnis@ca.ibm.com

Please fill out your session evaluation before leaving!

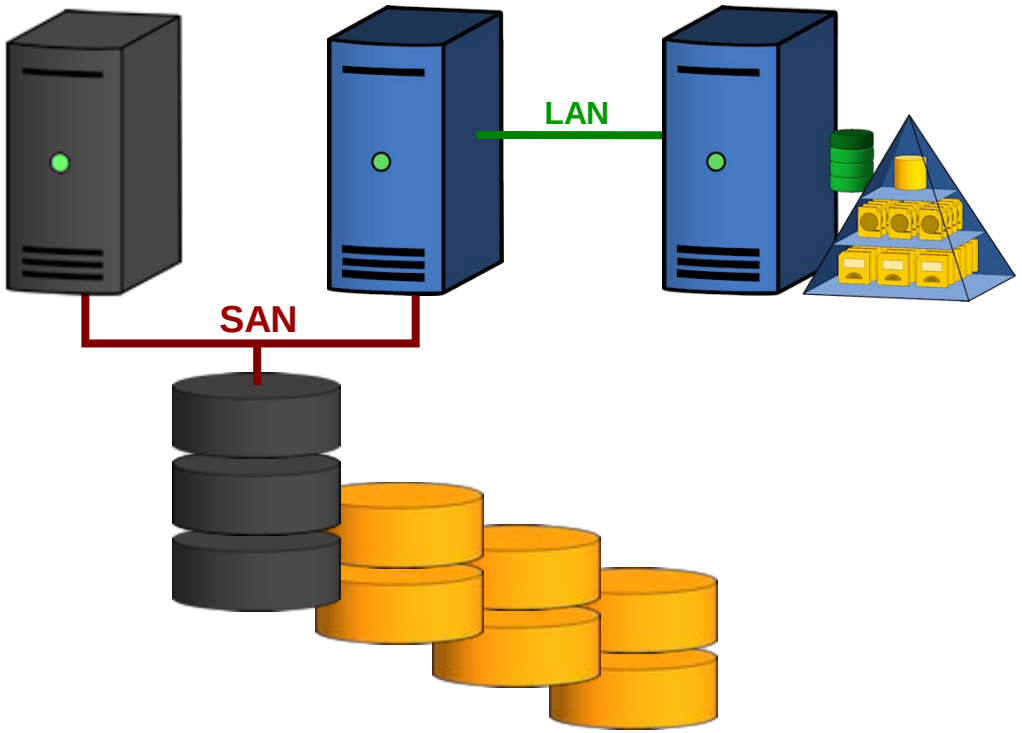
Session: XD17
DB2 Backup and Recovery Best Practices

Photo by Steve from Austin, TX, USA

BACKUP SLIDES

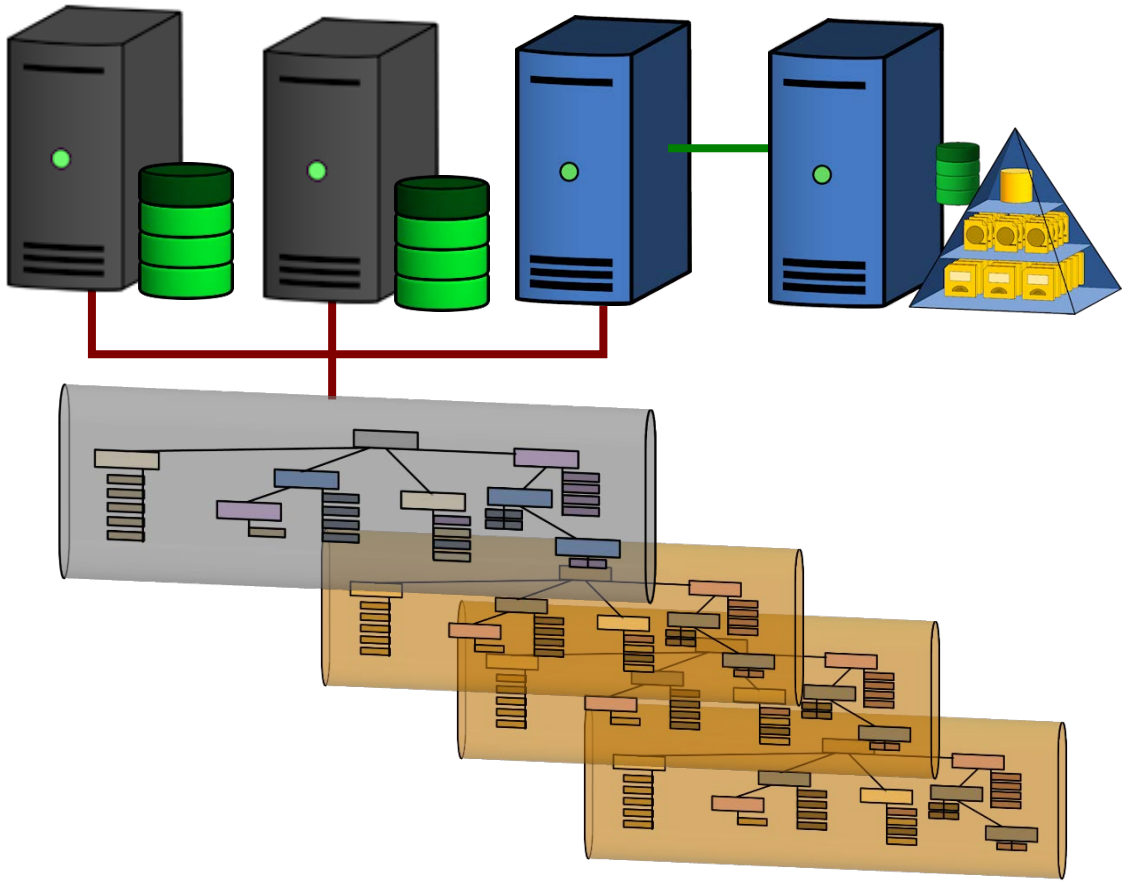
IBM Spectrum Protect Snapshot

- Allows you to protect application and file system disks using snapshot capabilities found in disk hardware
- Hardware snapshots provide near-instant backup and restore of large amounts of data
- Can operate standalone or in combination with Spectrum Protect
- Coordinates with the OS and application to ensure the snapshot creates an application consistent backup
- Allows for offloaded backups, moving the data transfer workload off the application machine to a proxy backup host



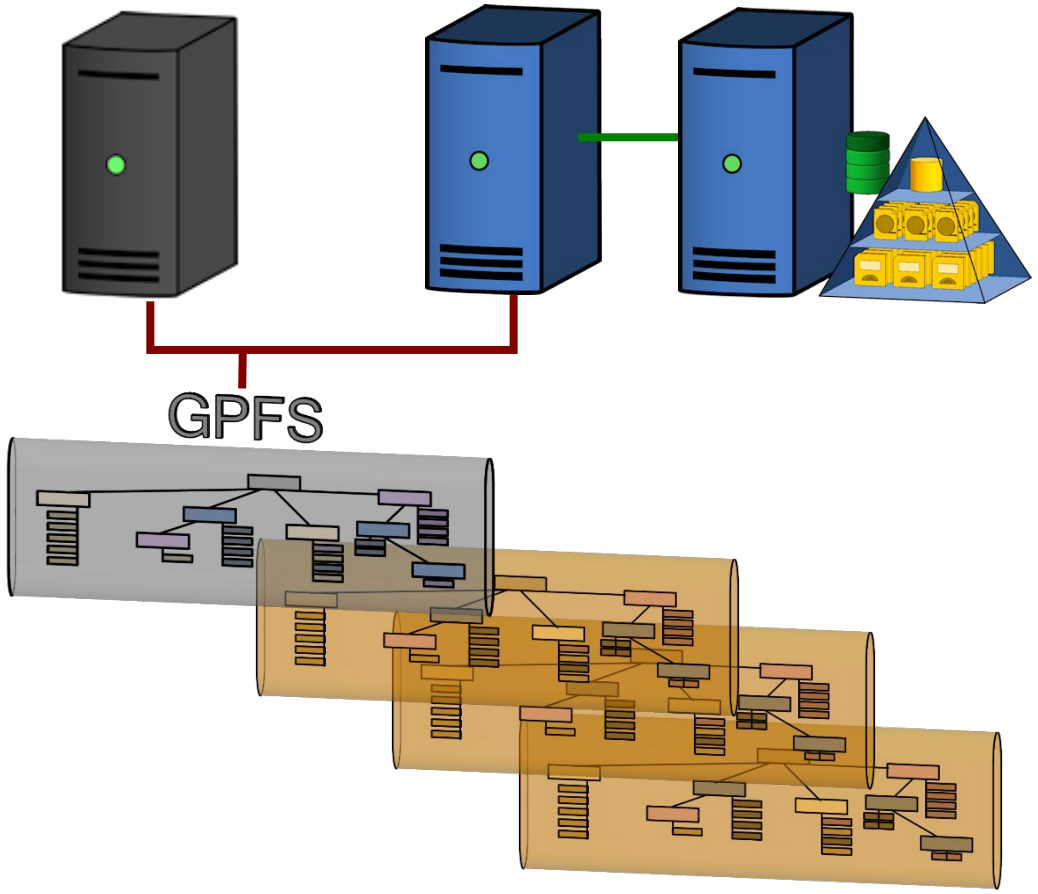
Spectrum Protect Snapshot (FlashCopy Manager)

- Spectrum Protect Snapshot currently supports exploiting Spectrum Scale (GPFS) Fileset snapshots for DB2 PureScale clusters.
- Includes support for sending a copy of the data to Spectrum Protect for longer term storage.
- Includes support for performing snapshots/backups even if PureScale node has experienced a HA failover.



