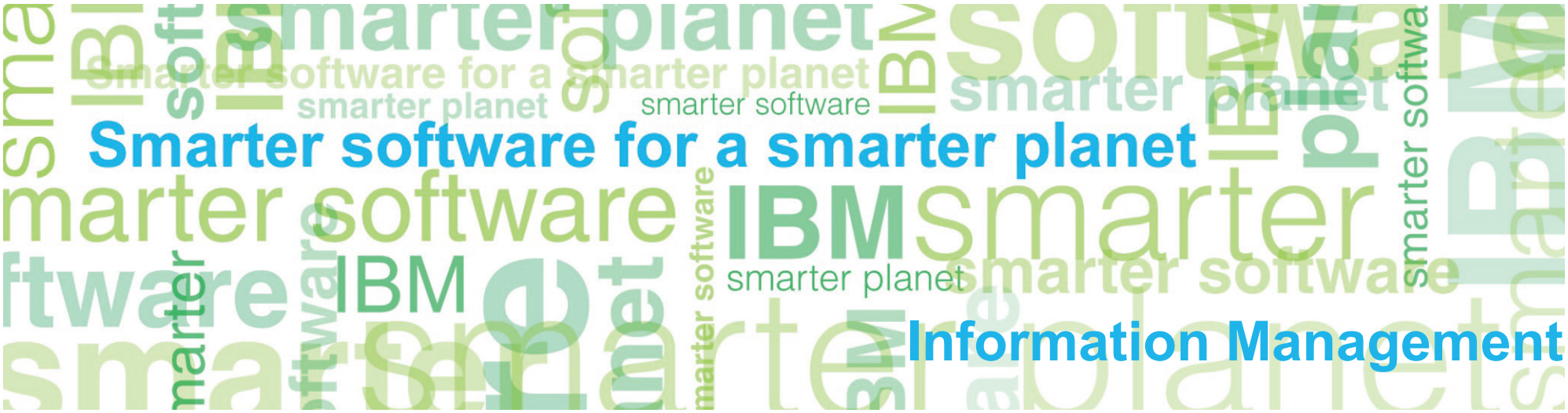
DB2® 10 for Linux, UNIX and Windows Fundamentals



Agenda

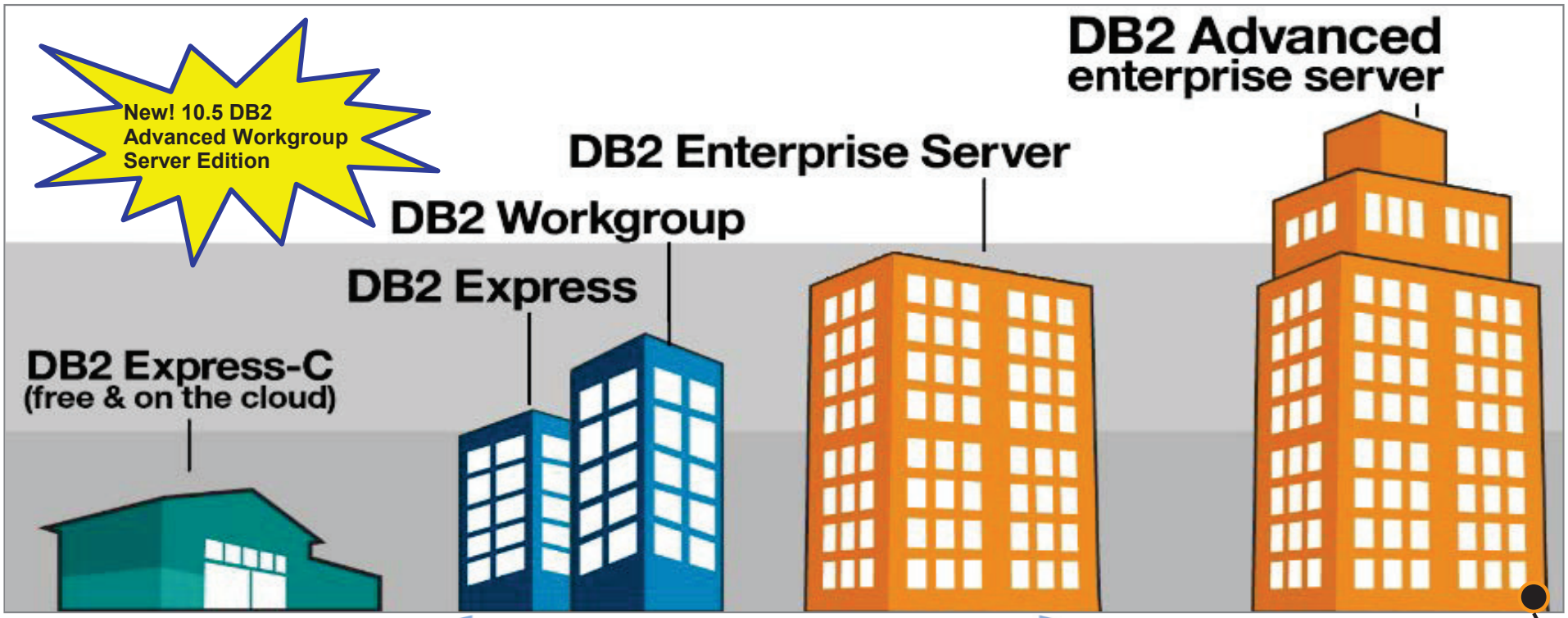
- **Product Overview**
  - Editions & Features
  - DB2 on Cloud
  - Licensing
- **Fundamentals**
  - Users
  - Architecture
    - Data server, Instances, Databases, Process model
  - Diagnostics Log
  - IBM Data Server Drivers and Clients
  - Database Fundamentals
    - Creating databases
    - Table spaces and Buffer pools
  - Security
  - Application Development and Language Support
  - Tools, Commands and APIs

DB2 for LUW Overview

- **Database server optimized to deliver high performance across multiple workloads.**
- **DB2 for LUW V10 officially released April 2012**
- **Full Multi-Platform Support**
  - Linux, UNIX (AIX, HP-UX, Solaris)
  - Windows 2003, 2008, XP, Vista, 7
- **Common code base "DB2 is DB2 is DB2"**
  - No need to port between platforms
  - New versions available on all platforms at the same time
- **Multiple editions for different business environments**

http://www.ibm.com/db2/linux-unix-windows/

DB2 Database Product Editions



DB2 AESE

- Storage Optimization
- Continuous Data Ingest
- Q-replication
- Federation
- Optim & InfoSphere tools

Database Enterprise Developer Edition

Allows developers to design, build, and prototype applications. The edition is a product bundle that includes many DB2 features.

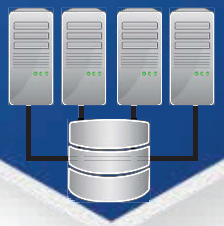
Highlights of Features

Storage Optimization
- Adaptive row compression
- Index compression
- Temporary table compression
- XML compression

Advanced Data Security Control
- Row and Column Access Control (RCAC)
- Label-based Access Control (LBAC)

Performance Optimization
- Multi-core parallelism
- Star schema query optimization
- Smart data and index prefetching

DB2 pureScale
- Clustering technology
- Extreme capacity
- High availability
- Application transparency



Manageability
- Multi-temperature storage
- Temporal tables
- Autonomic features
- WLM dispatcher

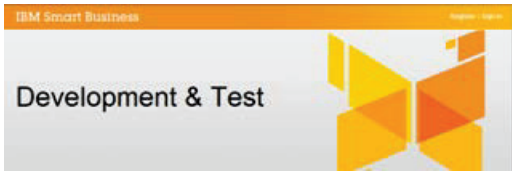
DB2 on the Cloud

IBM Cloud: IBM Smart Business Development and Test

- Dynamically provisioned and scaled runtime environment with tools for developing

and testing applications

- ✓ Instant self-service provisioning of software images
- ✓ lower TCO
- ✓ Dynamic/elastic computing for tests
- ✓ Virtualized development infrastructure to build solutions
- ✓ Flexible deployment options
- ✓ Can have IBM create this on a private cloud for you



https://www.ibm.com/cloud/enterprise/dashboard

IBM Partnership: Amazon Web Services

- IBM software in the Amazon Elastic Compute Cloud (EC2) virtual environment
- Pre-configured DB2 Amazon Machine Images (AMI)
- Product-level code with all features and options enabled
- Large breadth of IBM offerings on AWS from brands such as Information Management, WebSphere, Tivoli, and Lotus
  - ✓ Flexible payment options
  - ✓ Free for development
  - ✓ Pay-as-you-go for production
  - ✓ Bring your own license
  - ✓ Go to market on AWS with your value-added offerings on top of IBM software
  - ✓ Detailed guides to help you deploy and re-bundle IBM software on AWS

IBM Partnership: DB2 on GoGrid Cloud

- DB2 GoGrid Server Images (GSI) are pre-built cloud server images that include all the necessary operating system, middleware, and tools that ISVS are looking for in a database cloud solution
- GoGrid has many cloud specific features that set it apart from its competitors

http://go.gogrid.com/ibm-isv

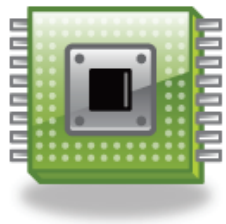
DB2 Key Features and Functionality by Edition

Functionality	DB2 Personal	DB2 Express-C	Express (incl. FTL)	Workgroup Server	Enterprise Server	Advanced Workgroup Server	Advanced Enterprise Server	Enterprise Developer
Advanced Copy Services	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Adaptive Compression and classic row compression	No	No	No	No	DB2 Storage Optimization Feature	Yes	Yes	Yes
Compression: backup	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Continuous Data Ingest	No	No	No	No	No	Yes	Yes	Yes
DB2 pureScale functionality	No	No	No	Up to 16 cores and 64GB of total cluster size	DB2 pureScale Feature	**DB2 pureScale Feature**	DB2 pureScale Feature	Yes
Federation with DB2 LUW and Informix Data Server	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Federation with DB2 LUW and Oracle	No	No	No	No	No	Yes	Yes	Yes
High availability disaster recovery (HADR)	No	No	Yes	Yes	Yes	Yes	Yes	Yes
IBM Data Studio	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
BM InfoSphere® Optim™ Configuration Manager	No	No	No	No	No	Yes	Yes	Yes
IBM InfoSphere Optim Performance Manager Extended	No	No	No	No	No	Yes	Yes	Yes
IBM InfoSphere Data Architect	No	No	No	No	No	Yes(10 licenses)	Yes (10 licenses)	Yes
IBM InfoSphere Optim Query Workload Tuner	No	No	No	No	No	Yes	Yes	Yes
LBAC / RCAC	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Materialized query tables (MQTs) Multidimensional clustering (MDC) tables	Yes	No	No	No	Yes	Yes	Yes	Yes
Multi-Temperature Storage	No	No	No	No	Yes	Yes	Yes	Yes
Online reorganization	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Oracle Compatibility	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
pureXML® storage	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Q Replication with two other DB2 LUW servers	No	No	No	No	No	Yes	Yes	Yes
Query parallelism	Yes	No	No	No	Yes	Yes	Yes	Yes
Replication tools	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SQL Replication between DB2 LUW and Informix	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes
Time Travel Query	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Table partitioning	No	No	No	No	Yes	Yes	Yes	Yes
Tivoli® System Automation	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Workload management	No	No	No	No	Yes	Yes	Yes	Yes