Spectrum Protect Snapshot (FlashCopy Manager)

- Spectrum Protect Snapshot supports for Spectrum Scale snapshot of independent filesets and custom applications (on AIX and Linux).
- Custom applications means that the application owner must provide any freeze and thaw scripts, otherwise it is just a file system backup.



Spectrum Protect and Spectrum Protect Snapshot for DB2

- DB2 has built-in interface using Spectrum Protect API on *nix and Windows.
  - Multiplexing, multithreading support
  - Awareness of Spectrum Protect deduplication (placement of data within data streams to not hide duplicates.
  - DB2 compression of streams supported
  - DB/Tablespace/Log backups (Full, Incremental, Delta).
  - Support for Partitioned Databases
- Spectrum Protect Snapshot provides explicit support for DB2 backups on *nix platforms (standard disk type requirements apply).
- DB2 can also directly call snapshot backups (no integration with Spectrum Protect for same backup, much less offloaded to Spectrum Protect).

Why do people reorg tables

What	Details	Potential benefits
Reclaim space	Compact data onto fewer data pages Return extents to the tablespace	Faster table scans Can use the space in other tables
Recluster	Reorder rows into the same sequence as an index	Faster index scans where data fetch needed (aka “ISCAN-FETCH”)
Remove pointer & overflow records	Convert pointer overflow pairs into a single normal record	Faster row access
(Re-)Compress	(Re-) builds compression dictionary with latest data, and (re-) compresses the data	Reduced storage consumption Reduced bufferpool consumption
Materiali Schema Changes	Physically materialize column changes – eg. Physically adds new column’s default value to existing rows (vs dynamically constructing it)	Returns table to full table availability if limited due to ALTER TABLE Reduced CPU consumption
Conversion to Large RIDS	Conversion from 4-byte to 6-byte records IDs (applies to tables created prior to 9.7)	Larger tables Greater than 255 rows per page

Should I run Reorg whenever I can

- Absolutely NOT!
  - The benefits may not be meaningful for your workloads
    - Eg. Faster table scans are not important if your workload does not do them
  - There can be better ways to achieve similar benefits
    - Several technologies have been delivered to either streamline or avoid reorgs completely
    - E.g. read-ahead prefetching in v 10.1
  - Some types of reorgs can consume significant resources and affect availability
- Please remember
  - Choose the right flavor of reorg, or no reorg at all, depending on your situation
  - Avoid issuing reorgs as a routine, business-as-usual strategy

Table Reorg concepts: Classic Reorg

1. (optional) compression dictionary build
   - If row table has COMPRESS attribute, and KEEPDICTIONARY option is not used
   - Uses optimized scan of table
2. (optional) Sort
   - For reclustering reorgs
     - Sort used by default
3. Shadow Table Build
   - Rows are compressed with existing or new dictionary
   - Rows ordered according to clustering index, if present
     - Order comes from sort by default
     - Indexscan if INDEXSCAN option used (not recommended)
   - Created in a temp tablespace if specified
4. Replace (T1 with shadow)

   -OR-

   Copy (shadow from Temp Tablespace)
5. Index rebuild

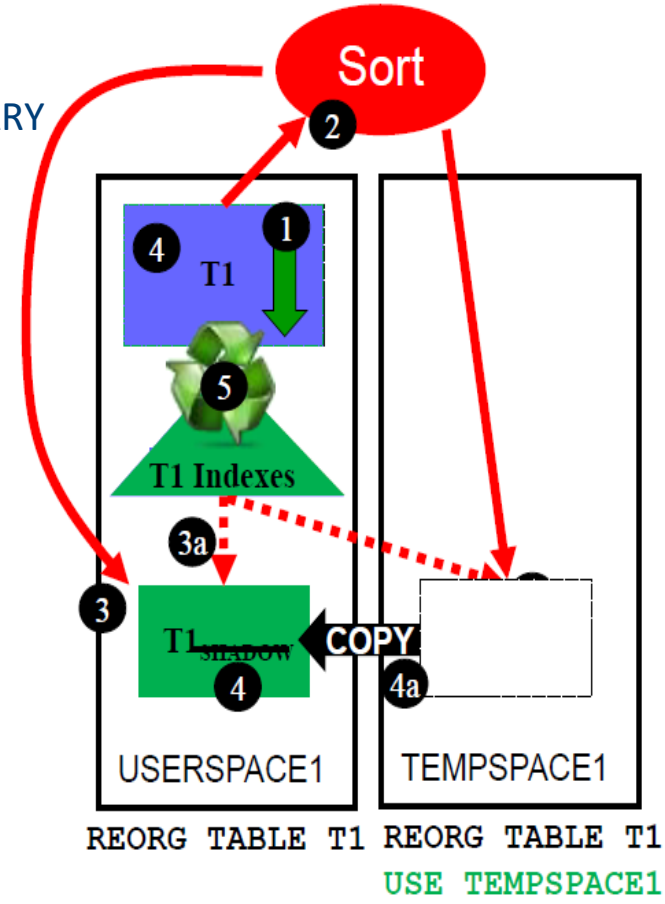


Table Reorg Concepts : Availability/Speed Comparison

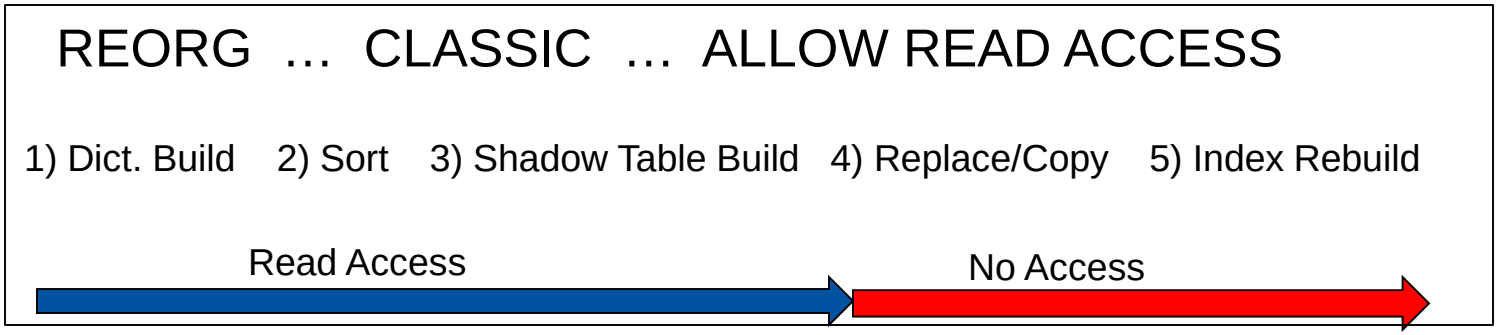


Table Reorg Concepts: ADMIN_MOVE_TABLE()

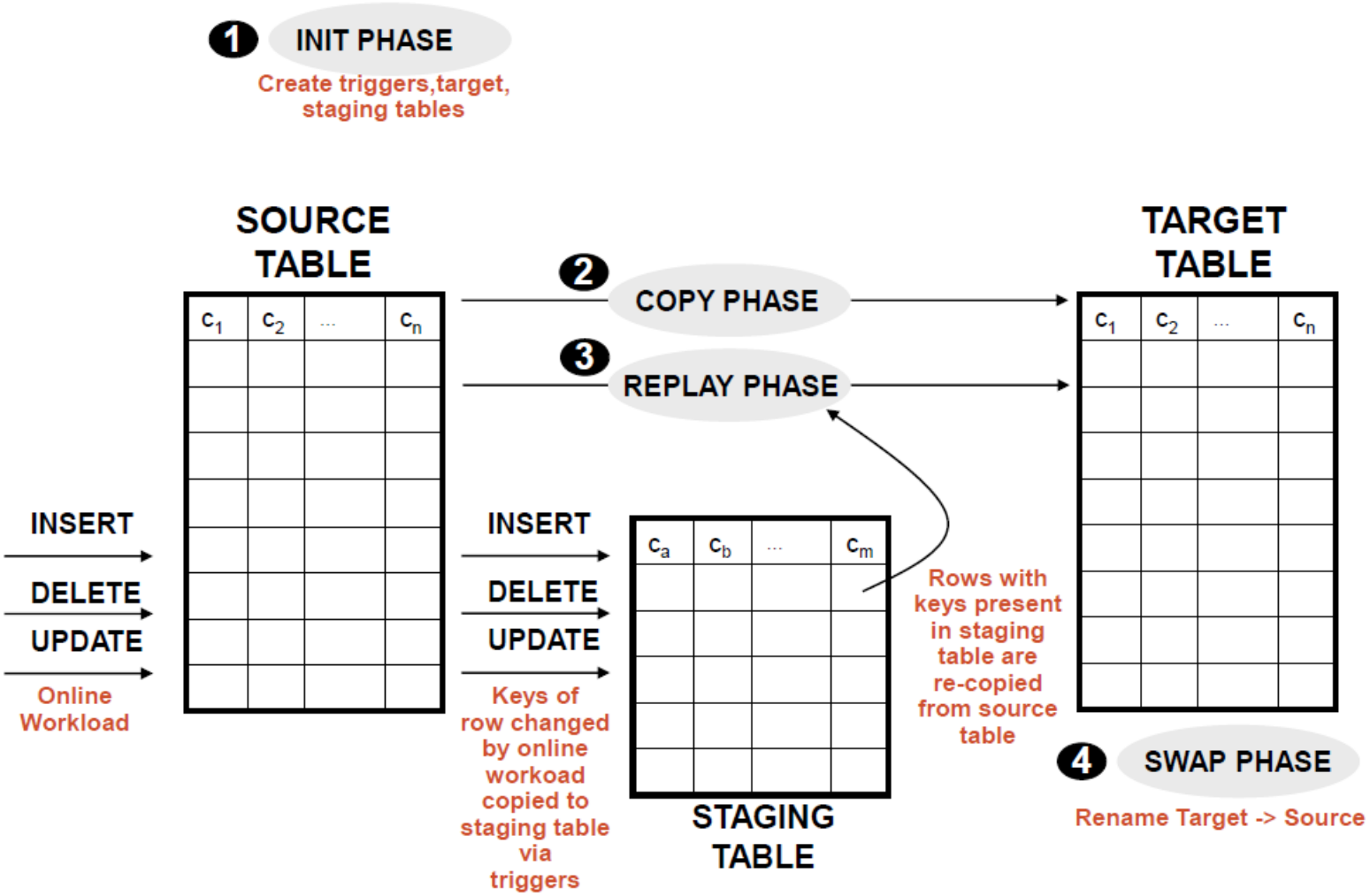


Table Reorg Concepts : Availability/Speed Comparison

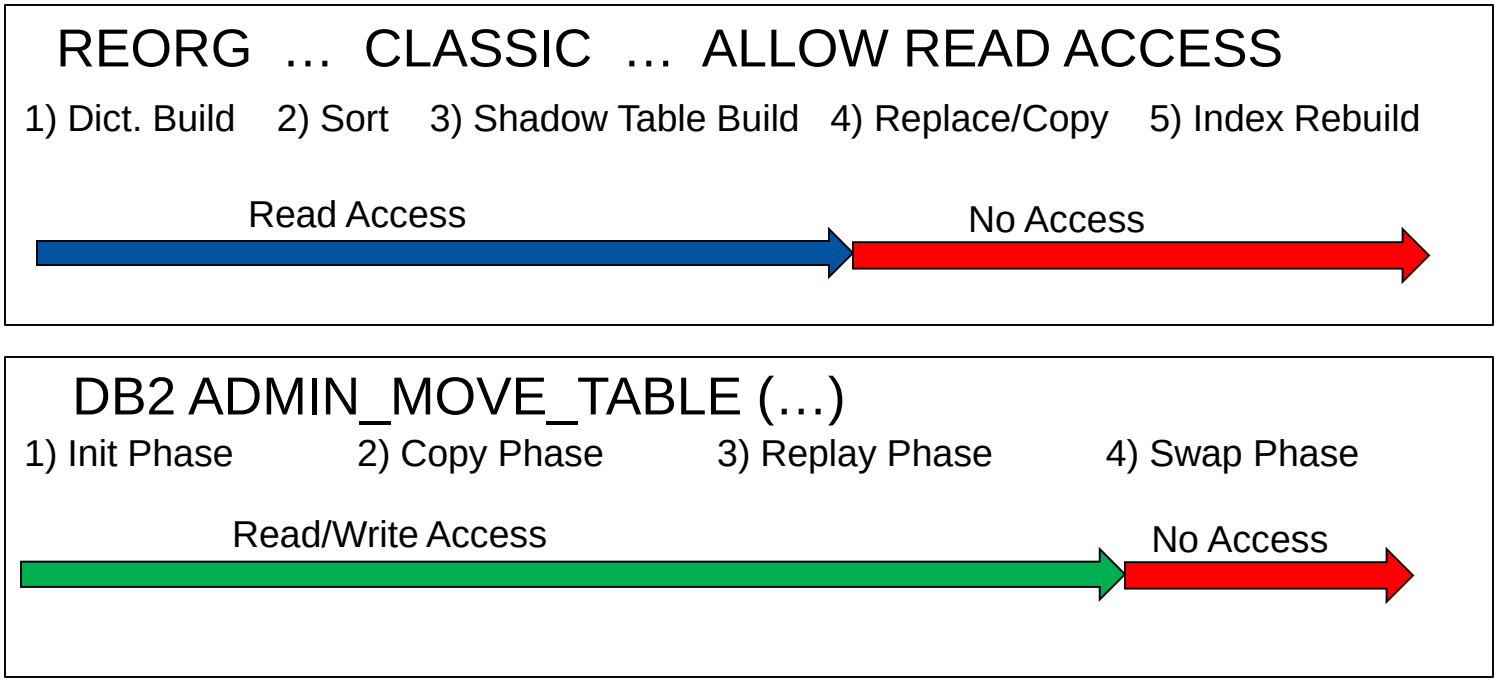
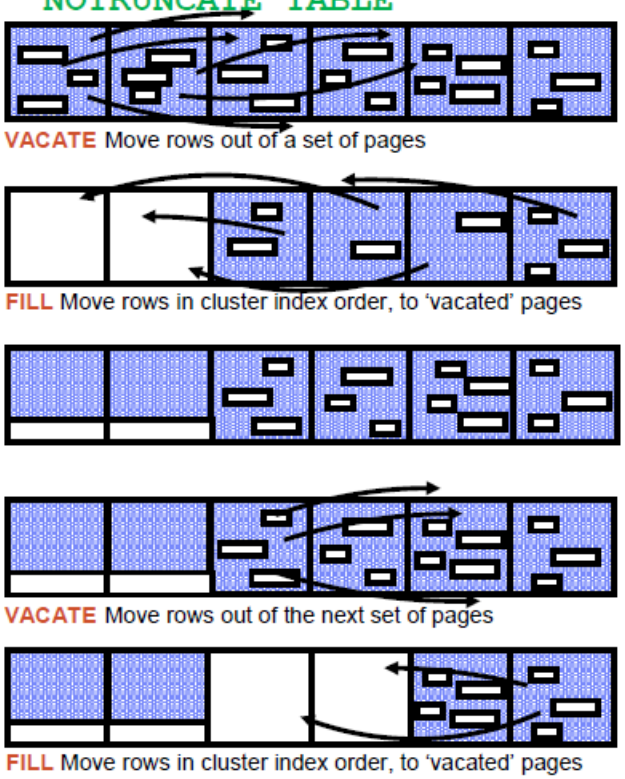


Table Reorg Concepts: Inplace Reorg

RECLUSTERING (invoked with Index)

```
DB2 REORG TABLE T1 INDEX I1 INPLACE
    NOTRUNCATE TABLE
```

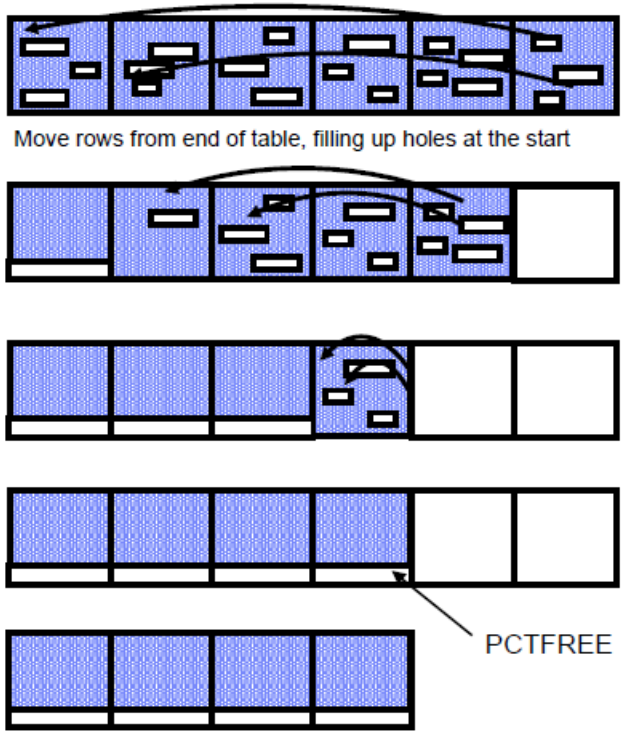


Uses clustering index during FILL phases to repopulate table with rows in index order. Very significant number of row moves.

TIME

SPACE RECLAMATION (no index)

```
DB2 REORG TABLE T1 INPLACE
    TRUNCATE TABLE
```



Backward scan starts at end, fills holes earlier in table identified by simultaneous forward scan. Can be significantly faster than the reclustering option, as it typically requires far fewer row moves.