Licensing

- **Authorized user**
  - IDs cannot be shared or transferred
  - Can establish one or more connections to the DB2 database system and counts as a single authorized user
  - ID is needed for each data server

- **Processor Value Unit (PVU)**
  - PVU is a unit of measure that is assigned to each processor core, depending on vendor, brand, type and model number
  - Sub-capacity Licensing: Enables the licensing of DB2 to a subset of the processor cores on the server
  - Allows unlimited users to access DB2 on that server
- **Per Socket**
  - Socket is defined as electronic circuitry that accepts a processor chip
  - Only available for Workgroup Edition
  - Allows unlimited users to access DB2 on that server
  - Limit to 64 GB of memory and 4 sockets on a physical server



Licensing (continued)

- **Per Server (virtual / physical)**
  - Limited use virtual server (LUV server): is a physical server OR a virtual server that is created by partitioning the resources available to a physical server
  - Only available for DB2 Express Edition
  - Allows unlimited users to access DB2 on that server
  - All instances cannot collectively exceed 4 processor cores and 4 GB of memory
- **Per ASL / OEM**
  - ASL
    - DB2 is restricted use and can only be used as part of the solution.
    - The Business Partner can ship DB2 as part of its application worldwide.
    - The Business Partner retains the licenses to DB2
  - OEM
    - DB2 not visible to the user
    - Partner lead sales
    - Embed DB2 with application
    - Simplified contracts – 1 for the solution
    - ISV owns relationship completely

- **Per usage**
  - Software as a Service Monthly Rental Model
    - Variable or Committed
  - Amazon Machine Images
    - “Bring your own IBM license” or Hourly pricing

Check the type of license applied with command: `db2licm -l`

Licensing – Metrics and Summary

	Personal	Express-C	Express	Workgroup	Advanced Workgroup	Enterprise / Advanced
Pricing metric	Per install (Assumes one user)	Free Download (Unsupported)	Authorized Users (minimum of 5 per server) or Per Server	Authorized Users (minimum of 5 per socket) or Per Socket	Authorized Users (minimum of 25 per 100 PVUs) or PVUs Eligible for Sub-capacity pricing	Authorized Users (minimum of 25 per 100 PVUs) or PVUs Eligible for Sub-capacity pricing
Processor limit	N/A	DB2 throttles itself to use maximum of 2 cores	DB2 throttles itself to use maximum of 4 cores	DB2 throttles itself to use maximum of 16 cores and 4 sockets	DB2 throttles itself to use maximum of 16 cores	Unlimited
Memory limit	N/A	DB2 throttles itself to use maximum of 4 GB	DB2 throttles itself to use a maximum of 8 GB	DB2 throttles itself to use a maximum of 64GB	DB2 throttles itself to use a maximum of 128 GB	Unlimited
Platforms supported	Windows, Linux	Windows, Linux, Solaris (x64)	Windows, Linux, Solaris (x64)	Windows, Linux, AIX, Solaris, HP-UX, zLinux	Windows, Linux, AIX, Solaris, HP-UX, zLinux	Windows, Linux, AIX, Solaris, HP-UX, zLinux

DB2 Installation

- **New in DB2 10:**
  - You can install the IBM® DB2 pureScale Feature while installing DB2 Enterprise Server Edition, DB2 Workgroup Server Edition, DB2 Advanced Workgroup Server Edition and DB2 Advanced Enterprise Server Edition.
  - You can now install IBM Data Studio from the DB2 Launchpad.

Installation Methods

Installation	Windows	UNIX
db2setup Wizard	✓	✓
db2_install command	✗	✓
Response file	✓	✓

⚠ **Deprecated in DB2 10!**

DB2 Installation – DB2 Users *(non-pureScale)*

- **On Linux or UNIX, three users and groups are created for a root installation**

Instance Owner

The instance owner home directory is where the DB2 instance will be created

db2inst1

Fenced User

Used to run UDF's and stored procedures outside of the address space used by the DB2 database

db2fenc1

DB2 Administration Server User

The user ID is used to run the DB2 administration server on the system

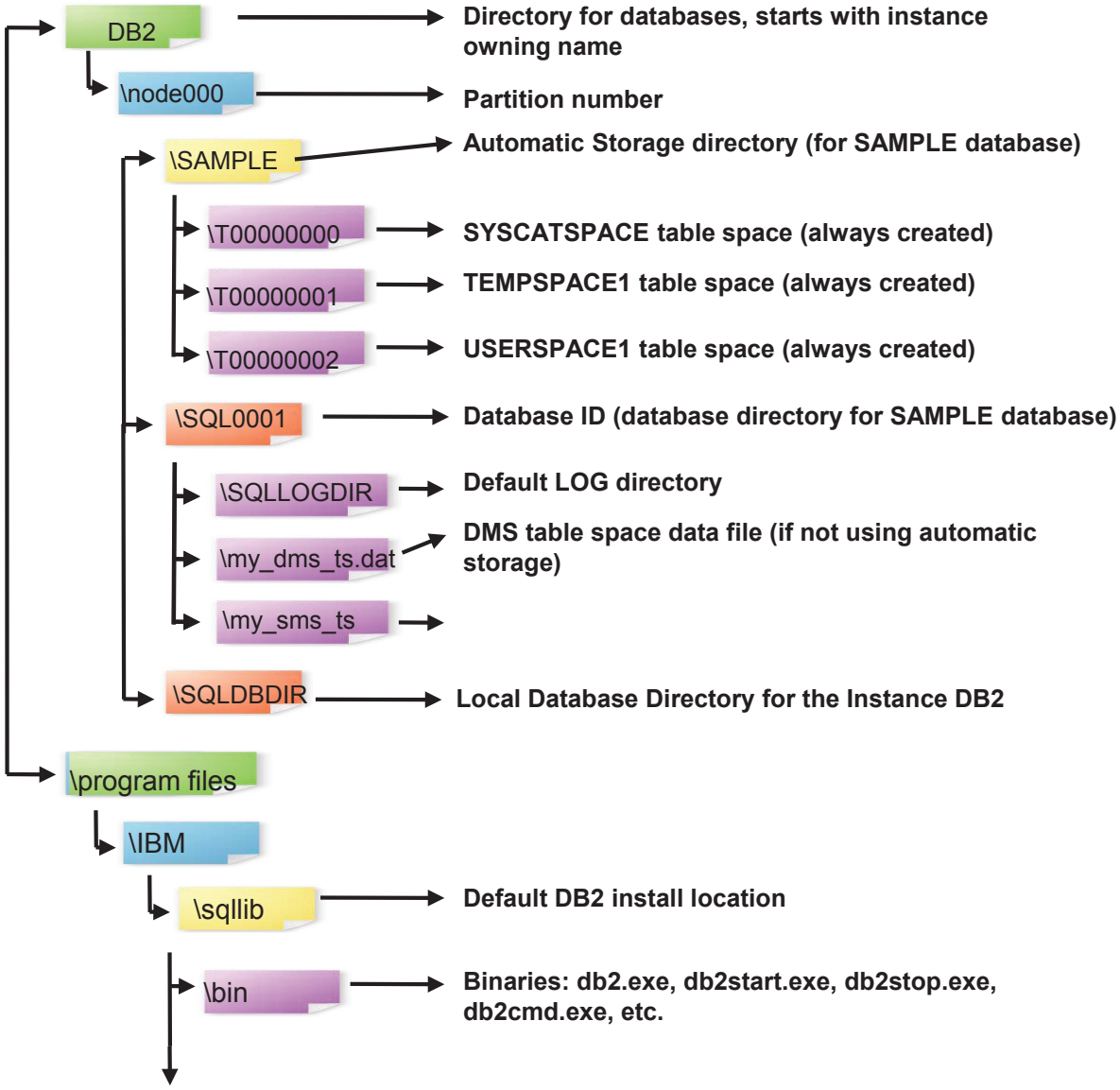
dasusr1

Administration Server has been deprecated in DB2 9.7!

- **On Windows, the following user accounts are required:**
  - Installation user account
    - Used to perform installation, normally a member of the Windows Administrators group
  - (Optional) one or more setup user accounts
    - DB2 instance user
    - DB2 Administration Server (DAS) user

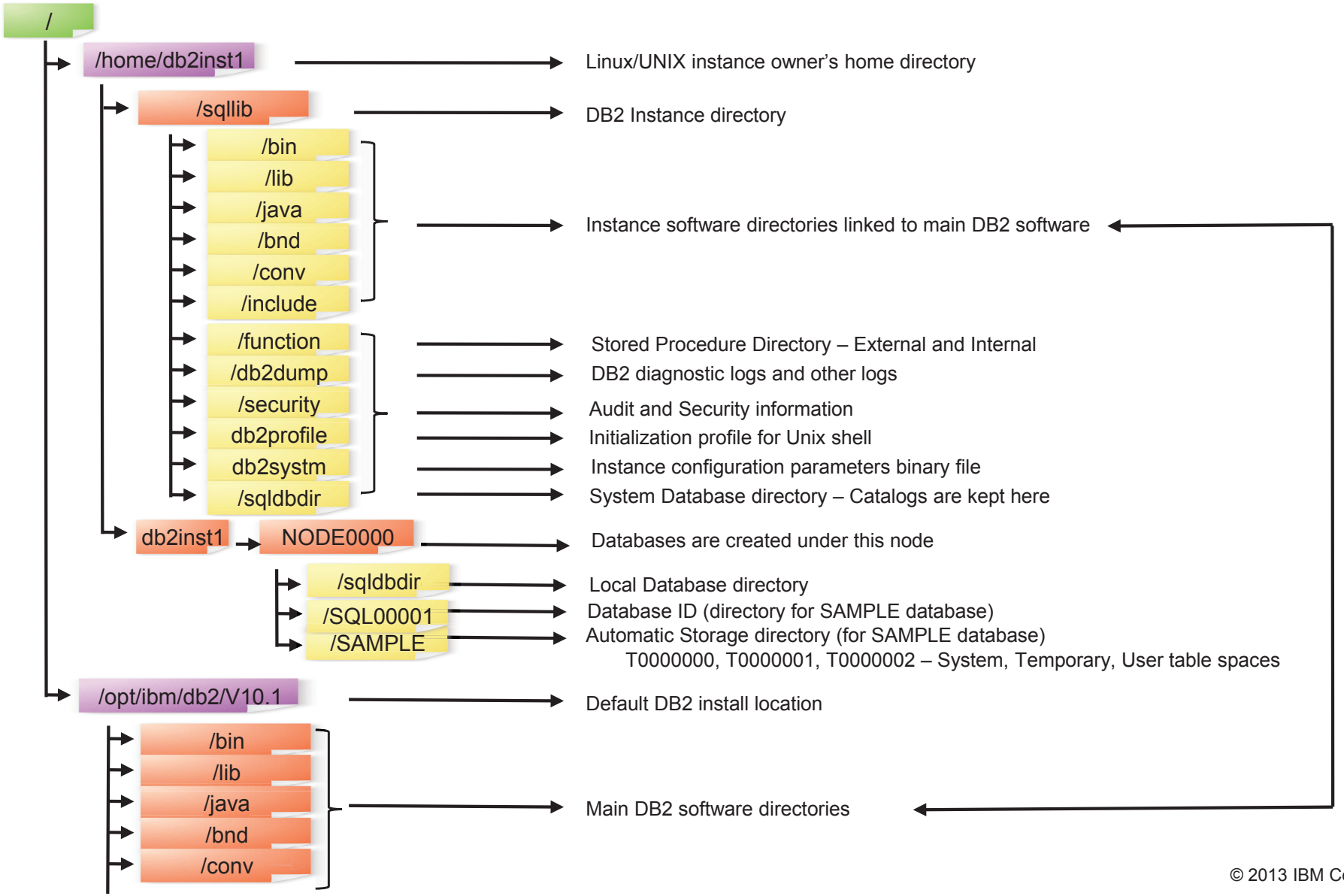
DB2 Installation – Directory Structure

- **Windows**



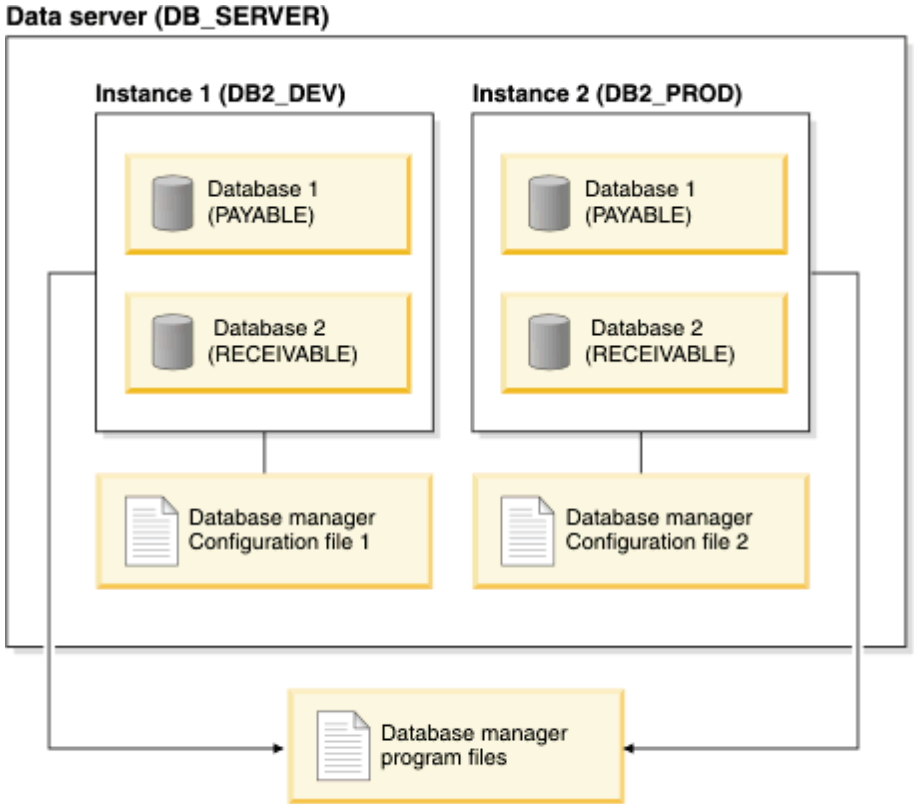
DB2 Installation – Directory Structure

- **Linux / UNIX (Automatic Storage)**



DB2 Environment – Data Server

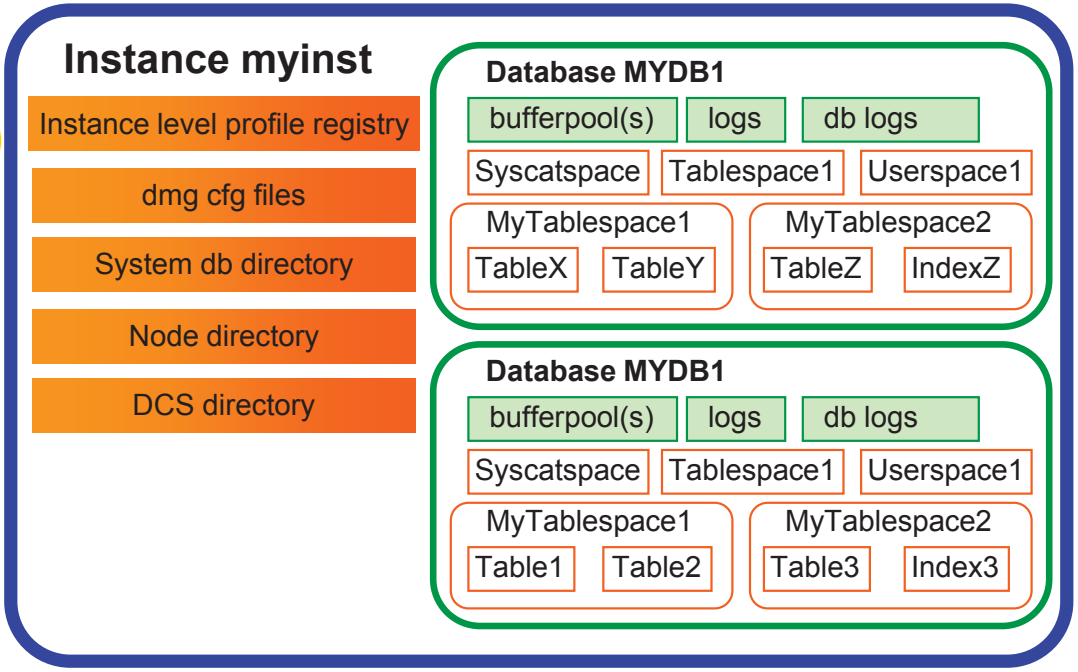
- A DB2 data server refers to a computer where the DB2 database engine is installed.



- A data server is able to host multiple instances and each instance can have and access more than one database at a time.

DB2 Environment – Instances

- **A DB2 instance is a logical database manager that serves as the access point to the databases structures**
- **All instances share the same executable binary files**
- **Each instance has**
  - its own configuration (dbm cfg)
  - multiple Engine Dispatchable Units (EDUs) shared among the databases in that instance



Command	Description	Example
db2start	Start the current instance	`db2start`
db2stop	Stop the current instance	`db2stop / db2stop force`
db2icrt	Create an instance	`db2icrt -u db2fenc1 db2inst1`
db2idrop	Drop an instance	`db2idrop db2inst1`
db2ilist	List all instances	`db2ilist`
db2iupgrade	Upgrades an instance to the current release It replaces the “db2imgr” command.	`db2iupgrade -u db2fenc1 db2inst1`
db2iupdt	Update an instance after installation of a fix pack	`db2iupdt -u db2fenc1 db2inst1`

DB and DBM Configurations

Description	Example
View Database Manager Settings	`db2 get dbm cfg show detail`
Change a Database Manager Setting	`db2 update dbm cfg using health_mon off`

Description	Example

View Database Settings	`db2 get db cfg for testdb`
	`db2 connect to testdb`
	`db2 get db cfg show detail`

Change a DB Setting	`db2 update db cfg using logprimary 10`

- **Examples of what can be changed using DB and DBM configuration**

Connection Management

- Define user authentication type
- Set communication protocols

Memory Tuning

- Set sort limits
- Set hash limits
- Set utility impact limits
- Share memory resources among the databases
- Instance memory

Monitoring

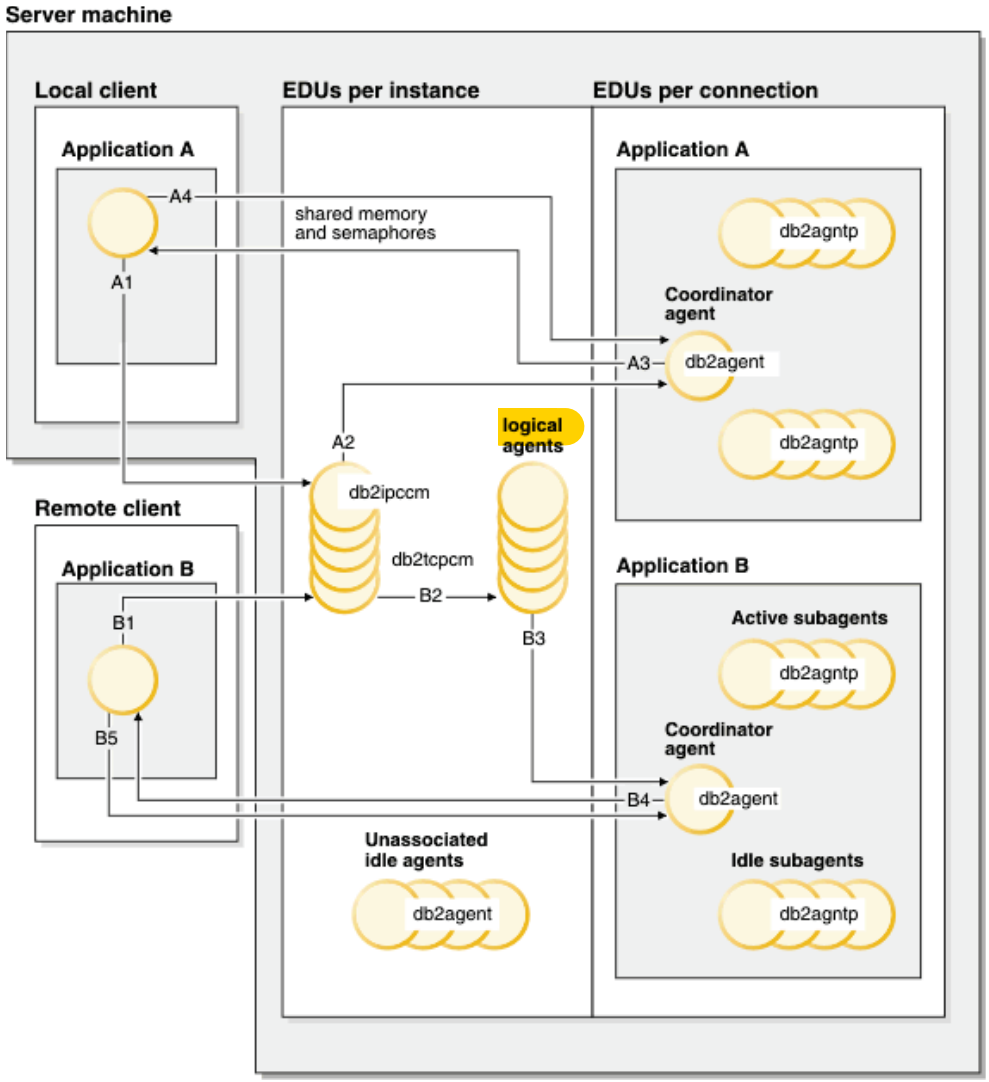
- Get database snapshots
- Check database health and performance