Table Reorg Concepts : Availability/Speed Comparison

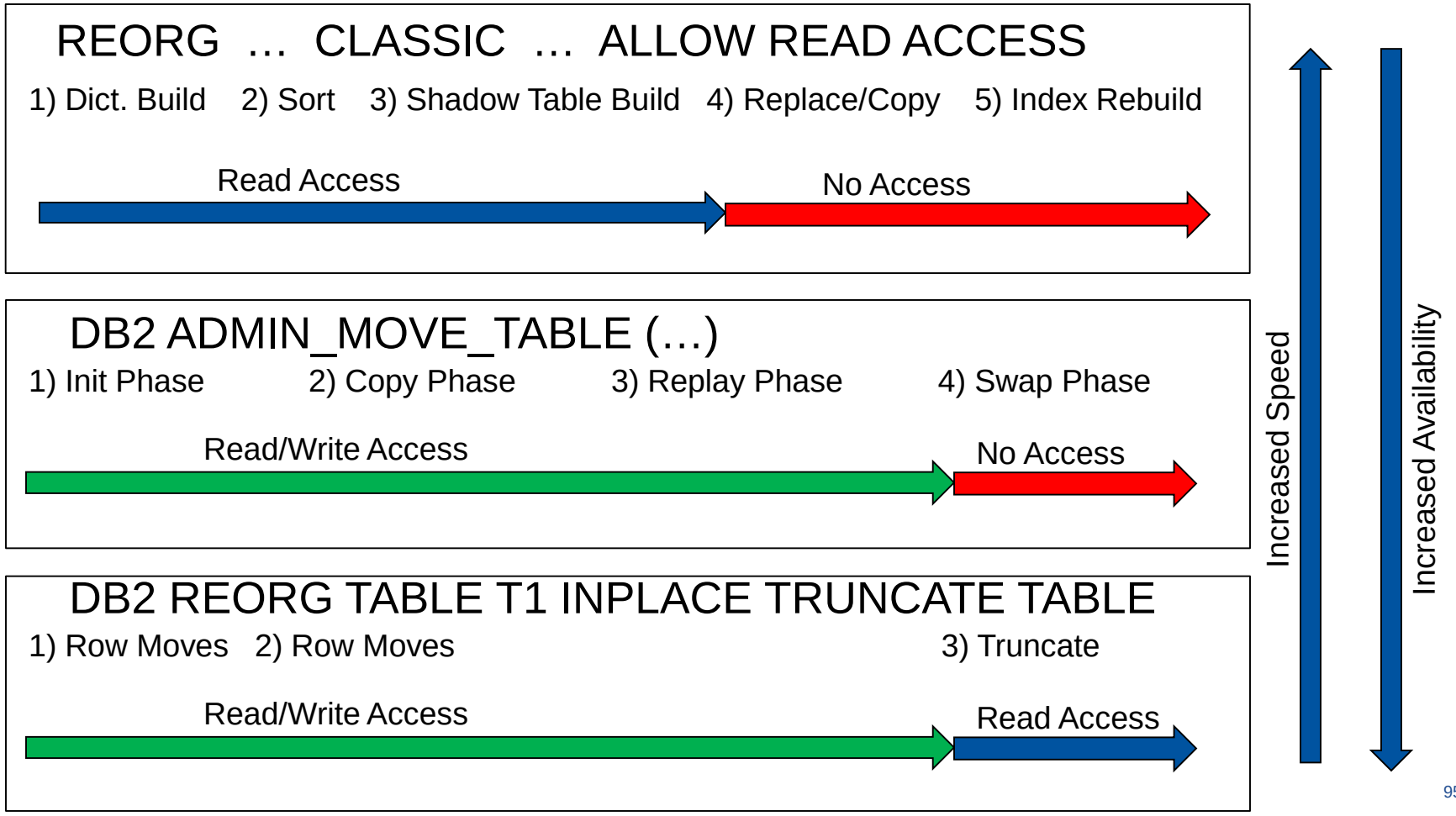


Table Reorg Best Practices

What	Details	*Potential* Benefits
Reclaim Space	Compact data onto fewer data pages Return extents to the tablespace	Faster table scans Can use the space in other tables
Recluster	Reorder rows into the same sequence as an index	Faster index scans where data fetch needed (aka "ISCAN-FETCH")
Remove Pointer & Overflows Records	Convert pointer overflow pairs into a single normal record	Faster row access
(Re-) Compress	(Re-) builds compression dictionary with latest data, and (re-) compresses data	Reduced storage consumption Reduced bufferpool consumption
Materialize Schema Changes	Physically materialize column changes - eg. physically adds a new column's default value to existing rows (vs dynamically constructing it)	Returns table to full table availability if limited due to ALTER TABLE Reduced CPU consumption
Conversion to Large RIDs	Conversion from 4-byte to 6-byte record IDs (applies only to tables created prior to 9.7)	Larger tables Greater than 255 rows per page

Use REORG … RECLAIM EXTENTS for fast, online space reclaim from tables

- **Without** `REORG … RECLAIM EXTENTS`, reclaiming space consumes significant CPU and IO resource, and requires at least an S lock
  - Inplace online REORG : most/all rows in the table moved within existing table; S lock for truncate
  - Classic REORG : new copy of table is populated, indexes created, S and/or excl. access
  - ADMIN_MOVE_TABLE : new copy of table is populated, indexes created, excl. access for SWAP

- **With** `REORG … RECLAIM EXTENTS`, reclaiming space is **very fast & fully online**
  - Minimal, 'surgically selected' row moves
  - Empty extents quickly found and returned to tablespace - **in place**
  - Works with Insert Time Cluster Tables, MDC Tables, BLU (Columnar) Tables, and Indexes

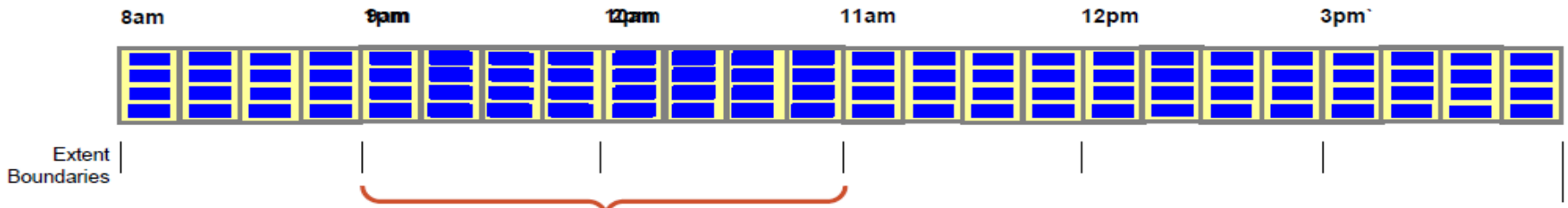
Use `RECLAIMABLE_SPACE` monitoring element as the motivator for this

```
SELECT TABNAME, RECLAIMABLE_SPACE FROM SYSIBMADM.ADMINTABINFO
  WHERE TABNAME LIKE 'mytable%' ORDER BY tabname WITH UR
```

Insert Time Clustered Tables - ITC

- Create table choice :
  `CREATE TABLE … ORGANIZE BY INSERT TIME`
- Designed to take advantage of a very common workload pattern:
  - Rows that are inserted at about the same time are often deleted at about the same time
  - e.g. DELETE WHERE *invoice date before Oct 1 2016*
- Changes DB2's insert algorithm from this …
  - *Find space for the row on any page in the table*

  … to this …
  - *Append the row to the table at the latest insert position for the table*
  - *The insert position starts at the end of the table, but can move to earlier extents after deletions*
- Result: many extents naturally become empty after row deletions
- Invoke extent reclamation explicitly (or rely on Automatic Table Maintenance daemon)



1) INSERTS …
2) DELETE WHERE …
3) REORG … **RECLAIM EXTENTS**
4) MORE INSERTS

Can't use BLU, MDC, ITC ?

- Hints/tips for space mgt/reclaim in a regular row organized table
  - Do you really need to reclaim the space now ?
    - Will subsequent INSERTs into the table need the space in the future anyways ?
  - Use `DB2MAXFSCRSEARCH` to optimize the balance between insert *speed* and how *exhaustively* DB2 searches for space
    - Specifies the number of 500-page units DB2 will search during INSERT before appending new extents to the table
      - Default is 5 (i.e. 2500 pages are searched for free space, before new extents are added)
      - Set it to -1 to tell DB2 to always search the entire table during INSERT
    - **Use higher values (or -1) to tell DB2 to more aggressively reuse space during INSERT**

- Take advantage of per-partition classic reorg, eg.

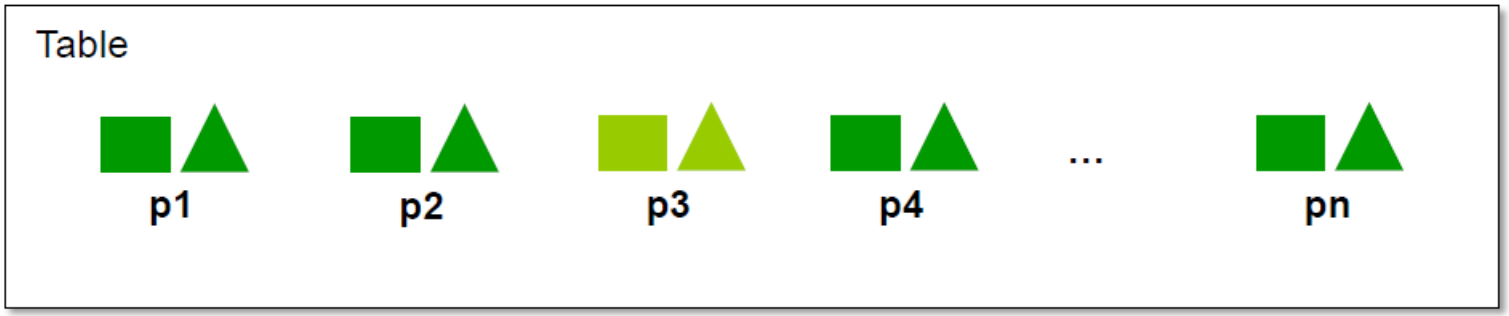
```
REORG ... ALLOW READ ACCESS ... ON DATA PARTITION p3 ...
```

  - Avoid global (i.e. non-partitioned) indexes, if possible
  - With no global indexes
    - The above `REORG` operation **will leave other partitions fully available**
    - You can launch parallel `REORG`s against separate partitions (with `ALLOW NO ACCESS`)

Can't use BLU, MDC, ITC ? (continued)

- Take advantage of per-partition **inplace online** reorg in V11

```
REORG … INPLACE … ALLOW WRITE ACCESS …
      … ON DATA PARTITION p3 …
```