Instance Management

- Control instance services
- Enable federation
- Set diagnostic log level
- Authorization user groups

DB2 Process Model

- **Single process and multithreaded model**
  - System controller: db2sysc (UNIX) or db2syscs.exe (Windows)
  - Threads: Engine Dispatchable Units (EDU)
- **DB2 Agents (db2agent)**
  - Special type of EDU to handle application requests
  - The DB2 engine keeps a pool of agents available to service requests
  - An application is mapped to a coordinator agent
- **DB2 has firewall to protect databases and DBM**
  - Application runs on different address space to prevent application errors leading to corruption of DBM files or internal buffer

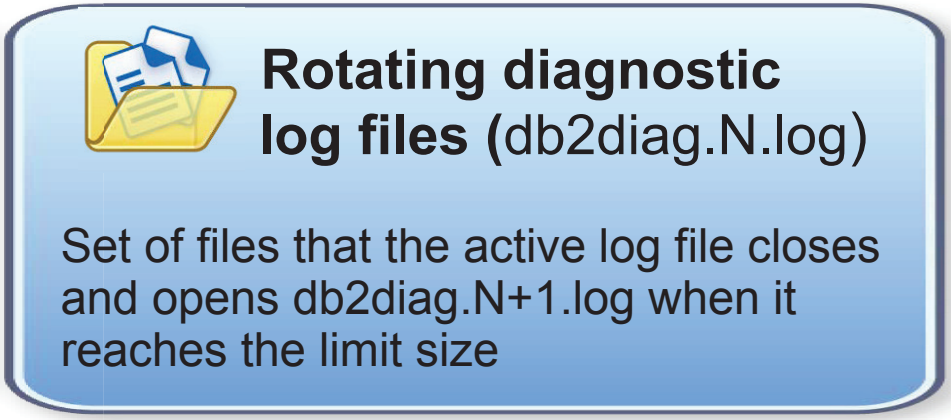


DB2 Diagnostic Log – db2diag.log

- Diagnostic and administration notification messages are logged into the DB2 diagnostic log files (db2diag). Primarily intended for troubleshooting purposes.
- Default location

  Linux/UNIX: `<$DB2INSTANCE_HOME>/sqllib/db2dump/DIAG<member#>`
  - E.g.: `/home/db2inst1/sqllib/db2dump/DIAG0000`
  - **Windows:** `<$DB2INSTPROF>\<instance_name>\DIAG<member#>`
    - E.g. (Windows 7): `C:\ProgramData\IBM\DB2\DB2COPY1\DB2\DIAG0000`

- **2 forms:**



- **Configuration parameters:**
  - `diagsize`: size of the log files for rotating log files form; 0 for single log file form
  - `diagpath`: Location of the log file(s)
  - `diaglevel`: Types of errors to be written to log
- **The db2diag tool serves to filter and format the volume of information available in the db2diag log files.**

DB2 Data Server Clients

IBM Data Server Driver Package

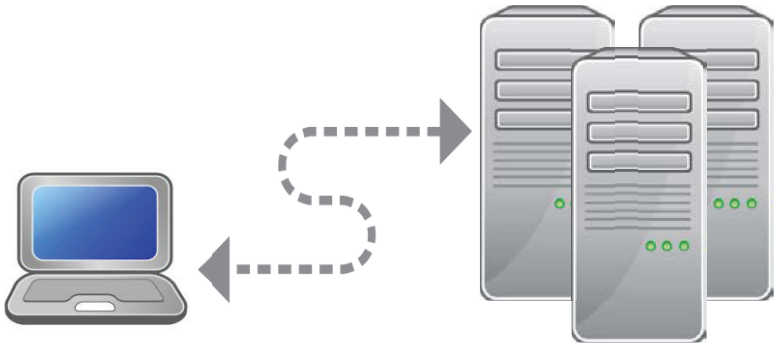
- Lightweight deployment solution that provides runtime support for applications using ODBC, CLI, .NET, OLE DB, PHP, Ruby, JDBC, or SQLJ

IBM Data Server Driver for JDBC and SQLJ

- Provides support for applications that use JDBC or SQLJ to access local or remote servers

IBM Data Server Driver for ODBC and CLI

- Runtime support for applications using ODBC API, or CLI API



IBM Data Server Runtime Client

- Include all the functionality from IBM Data Server Driver
- Has CLP but GUI tools are not included
- Support LDAP exploitation, TCP/IP and Named Pipe, cataloging

IBM Data Server Client

- Includes all the functionality of IBM Data Server Runtime Client
- Plus functionality for database administration, application development, and client/server configuration.
- Visual Studio tools, precompilers for various languages
- Samples and tutorials

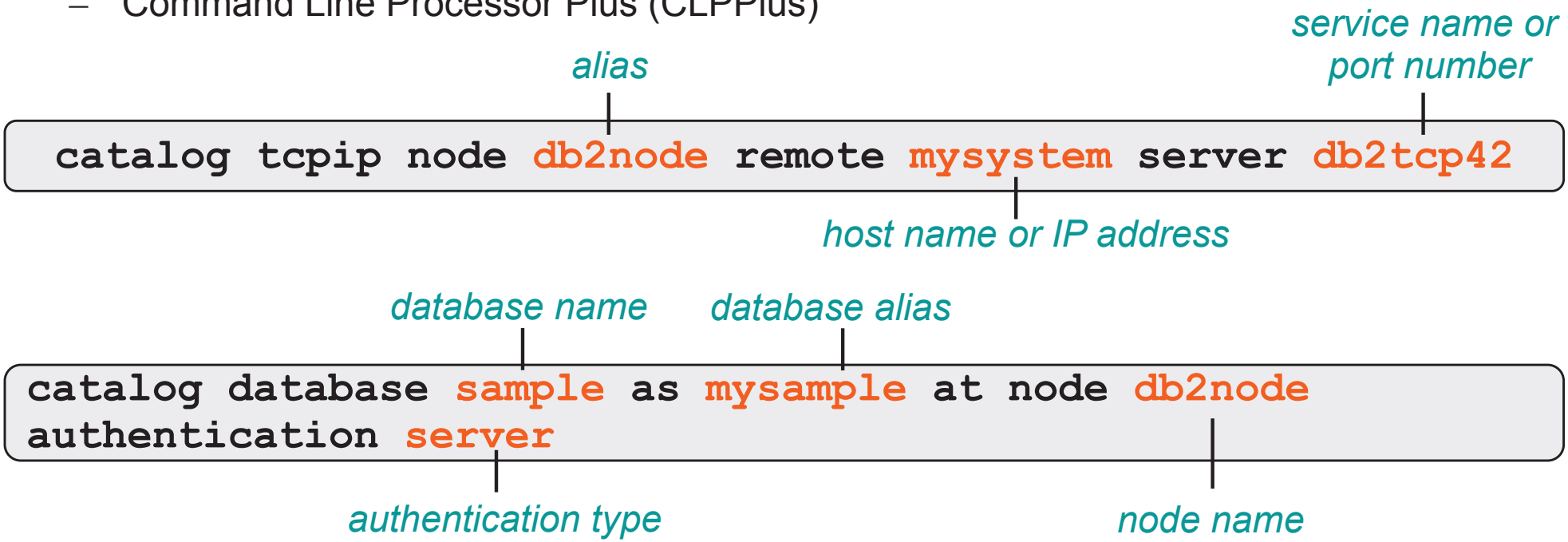
Cataloging – Setting up Communications

- **To use a remote database:**
  - Catalog the remote system (node)
  - Catalog the database within the remote node

```
db2 list [database | node] directory
```

can be used to find the locally catalogued DB or node

- **Tools for the job:**
  - Data Studio (graphical)
  - Command Line Processor (CLP)
  - Command Line Processor Plus (CLPPlus)



DB2 Databases

- **A database contains a set of objects used to store, manage, and access data according to the relational model of data.**
- **When creating a database, these tasks are performed:**
  - Setting up of all the system catalog tables that are needed by the database
  - Allocation of the database recovery log
  - Creation of the database configuration file and the default values are set
  - Binding of the database utilities to the database

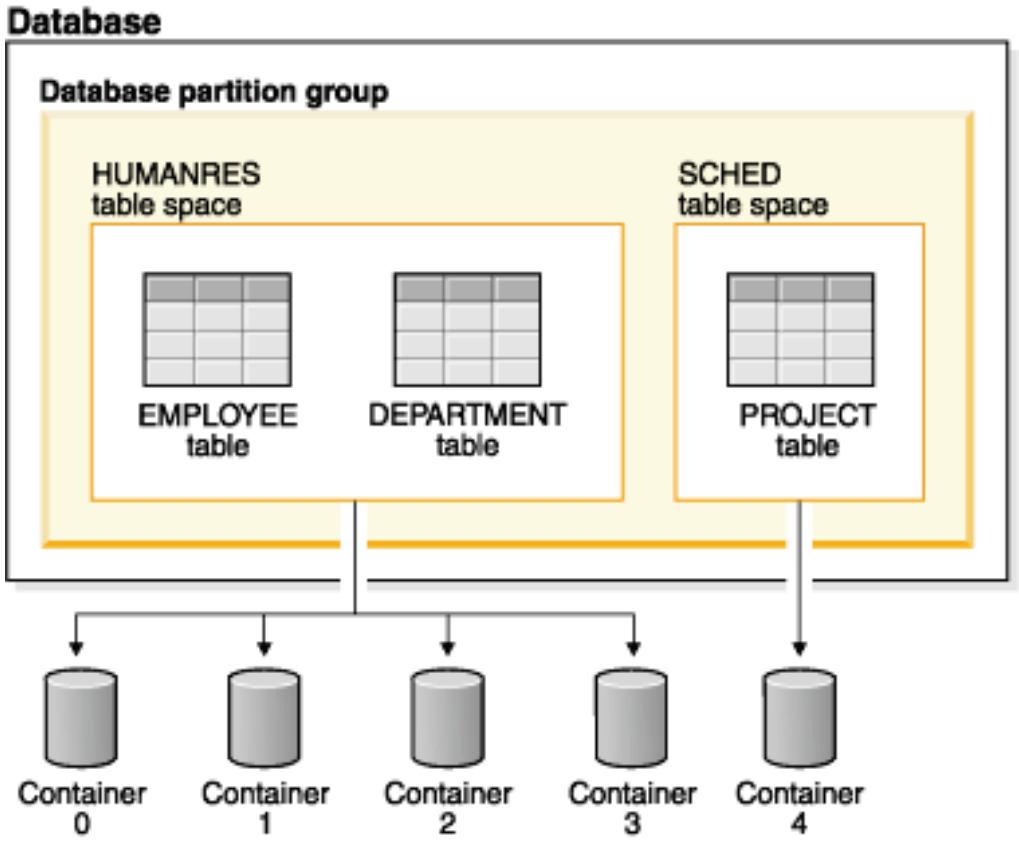
```
CREATE DATABASE prod1 ON /data1 DBPATH ON /dbfiles
```

storage path *database path*

- **By default:**
  - Configuration advisor is invoked to tune configuration parameters
  - A detailed deadlocks event monitor is created
  - The database uses UTF-8 (Unicode) code set
  - PUBLIC is granted these privileges: CREATETAB, BINDADD, CONNECT, IMPLICIT_SCHEMA, and SELECT on the system catalog views