- Currently requires that no global global (i.e. non-partitioned) indexes exist
- Works with other "inplace" options – e.g. CLEANUP OVERFLOWS

Table Reorg Best Practices

What	Details	*Potential* Benefits
Reclaim Space	Compact data onto fewer data pages Return extents to the tablespace	Faster table scans Can use the space in other tables
Recluster	Reorder rows into the same sequence as an index	Faster index scans where data fetch needed (aka "ISCAN-FETCH")
Remove Pointer & Overflows Records	Convert pointer overflow pairs into a single normal record	Faster row access
(Re-) Compress	(Re-) builds compression dictionary with latest data, and (re-) compresses data	Reduced storage consumption Reduced bufferpool consumption
Materialize Schema Changes	Physically materialize column changes - eg. physically adds a new column's default value to existing rows (vs dynamically constructing it)	Returns table to full table availability if limited due to ALTER TABLE Reduced CPU consumption
Conversion to Large RIDs	Conversion from 4-byte to 6-byte record IDs (applies only to tables created prior to 9.7)	Larger tables Greater than 255 rows per page

Use DB2's reorg avoidance capabilities to minimize or eliminate the need to recluster

- BLU tables

  `CREATE TABLE ... ORGANIZE BY COLUMN ...`

  - Introduced in V10.5
  - New columnar technology designed in-memory analytics
  - Access plans maintain exceptional performance without the need to cluster data relative to an index
  - Fast online space reclaim can be done with **`REORG ... RECLAIM EXTENTS`**

- Multi-dimensional clustering (MDC) tables

  `CREATE TABLE ... ORGANIZE BY DIMENSIONS ...`

  - Introduced in V8.1
  - Clustering is always permanently maintained during run-time
  - No need to reorganize for the purposes of reclustering
  - Fast online space reclaim can be done with **`REORG ... RECLAIM EXTENTS`**

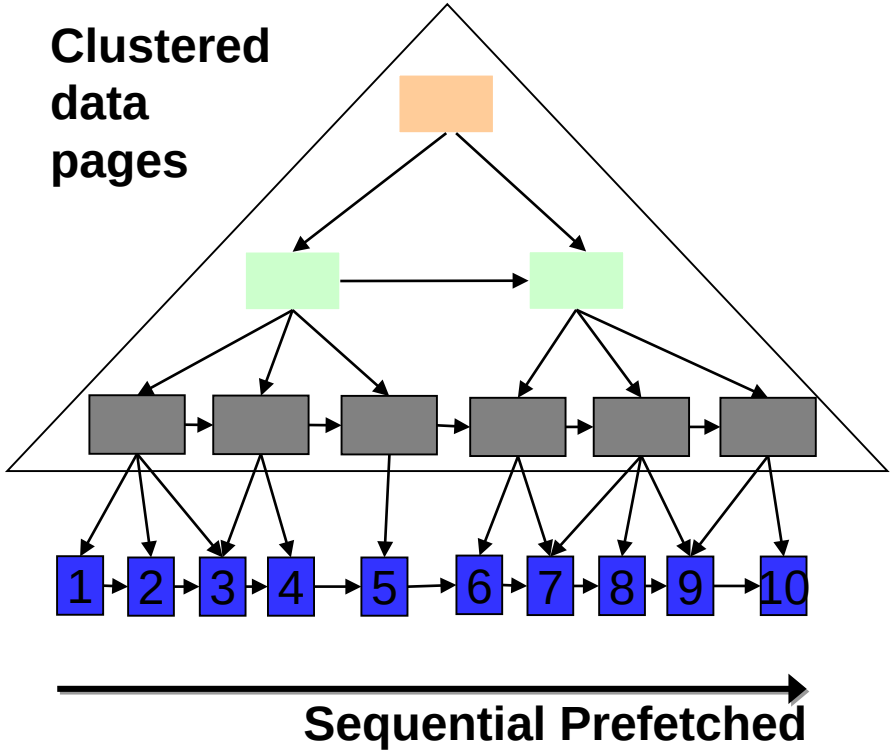
- Read-ahead prefetching
  - Introduced in V10.1 - enabled automatically if **`SEQDETCT = YES`**
  - New prefetching technology for both ISCAN-FETCH and ISCAN-only plans that maintains good performance despite de-clustered data
  - Significantly reduces the need for re-clustering reorgs
  - More on this coming up

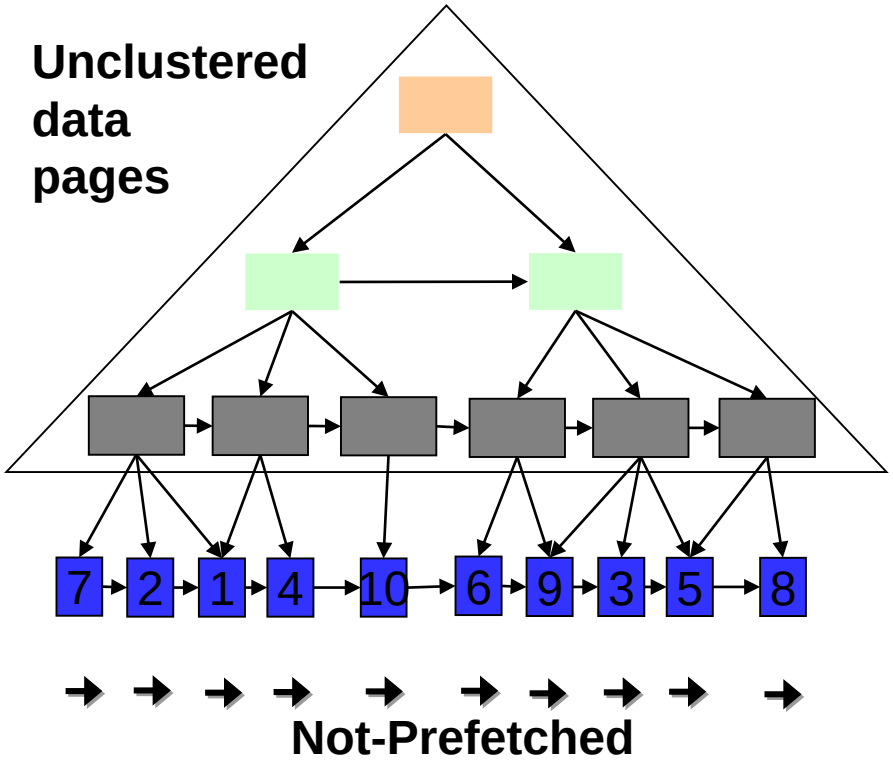
- Consider tactics regarding optimizer plan selection …
  - Are you performing reclustering reorgs to change statistics such as cluster ratio, to influence plan selection ?
    - eg. to prefer an ISCAN vs table scan ?
  - If so, consider other ways to influence plan selection that are less expensive and more online
    - eg. Volatile table, plan guidelines (eg. `<IXSCAN TABLE='S' INDEX='I_SUPPKEY'/>`

Read Ahead Prefetching : Motivation

- Prior to V10, sequential read-ahead prefetching requires clustered data pages

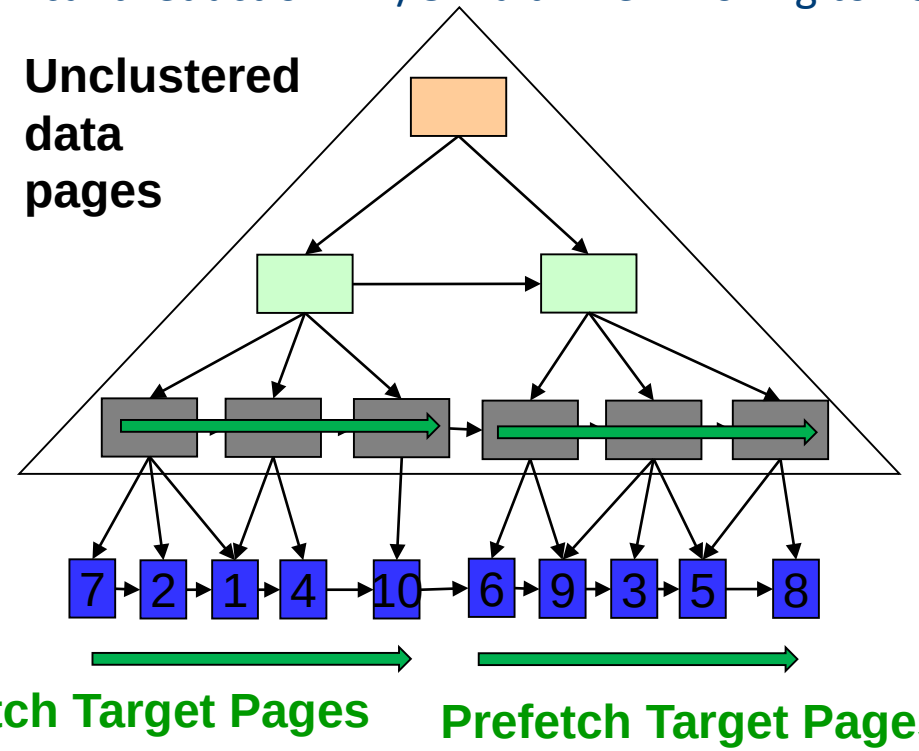
```
CREATE INDEX i1 ON t1 (c1)
SELECT * FROM t1 WHERE c1>0
```





Read Ahead Prefetching : How ?

- In V 10 and beyond, read ahead prefetching enables prefetching for non-sequential or unclustered pages
- Collect data page numbers one level up (index leaf pages) and issue multiple data page I/Os in each prefetch I/O
- Result: very significant reduction in I/O wait when moving to next record



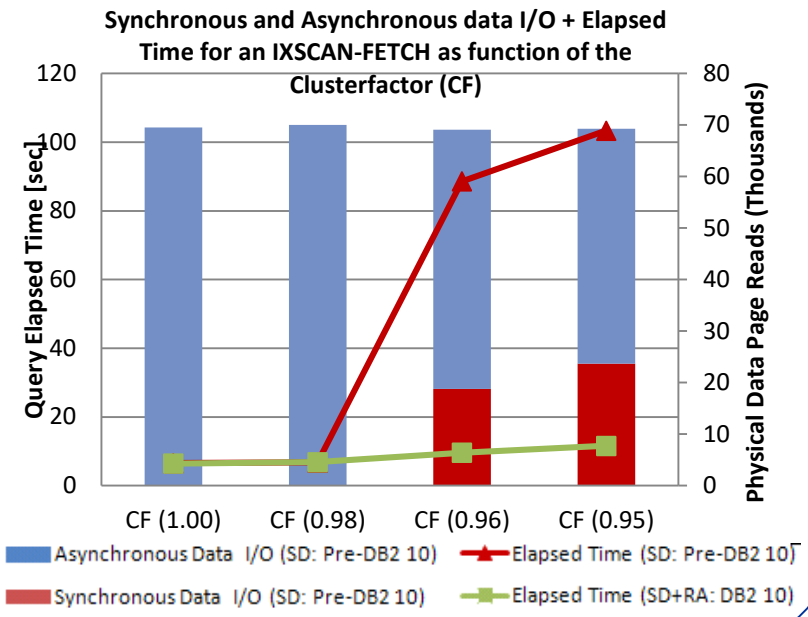
The example here uses an ISCAN-FETCH plan. Read-ahead prefetching can also work with ISCAN only, where the 'read-ahead' occurs at the next-to-leaf pages in the Btree.

Read Ahead Prefetching: Example Benefit

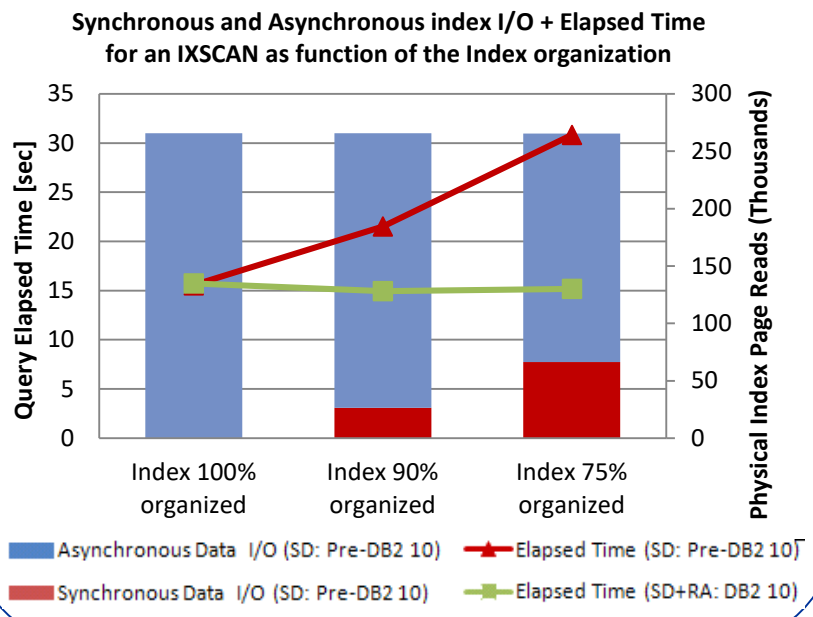
Queries run up to 90% faster in cold bufferpool environment

Queries run up to 65% faster in cold bufferpool environment

Table Scan



Index Scan



As of V 10.1, DB2 performs read ahead and sequential prefetching automatically, as appropriate

If the prefetch explain operator value shows READHEAD or SEQUENTIAL, READHEAD the plan is eligible for readahead prefetch

If the prefetch explain operator value shows READHEAD or SEQUENTIAL, READHEAD the plan is eligible for readahead prefetch

Table Reorg Best Practices

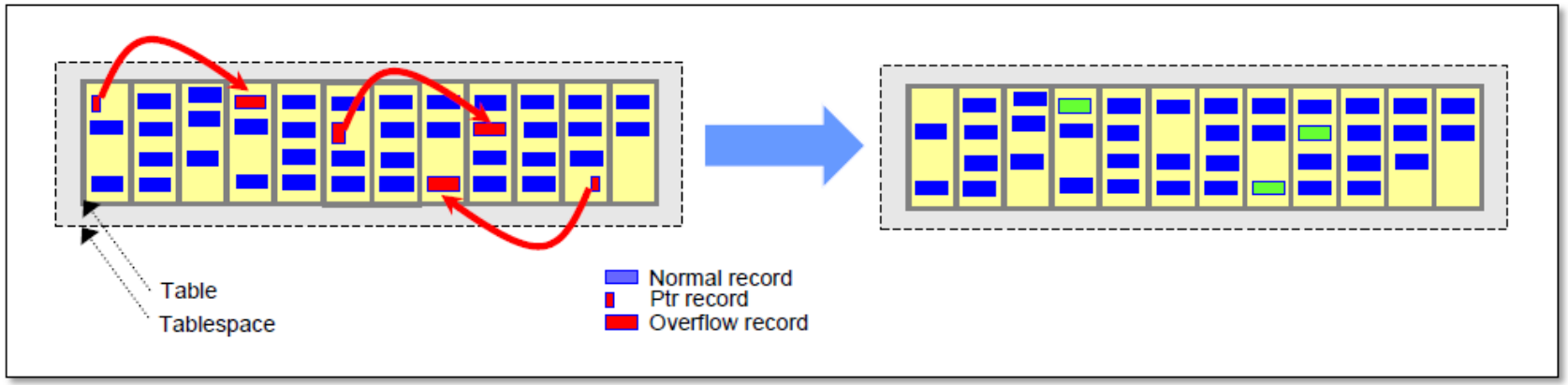
What	Details	*Potential* Benefits
Reclaim Space	Compact data onto fewer data pages Return extents to the tablespace	Faster table scans Can use the space in other tables
Recluster	Reorder rows into the same sequence as an index	Faster index scans where data fetch needed (aka "ISCAN-FETCH")
Remove Pointer & Overflows Records	Convert pointer overflow pairs into a single normal record	Faster row access
(Re-) Compress	(Re-) builds compression dictionary with latest data, and (re-) compresses data	Reduced storage consumption Reduced bufferpool consumption
Materialize Schema Changes	Physically materialize column changes - eg. physically adds a new column's default value to existing rows (vs dynamically constructing it)	Returns table to full table availability if limited due to ALTER TABLE Reduced CPU consumption
Conversion to Large RIDs	Conversion from 4-byte to 6-byte record IDs (applies only to tables created prior to 9.7)	Larger tables Greater than 255 rows per page

Use REORG … INPLACE … CLEANUP OVERFLOWS

- This option of inplace reorg was added in V10.5

  `REORG … INPLACE … CLEANUP OVERFLOWS`

  - Optimized implementation of inplace reorg that targets **only** pointer / overflow pairs
  - Finds them, and converts each pair to a single normal record
  - Done while maintaining full read/write access
  - Note : does not return space to tablespace



Consider using overflow ***accesses*** as the motivator for this (as opposed to their ***existence***)

- Runtime accesses : OVERFLOW_ACCESSES from MON_GET_TABLE()
- Existing overflows : OVERFLOW column in the SYSSTAT.TABLES view

INPLACE Table Reorganization

- The manageability of large range partitioned tables is improved
  - A single partition of a partitioned table can now be reorganized with the INPLACE option if:
    - the table has no non-partitioned indexes
    - ON DATA PARTITION is specified
  - Only one data partition can be reorganized at a time
  - INPLACE table reorganization can be run only on tables that are at least three pages in size

Table Reorg Best Practices

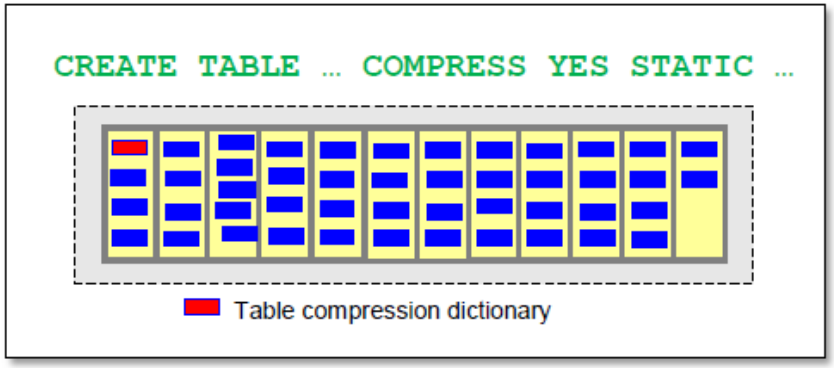
What	Details	*Potential* Benefits
Reclaim Space	Compact data onto fewer data pages Return extents to the tablespace	Faster table scans Can use the space in other tables
Recluster	Reorder rows into the same sequence as an index	Faster index scans where data fetch needed (aka "ISCAN-FETCH")
Remove Pointer & Overflows Records	Convert pointer overflow pairs into a single normal record	Faster row access
(Re-) Compress	(Re-) builds compression dictionary with latest data, and (re-) compresses data	Reduced storage consumption Reduced bufferpool consumption
Materialize Schema Changes	Physically materialize column changes - eg. physically adds a new column's default value to existing rows (vs dynamically constructing it)	Returns table to full table availability if limited due to ALTER TABLE Reduced CPU consumption
Conversion to Large RIDs	Conversion from 4-byte to 6-byte record IDs (applies only to tables created prior to 9.7)	Larger tables Greater than 255 rows per page