Table Spaces

- **Structure to organize data into logical storage groupings**
- **All tables, indexes, large objects and long data are stored in a table space**
- **A table space is consisted of one or more containers**
  - Containers can be files, directories or raw devices
- **It is associated to a specific buffer pool**



Types of Table Spaces

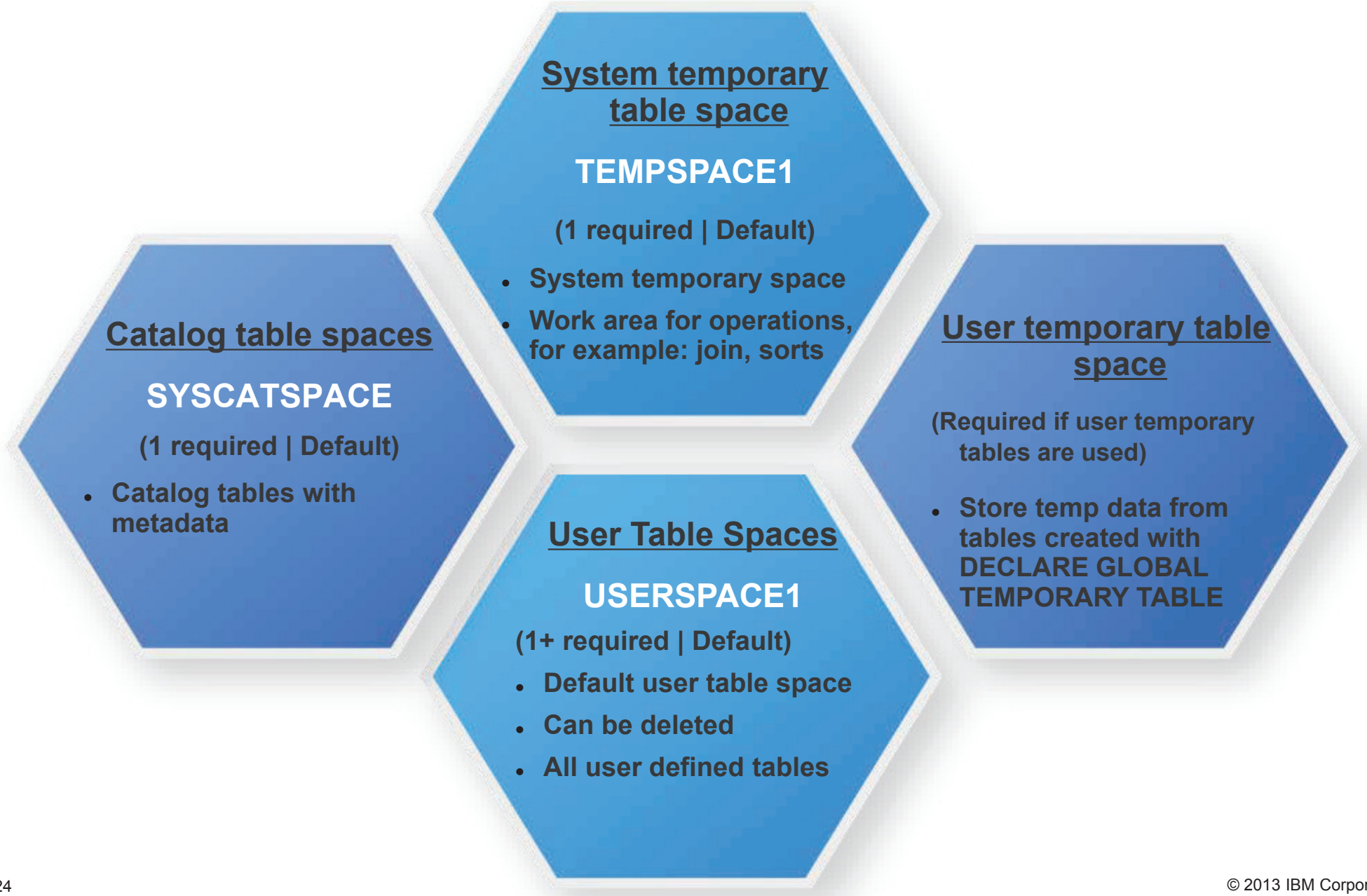


Table Space Management

Deprecated in DB2 10.1 for user permanent table spaces

- **System Managed Spaces (SMS)**
  - Data stored in files representing data objects
  - Space is allocated on demand
  - Access to data controlled using standard I/O functions of the OS
  - ✔ Low maintenance
  - ✖ Performance is not optimal

```
CREATE TABLESPACE tbsp1 MANAGED BY SYSTEM
USING ('d:\acc_tbsp', 'e:\acc_tbsp', 'f:\acc_tbsp')
```

- **Database Managed Spaces (DMS)**
  - Data stored in files or on raw devices
  - Storage space pre-allocated in file system, typically contiguous physically
  - ✔ Ideal for performance-sensitive applications
  - ✖ Increased maintenance and monitoring

```
CREATE TABLESPACE tbsp2
PAGESIZE 8K MANAGED BY DATABASE
USING (FILE ' /storage/dms1' 10 M) AUTORESIZE YES
```

Table Space Management

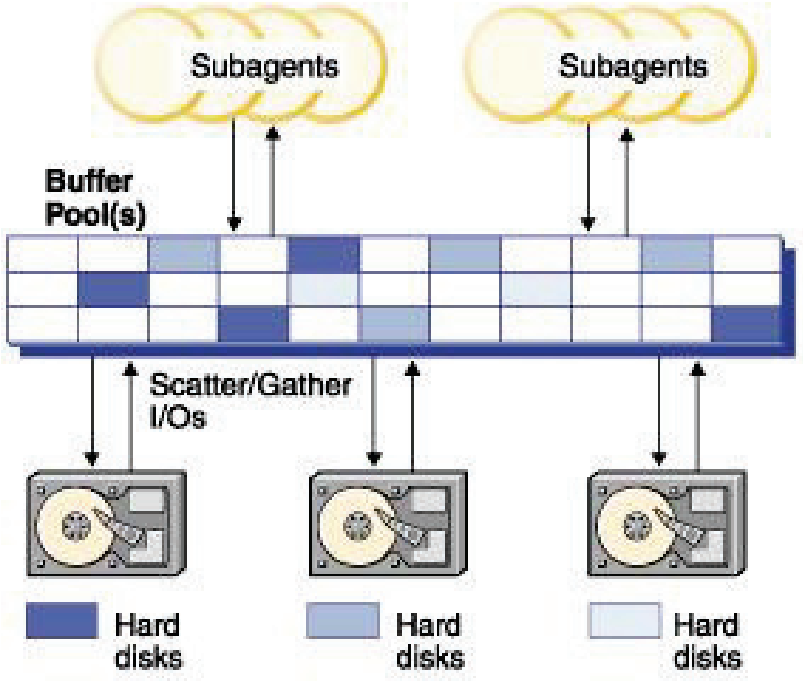
- **Automatic Storage Table Space**
  - DBM creates and extends containers as needed up the limits imposed by the storage paths associated with the database
  - Automatically handles resizing table spaces
  - Creates a DMS table space for regular/large table spaces
  - Creates a SMS table space for user or system temporary table spaces
  - New databases and table spaces use Automatic Storage by default

Parameter Deprecated in DB2 10

```
CREATE DATABASE mydb AUTOMATIC STORAGE YES

CONNECT TO mydb

CREATE TABLESPACE tbsp1 MANAGED BY AUTOMATIC STORAGE
```

Buffer Pools

- **Area of main memory used to cache table and index data**
- **Each database must have at least one buffer pool**
  - By default IBMDEFAULTBP is used
  - Buffer pools can be created, dropped or altered
  - SYSCAT.BUFFERPOOLS catalog view accesses the information for the buffer pools defined in the database
- **Every table space associates a specific buffer pool of the same page size**
  - Match buffer pool size with purpose of table to increase hit ratio
- **Self-Tuning Memory Manager (STMM) available**




```
CREATE BUFFERPOOL bp4k PAGESIZE 4K
CREATE TABLESPACE tbsp1 PAGESIZE 4K BUFFERPOOL bp4k
```

DB2 Security

- **Authentication vs. Authorization**
  - Authentication
    - Access to the DB2 database system
    - Identification of a user with a password
  - Authorization
    - Access within DB2 database system
    - Authority and privileges to perform database operations, access data objects
- **Explicit vs. Implicit vs. Indirect Access**

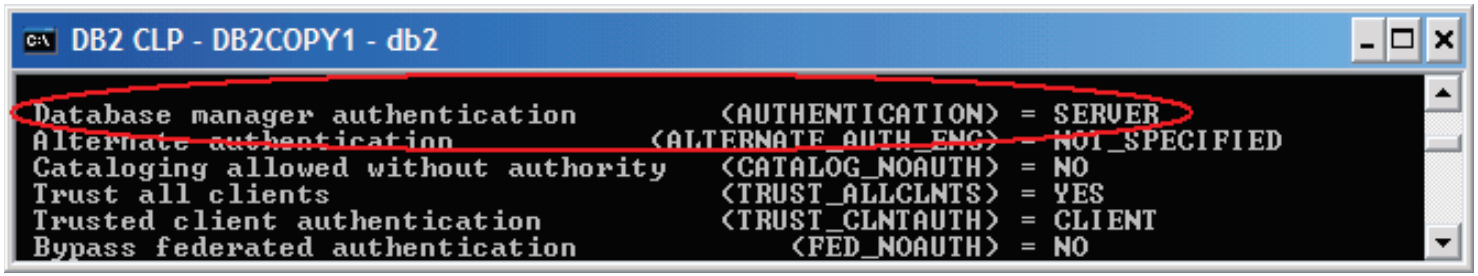
Explicit	Implicit	Indirect
• User • Group • Role	• When a database or database object is created	• Inherited through execution of packaged code

- **More Security Features**
  - Data Encryption
  - Fine grained security: Label Based Access Control, **Row and Column Access Control (New in DB2 10)** ⭐
  - Auditing