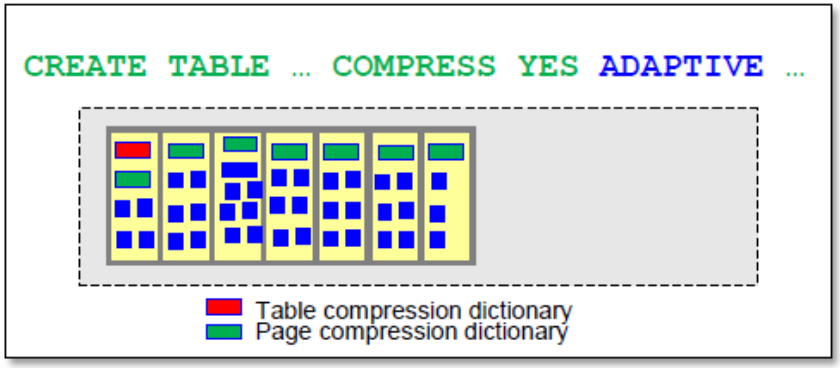
Use adaptive compression to minimize the need to rebuild the table level compression dictionary

- Create / Alter table option

```
CREATE / ALTER TABLE … COMPRESS YES ADAPTIVE …
```

  - Allows a page-level dictionary to be created for each data page
  - Allows compression rates to adapt to changing data patterns
  - Now compatible with INPLACE reorg (as of 10.5)





- Still useful to periodically check if a table level dictionary rebuild will help

```
SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO ('SCHEMA1', 'TABLE1')
```

  - Compare `PCTPAGESSAVED_CURRENT` to `PCTPAGESSAVED_ADAPTIVE`

Table Reorg Best Practices

What	Details	*Potential* Benefits
Reclaim Space	Compact data onto fewer data pages Return extents to the tablespace	Faster table scans Can use the space in other tables
Recluster	Reorder rows into the same sequence as an index	Faster index scans where data fetch needed (aka "ISCAN-FETCH")
Remove Pointer & Overflows Records	Convert pointer overflow pairs into a single normal record	Faster row access
(Re-) Compress	(Re-) builds compression dictionary with latest data, and (re-) compresses data	Reduced storage consumption Reduced bufferpool consumption
Materialize Schema Changes	Physically materialize column changes - eg. physically adds a new column's default value to existing rows (vs dynamically constructing it)	Returns table to full table availability if limited due to ALTER TABLE Reduced CPU consumption
Conversion to Large RIDs	Conversion from 4-byte to 6-byte record IDs (applies only to tables created prior to 9.7)	Larger tables Greater than 255 rows per page

Use ADMIN_MOVE_TABLE() to make schema changes, with minimal outage

- For example, here, for table MYSCHEMA.T1, we are changing the definition for each of columns C1, C2, C4, and dropping C3. Other options are defaulted.

```
CALL SYSPROC.ADMIN_MOVE_TABLE ( 'MYSCHEMA', 'T1', '', '', '',
                                '', '', '',
                               'C1 VARCHAR(100), C2 INT, C4 INT'
                               ,'', 'MOVE')
```

- If using ALTER TABLE directly, be aware of the implications of "reorg-recommended" ALTER TABLE operations:

Reorg Recommended ALTER TABLE Operations

DROP COLUMN
ALTER COLUMN SET/DROP NOT NULL
ALTER COLUMN SET DATA TYPE, except in the following situations:
- Increasing the length of a VARCHAR or VARGRAPHIC column
- Decreasing the length of a VARCHAR or VARGRAPHIC column without truncating data

- **After 1 transaction containing a reorg-recommended ALTER operation**, the table is placed in a reorg-recommended state - this allows table scans but not most other accesses
- **After 3 such ALTER transactions**, a CLASSIC REORG must be run before any further ALTERs

SYSIBMADM.ADMINTABINFO can be used to determine the number of reorg recommended ALTERs outstanding against a table.

New options for the ADMIN_MOVE_TABLE

- REPORT
  - The REPORT option can be used to monitor the progress of table moves
  - Calculates a set of values to monitor the progress of a single or multiple table moves
  - Focus is the COPY and REPLAY phase of a running table move
  - To get values for all table moves, tabschema and tabname must be NULL or the empty string
- TERM
  - The TERM option can be used to terminate a table move in progress
  - Terminates a running or table move
  - TERM will force off the application running the table move, roll back all open transactions and set the table move to a well defined operational status
  - From here, the table move can be canceled or continued
- SWAP – FORCE_ALL
  - ADMIN_MOVE_TABLE(<schema>,<table>,'','FORCE_ALL','SWAP')
  - Helps avoid deadlocks and timeouts due to other transactions with locks
  - Requires SYSADM, SYSCTRL, or SYSMAINT privileges

If necessary, consider a full reorg only as a last resort and use the most suitable method for your workload / scenario

- Best availability, slowest reorg :

  `REORG ... INPLACE ... [ NOTRUNCATE TABLE ]`

  - This 'shuffles' rows to the front of the table, creating empty pages at the end of the table, and then truncates those pages, returning them to the tablespace
  - Full read/write access is allowed throughout, except during the TRUNCATE operation which acquires a SHARE lock
  - Does not rebuild table dictionary; does not reorg LOBs
  - **Tip:** If you do not need the space returned to the tablespace use the NOTRUNCATE option to avoid the SHARE lock
  - **Tip:** If the SHARE lock is causing a concern, PAUSE the reorg, then RESUME it at a better time

- Good availability, medium reorg speed :

  `ADMIN_MOVE_TABLE()`

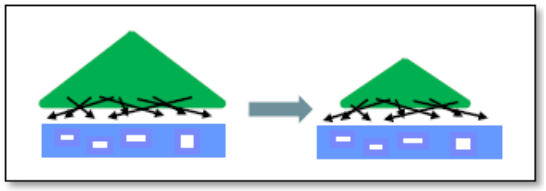
  - This builds another copy of the table and its indexes and then swaps the old table and indexes for the new.
  - Full read/write access is allowed throughout, except during the SWAP phase, which ensures all writes are applied to the new table
  - **Tip:** You can initiate the SWAP phase at a convenient (eg. low activity) time, by invoking ADMIN_MOVE_TABLE() separately for each phase (eg. INIT, COPY, REPLAY, SWAP) instead of a single MOVE invocation

- Minimal availability, fastest reorg :

  `REORG ... CLASSIC ... ALLOW READ ACCESS`

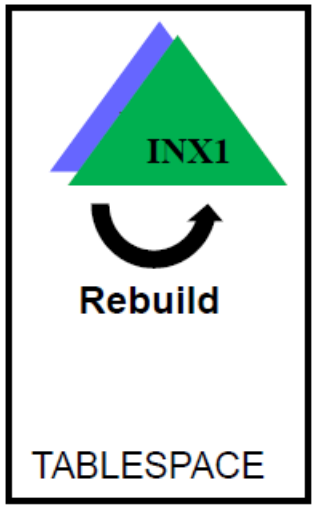
  - This builds another copy of the table, then rebuilds its indexes and finally swaps the old table and indexes for the new.
  - By default, no table access is allowed during the operation.
  - **Tip:** Use **`ALLOW READ ACCESS`** to allow read access until the REPLACE phase
  - **Tip:** If you can afford the extra space in your tablespace(s) perform the reorg without a TEMP tablespace to avoid the copy phase

Index Reorg Concepts



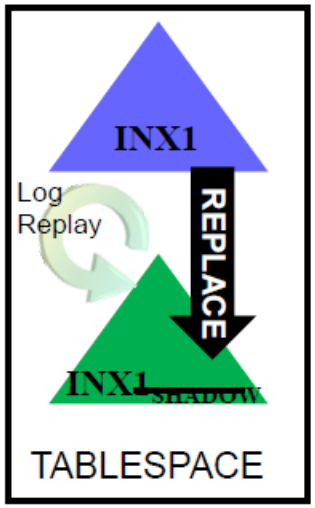
REORG INDEXES FOR TABLE ...