Configuration of Authentication on DB2 Server

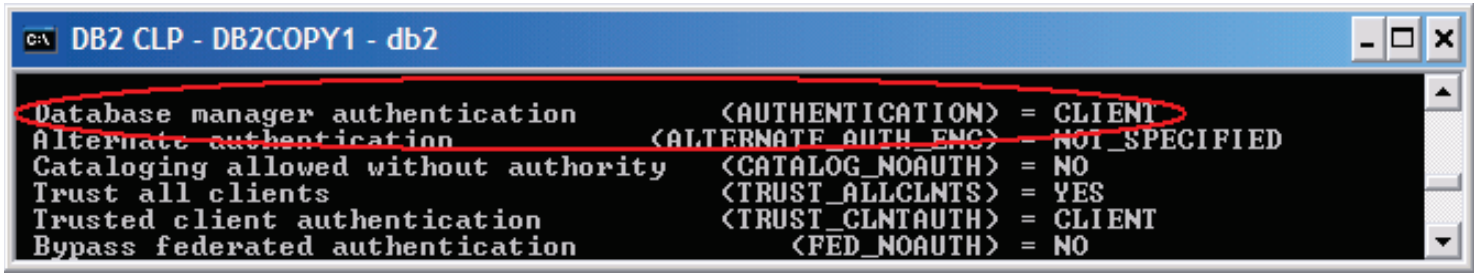
- **Authentication type is defined in the Database Manager configuration file (DBM CFG)**

```
db2 "GET DBM CFG"
```



- **To configure how and where DB2 authenticates users, set the authentication parameter at the DB2 server**

```
db2 "UPDATE DBM CFG USING AUTHENTICATION CLIENT"
```



DB2 Sample Database

- To create the sample database populated with both relational data and XML data
- Verify the database creation by simply connecting and querying the data

```
db2sampl -sql -xml

db2 connect to sample
```

- For remote databases:

```
db2 catalog database sample as db_sample
at node mynode1

db2 connect to db_sample
```

Application Development and Language Support

- **Programming Languages Supported**
  - ADO.NET, OLE DB, DB2CI, CLI and ODBC, Embedded SQL
  - Java (JDBC, SQLJ)
  - Ruby on Rails, Perl, PHP, Python, SQLAlchemy and Django framework
  - RDF – Resource Description Framework (**NEW in DB2 10**) 
- **SQL – Structure Query Language**
- **pureXML – storage and manipulation of XML documents**
  - Supports XQuery and SQL/XML functions
- **SQL Procedural Language (SQL PL)**
  - It can be used to implement procedural logic in SQL statements
- **PL/SQL – Procedural Language/Structured Query Language**
  - Reduces the complexity of enabling existing PL/SQL solutions so that they will work with the DB2 data server

Interfaces – Tools, Commands, APIs

- **Built-in administrative routines and views**
  - Programmatic interface to administer databases through SQL
- **Command Line tools**
  - CLP commands (e.g.: list applications)
  - CLPPlus commands (e.g.: connect, clear)
  - System commands (e.g.: db2start, db2diag)
- **DB2 Administrative APIs**
  - APIs providing functions for performing administrative tasks on instances, databases, databases objects and data
  - E.g.: SQLECREA API - Create database

IBM Data Studio

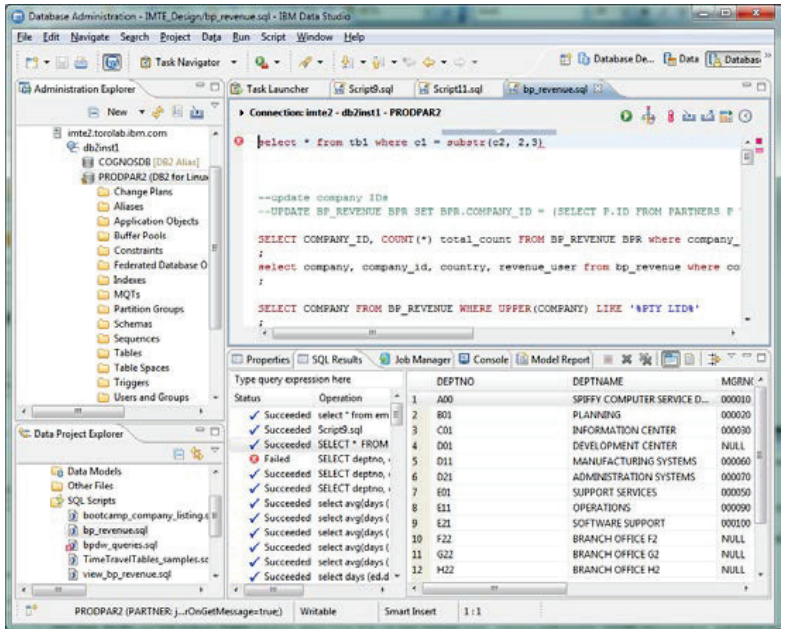
Interfaces – Tools, Commands, APIs

- **Control Center tools have been discontinued**

  Activity Monitor, Command Editor, Configuration Assistant,
  Control Center and associated wizards and advisors,
  Control Center plug-in extensions, Event Analyzer, Health Center,
  Indoubt Transaction Monitor, Journal, License Center, Memory Visualizer,
  Query Patroller Center, Satellite Administration Center, Task Center
  User interface to access Spatial Extender functionality
  User interface to Visual Explain

- **Replication Center is still available**

- **Data management and application development tools**
  - IBM Data Studio
  - IBM Data Studio web console
  - IBM InfoSphere® Optim™ tools

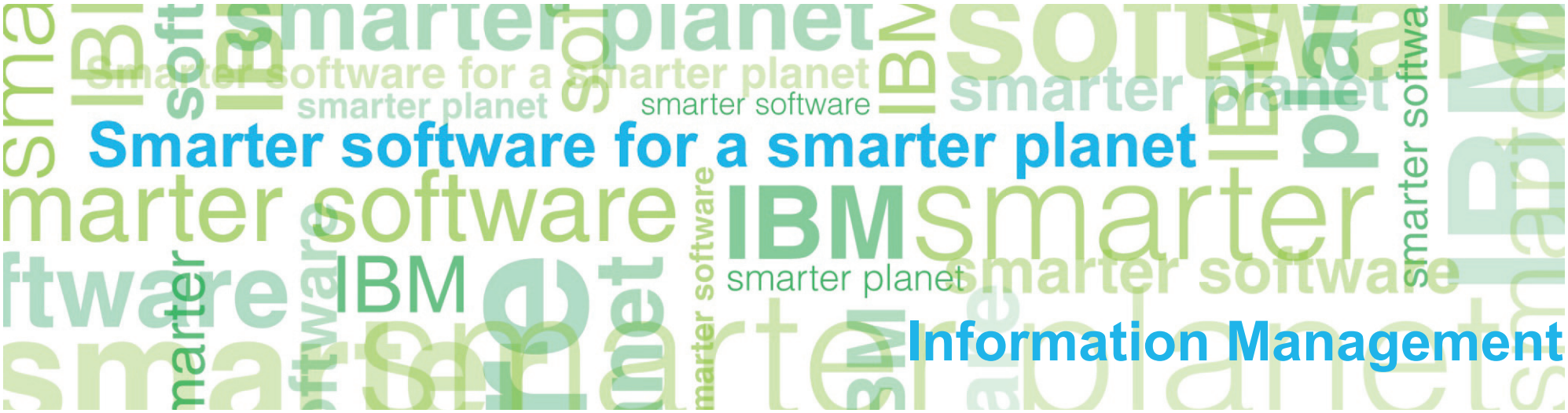


IBM Data Studio 3.1

Summary

- **DB2 for LUW is a multi-platform hybrid data server**
- **Several editions to fit your business needs**
- **Flexible licensing options**
- **Optimized architecture for high performance**
- **Practical troubleshooting utilities**
- **Comprehensive security structure**
- **Full suite of tools for easy data management and application development**

DB2 Storage Optimization



Current IT Storage Spending

IDC
Analyze the Future

"Reducing storage-related costs is again a top priority in 2011, as many organizations continue to struggle with taming data growth"

Gartner

"Data growth is the biggest data center hardware infrastructure challenge for large enterprises"

"…power consumption by data centers doubled from 2000 to 2005 …in 2012, [the census] did predict growth in the Eastern United States of some 22 percent"

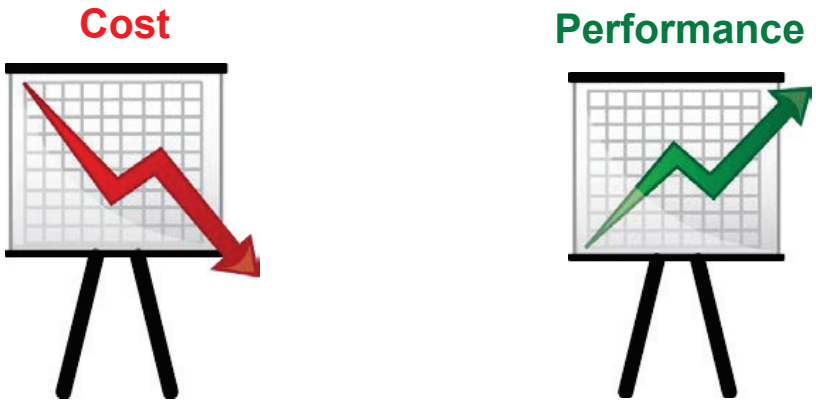
The New York Times

Agenda

- **Compression Overview**
- **Storage Optimization in DB2**
  - Value Compression
  - Row Compression
  - Temporary Table Compression
  - XML Compression
  - Index Compression
  - Backup Compression
  - Compression in BLU
- **Compression Benefits**
- **Client Success**

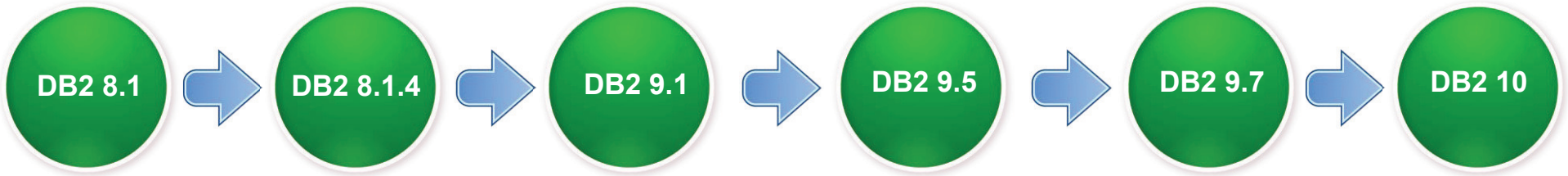
Compression Overview

- **Requires less storage space and memory**
  - Usually the most expensive component in a database solution
- **Helps manage storage growth**
  - Saves on power and cooling
- **Helps improve database performance**
  - Fewer I/O operations required
- **Reduce administration costs and increase productivity of the IT team**
- **Facilitates lower cost of ownership**



Storage Optimization in DB2

- **Provides storage compression services to optimize the performance and footprint of your data**
- **Basic compression features included in several editions**
- **Advanced compression features bundled as DB2 Storage Optimization Feature, included in Advanced Enterprise Server edition**
  - Available for DB2 Enterprise Server edition with a PVU license



- Value Compression
- Backup Compression
- Row Compression*
- Automatic Dictionary Creation (ADC)*
- XML compression*
- Temporary table compression*
- Index compression*
- LOB inlining
- Adaptive compression*