... ALLOW NO ACCESS



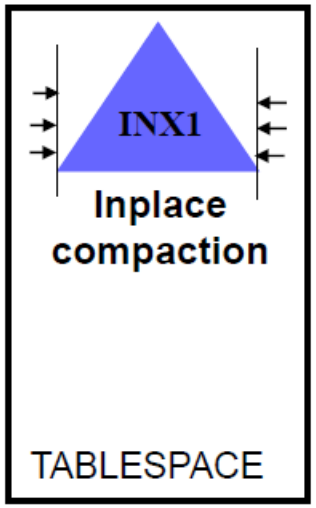
- Index rebuilt
- Inplace in same tablespace

... ALLOW WRITE ACCESS



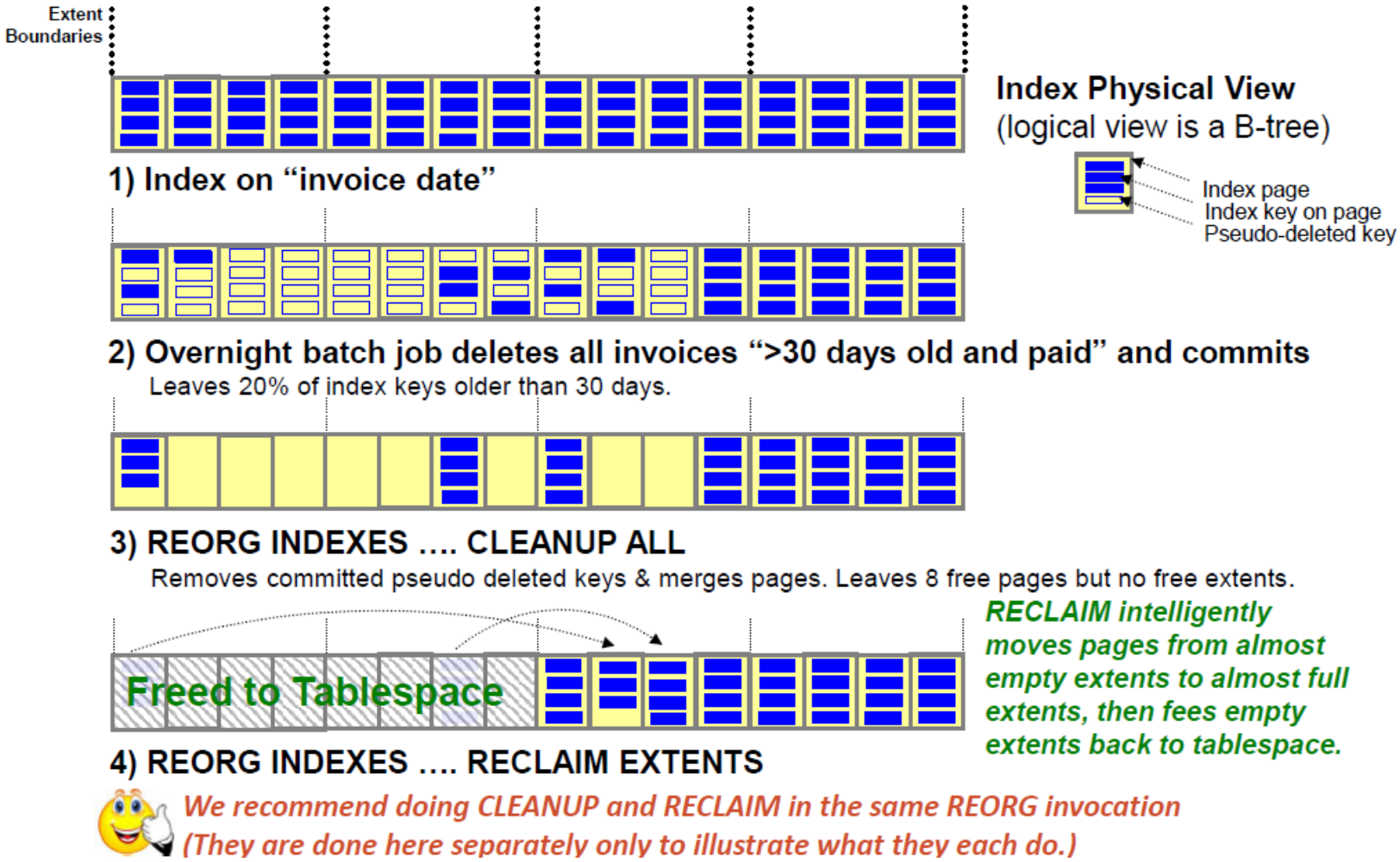
- Shadow index built
- Iterative log apply to bring shadow index up to date
- Replace index with shadow (period of no access)

... ALLOW WRITE ACCESS CLEANUP RECLAIM EXTENTS



- No index rebuild
- Pseudo-deleted keys cleaned up
- Selected page merges to form empty extents which are returned to tablespace
- All inplace in index

Use REORG ... CLEANUP ALL RECLAIM EXTENTS for fast and online index reclaim



Other Index Reorg Hints and Tips

- When to run **CLEANUP ALL RECLAIM EXTENTS** ?
  - Significant reclaimable space reported through **ADMIN_GET_INDEX_INFO()**

```
SELECT SUBSTR(INDNAME,1,10) AS INDNAME, INDEX_OBJECT_P_SIZE,
          RECLAIMABLE_SPACE FROM TABLE(SYSPROC.ADMIN_GET_INDEX_INFO('','OLIVIA','T1'))
          AS INDEXINFO WHERE INDNAME='INX1'

INDNAME    INDEX_OBJECT_P_SIZE  RECLAIMABLE_SPACE
---------- -------------------- --------------------
INX1       1106752              846592


REORG INDEXES ALL FOR TABLE T1 ALLOW WRITE ACCESS CLEANUP ALL RECLAIM EXTENTS



SELECT SUBSTR(INDNAME,1,10) AS INDNAME, INDEX_OBJECT_P_SIZE,
          RECLAIMABLE_SPACE FROM TABLE(SYSPROC.ADMIN_GET_INDEX_INFO('','OLIVIA','T1'))
          AS INDEXINFO WHERE INDNAME='INX1'

INDNAME    INDEX_OBJECT_P_SIZE  RECLAIMABLE_SPACE
---------- -------------------- --------------------
INX1       259776               0
```

Other Index Reorg Hints and Tips (continued)

- Use an index rebuild operation only as a last resort
  - Many DB2 installations now use CLEANUP and RECLAIM EXTENTS exclusively
  - The benefit of reordering index pages is significantly reduced with read-ahead prefetching in 10.1 and up

- If an index rebuild does become necessary and availability is critical, use

```
REORG … ALLOW WRITE ACCESS
```

  - Ensure you have a healthy active log defined to accommodate the lengthy transactions and online workload requirements
  - Ensure you have sufficient space in your index tablespace

- For very large tables, consider using data partitioning (aka range partitioning) to help "divide-and-conquer" your reorgs, for eg.
  - For local (aka partitioned) indexes, use per-partition index reorg

```
REORG INDEXES … ALLOW WRITE ACCESS … ON DATA PARTITION p1 …
```

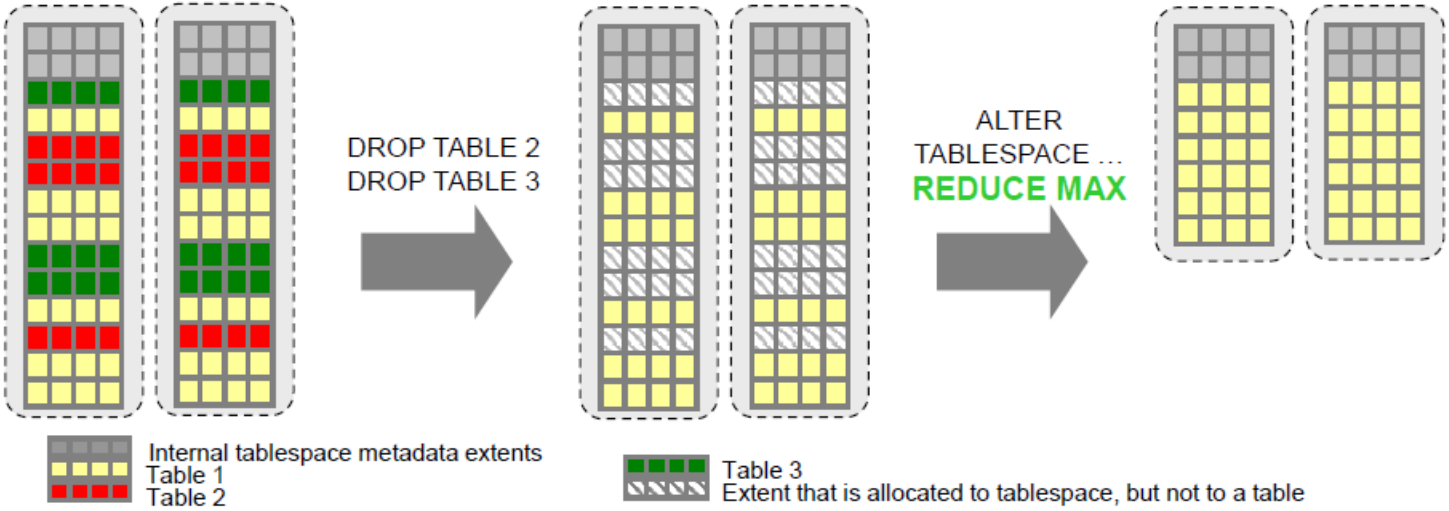
    - Reduces amount of additional log and tablespace space needed
    - Allows parallel invocation on different data partitions
  - For global (aka non-partitioned) indexes, user per-index reorg

```
REORG INDEX I1 … ALLOW WRITE ACCESS
REORG INDEX I2 … ALLOW WRITE ACCESS
```

  - **Tip:** consider a **single-partition** data-partitioned table to get per-index reorg

```
CREATE TABLE T1(C1 INT)PARTITION BY RANGE … STARTING FROM (MINVALUE) ENDING(MAXVALUE)
```

Use ALTER TABLESPACE … REDUCE MAX to return free space to the file system



- Moves used extents from higher addresses in the tablespace to unused lower addresses
- Shrinks/removes containers to return space back to the file system(s)
- Can specify amount (size) to free up, or specify as much as possible (MAX)
- Runs in the background
  - Works in batches, committing free extents as it progresses
- Hints/Tips:
  - Runs online; can use **STOP** option if I/O bandwidth consumption is a concern
  - Can monitor progress with `MON_GET_EXTENT_MOVEMENT_STATUS()`
  - Tablespace storage is reclaimable if the tablespace was created on V9.7 or later