** Included in DB2 Storage Optimization Feature*

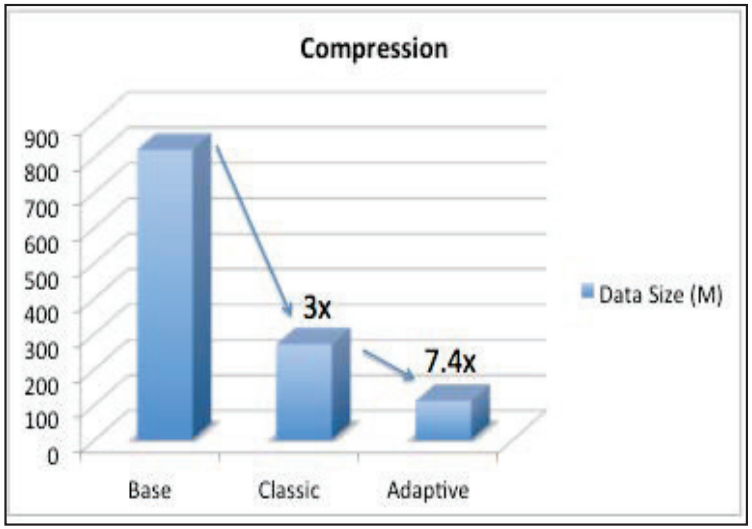
Row Compression

- **Also known as deep compression**
- **Uses a dictionary-based compression algorithm to replace recurring strings with shorter symbols within rows**
- **Continuous enhancement since it was introduced in DB2 9.1**

- Row Compression*
- Automatic Dictionary Creation (ADC)*
- XML compression*
- Temporary table compression*
- Index compression*
- LOB inlining
- Adaptive compression*

- **Two types available:**
  - **Classic** (static) row compression
  - **Adaptive** row compression
    - An enhancement to classic row compression to provide extra storage savings

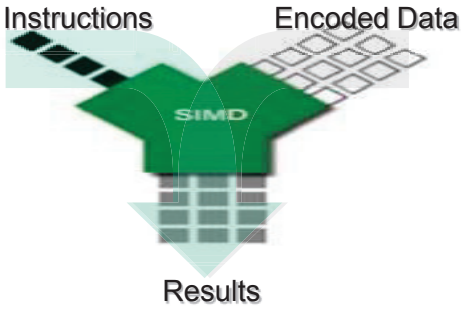
New in DB2 10



- **Included in DB2 Storage Optimization Feature**

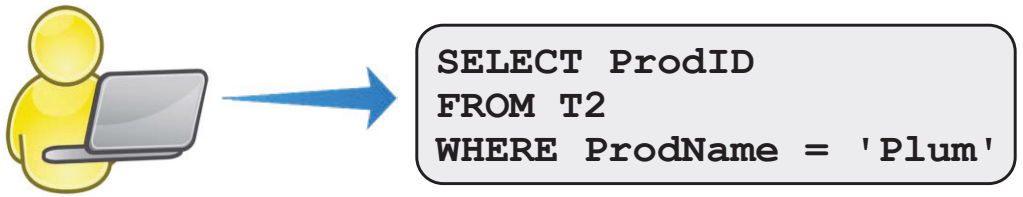
BLU uses Multiple Compression Techniques

- Approximate Huffman-Encoding ("frequency-based compression"), prefix compression, and offset compression
- For column-organized tables, COMPRESSION cannot be enabled/disabled
- Frequency-based compression: Most common values use fewest bits
  - Exploiting skew in data distribution improves compression ratio
  - Very effective since all values in a column have the same data type
  - Maps entire values to dictionary codes
- Actionable compression :- Order-preserving encoding allows predicates to be evaluated on compressed data
- SIMD (Single Instruction Multiple Data) parallelism used for fast predicate evaluation on multiple compressed values
- Avoiding decompression during predicate evaluation provides significant query performance gains



Understanding the Basics

The DB2 Storage - Example



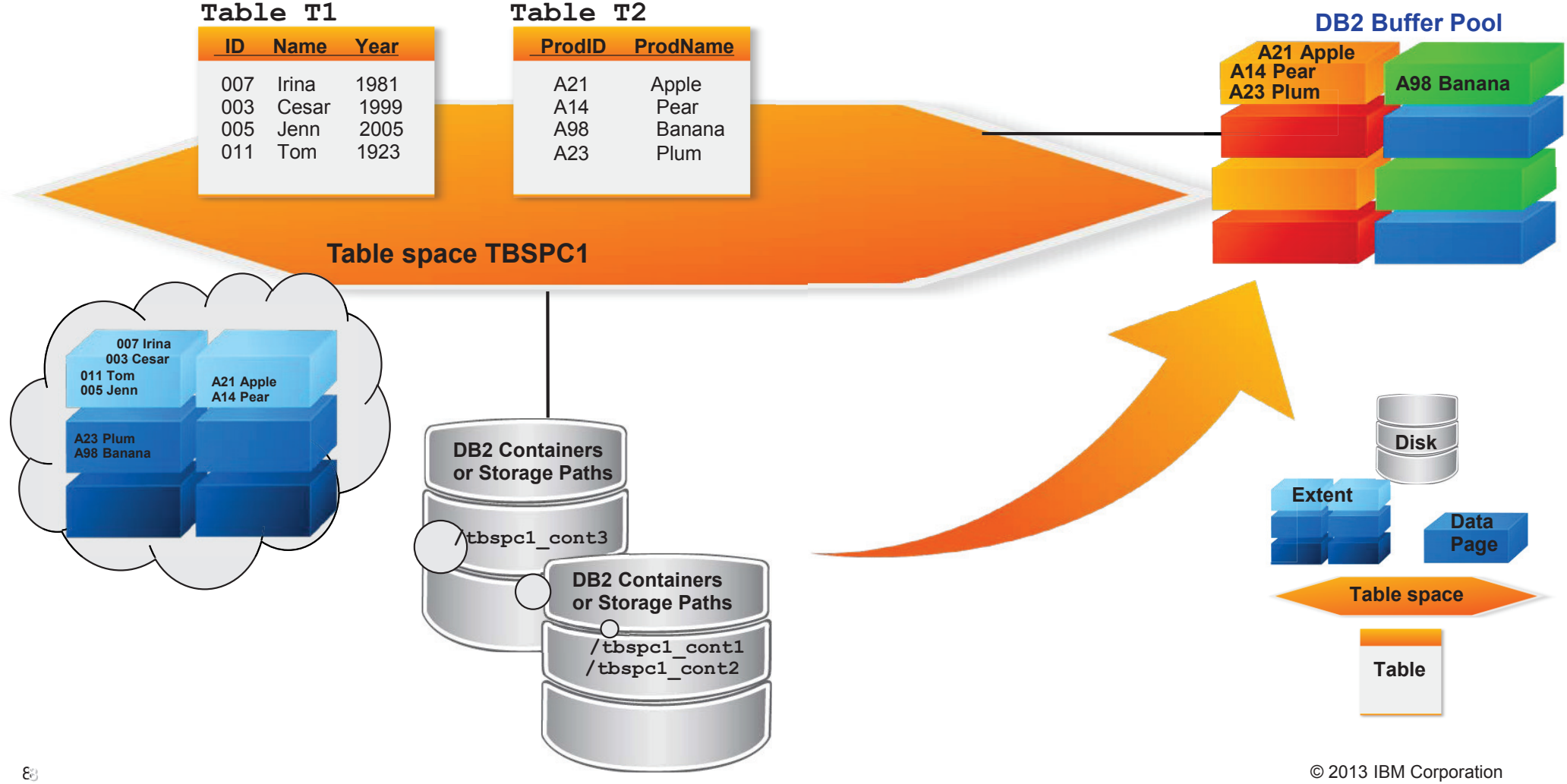
```
SELECT ProdID
FROM T2
WHERE ProdName = 'Plum'
```

Table T1

ID	Name	Year
007	Irina	1981
003	Cesar	1999
005	Jenn	2005
011	Tom	1923

Table T2

ProdID	ProdName
A21	Apple
A14	Pear
A98	Banana
A23	Plum

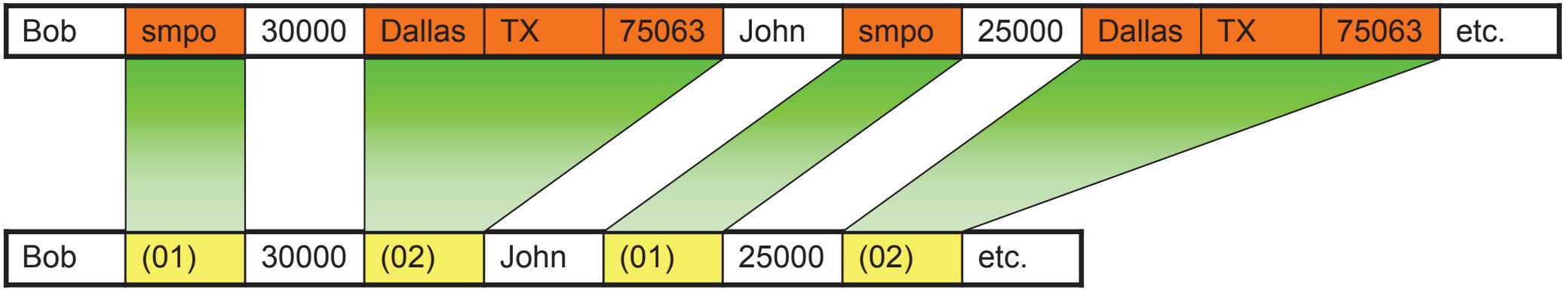


Row Compression – Classic

- **Also referred to as static row compression**
- **Uses a table-level compression dictionary (1 dictionary per table) to compress data by row, across multiple columns**
- **Dictionary is used to map repeated byte patterns to smaller symbols. These smaller symbols replace long patterns in table rows.**
- **After dictionary is created, data is compressed as it is inserted/updated in the table.**
  - DB2 automatically creates the dictionary when enough the table has enough data for sampling

Name	Dept	Salary	City	ST	ZIP
Bob	smpo	30000	Dallas	TX	75063
John	smpo	25000	Dallas	TX	75063

Dictionary	
(01)	smpo
(02)	Dallas, TX, 75063

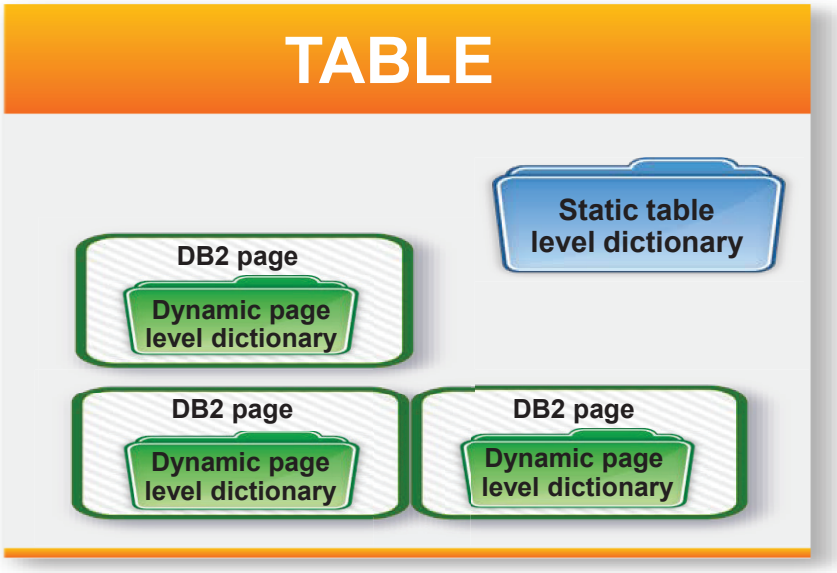


Why Choose Adaptive Row Compression

- **Classic row compression works very well**
  - Very fast and robust
  - Not sensitive to data clustering and ordering
- **But it has limitations**
  - Dictionary requires classic table reorganization to refresh
  - Not sensitive to data clustering and ordering
  - Dictionary capacity limits compression ratios for some large tables
- **DB2 10 introduces Adaptive Compression**
  - An **enhancement to** classic row compression
  - **Industry-leading compression** technology for row stores
- **Reduces TCO, providing extra compression savings:**
  - Classic row compression: Typically saves **~40%-75%**
  - Adaptive row compression: Typically saves **~75%-85%**
  - Adaptive typically saves 30% over classic row compression

Row Compression – Adaptive

- Adaptive Compression is an enhancement to the Classic Row Compression
  - Compress rows by using a combination of two types of algorithms
    - Globally recurring byte sequences
    - Locally recurring byte sequences



- Benefits
  - Page level dictionaries adapt to data skew over a period of time
  - No REORGs required to maintain high compression ratio
  - Less disk space for data and logs
  - Reduced I/O. Fewer pages to process

- **Better compression ratios than Classic Row Compression**
- **Over time reduces need of a REORG as page-level dictionaries adapt to data skew over time (i.e. maintain compression rates)**
- **Default row compression method in DB2 10**

How Does Adaptive Compression Work?

- **Step 1: Compression with static table level dictionary**

Christine	Haas	(408) 463-1234	555 Bailey Avenue	San Jose	California	95141
John	Thompson	(408) 463-5678	555 Bailey Avenue	San Jose	California	95141
Jose	Fernandez	(408) 463-1357	555 Bailey Avenue	San Jose	California	95141
Margaret	Miller	(408) 463-2468	555 Bailey Avenue	San Jose	California	95141
Bruce	Kwan	(408) 956-9876	4400 North 1st Street	San Jose	California	95134
James	Geyer	(408) 956-5432	4400 North 1st Street	San Jose	California	95134
Linda	Hernandez	(408) 956-9753	4400 North 1st Street	San Jose	California	95134
Theodore	Mills	(408) 927-8642	650 Harry Road	San Jose	California	95134
Susan	Stern	(408) 927-9630	650 Harry Road	San Jose	California	95134
James	Polaski	(415) 545-1423	425 Market Street	San Francisco	California	94105
John	Miller	(415) 545-5867	425 Market Street	San Francisco	California	94105
James	Walker	(415) 545-4132	425 Market Street	San Francisco	California	94105
Elizabeth	Brown	(415) 545-8576	425 Market Street	San Francisco	California	94105
Sarah	Johnson	(415) 545-1928	425 Market Street	San Francisco	California	94105

[1]	California 9
[2]	San
[3]	Jose
[4]	Francisco
[5]	Avenue
[6]	Street
[7]	Road

Compression with global table static dictionary

Christine	Haas	(408) 463-1234	555 Bailey [5]	[2][3]	[1]	95141
John	Thompson	(408) 463-5678	555 Bailey [5]	[2][3]	[1]	95141
[3]	Fernandez	(408) 463-1357	555 Bailey [5]	[2][3]	[1]	95141
Margaret	Schneider	(408) 463-2468	555 Bailey [5]	[2][3]	[1]	95141
Bruce	Kwan	(408) 956-9876	4400 North 1st [6]	[2][3]	[1]	95134
James	Geyer	(408) 956-5432	4400 North 1st [6]	[2][3]	[1]	95134
Linda	Hernandez	(408) 956-9753	4400 North 1st [6]	[2][3]	[1]	95134
Theodore	Mills	(408) 927-8642	650 Harry [7]	[2][3]	[1]	95134
Susan	Stern	(408) 927-9630	650 Harry [7]	[2][3]	[1]	95134
James	Polaski	(415) 545-1423	425 Market [6]	[2][4]	[1]	94105
John	Miller	(415) 545-5867	425 Market [6]	[2][4]	[1]	94105
James	Walker	(415) 545-4132	425 Market [6]	[2][4]	[1]	94105
Elizabeth	Miller	(415) 545-8576	425 Market [6]	[2][4]	[1]	94105
Sarah	Johnson	(415) 545-1928	425 Market [6]	[2][4]	[1]	94105

- Table level compression symbol dictionary containing globally recurring patterns
- Table-level dictionary can only be rebuilt during Classic Table REORG
  - Involves re-compressing all data

How Does Adaptive Compression Work?

- **Step 2: Compression w/ Page-Level Dictionaries**

Data page

Christine	Haas	(408) 463-1234	555 Bailey [5]	[2][3]	[1]	5141
John	Thompson	(408) 463-5678	555 Bailey [5]	[2][3]	[1]	5141
Ellen	Fernandez	(408) 463-1357	555 Bailey [5]	[2][3]	[1]	5141
Margaret	Schneider	(408) 463-2468	555 Bailey [5]	[2][3]	[1]	5141
Bruce	Kwan	(408) 956-9876	4400 North 1st [6]	[2][3]	[1]	5134
James	Geyer	(408) 956-5432	4400 North 1st [6]	[2][3]	[1]	5134
Linda	Hernandez	(408) 956-9753	4400 North 1st [6]	[2][3]	[1]	5134

Data page

Theodore	Mills	(408) 927-8642	650 Harry [7]	[2][3]	[1]	5134
Susan	Stern	(408) 927-9630	650 Harry [7]	[2][3]	[1]	5134
James	Polaski	(415) 545-1423	425 Market [6]	[2][4]	[1]	4105
John	Miller	(415) 545-5867	425 Market [6]	[2][4]	[1]	4105
James	Walker	(415) 545-4132	425 Market [6]	[2][4]	[1]	4105
Elizabeth	Miller	(415) 545-8576	425 Market [6]	[2][4]	[1]	4105
Sarah	Johnson	(415) 545-1928	425 Market [6]	[2][4]	[1]	4105

Christine	Haas	(2)1234	(4)
John	Thompson	(2)5678	(4)
Ellen	F(1)	(2)1357	(4)
Margaret	Schneider	(2)2468	(4)
Bruce	Kwan	(3)9876	(5)
James	Geyer	(3)5432	(5)
Linda	H(1)	(3)9753	(5)

(1)	ernandez
(2)	(408) 463-
(3)	(408) 956-
(4)	555 Bailey [5] [2][3] [1] 5141
(5)	4400 North 1st [6] [2][3] [1] 5134

Page level dictionary

Theodore	(3)s	(4)8642	(6)
Susan	Stern	(4)9630	(6)
(1)	Polaski	(5)1423	(7)
(2)	(3)er	(5)5867	(7)
(1)	Walker	(5)4132	(7)
Elizabeth	(3)er	(5)8576	(7)
Sarah	(2)son	(5)1928	(7)

(1)	James
(2)	John
(3)	Mill
(4)	(408) 927-
(5)	(415) 545-
(6)	650 Harry [7] [2][3] [1] 5134
(7)	425 Market [6] [2][4] [1] 4105

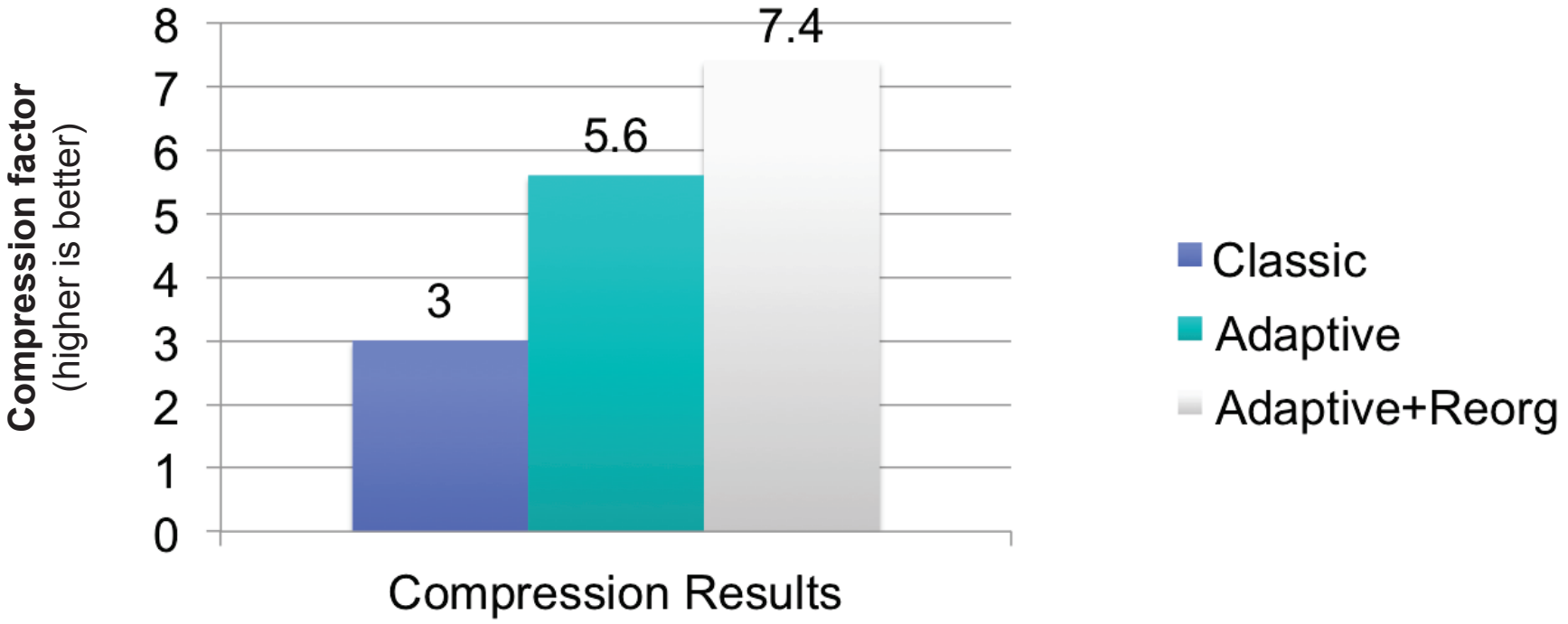
Page level dictionary

- Page-level compression dictionaries contain locally frequent patterns
- Page-level compression dictionary building & rebuilding is fully automatic
- Algorithm optimized for compressing data already compressed by table-level dictionary
- Page-level compression dictionaries are stored as special records in each page

Data Warehouse Compression Results

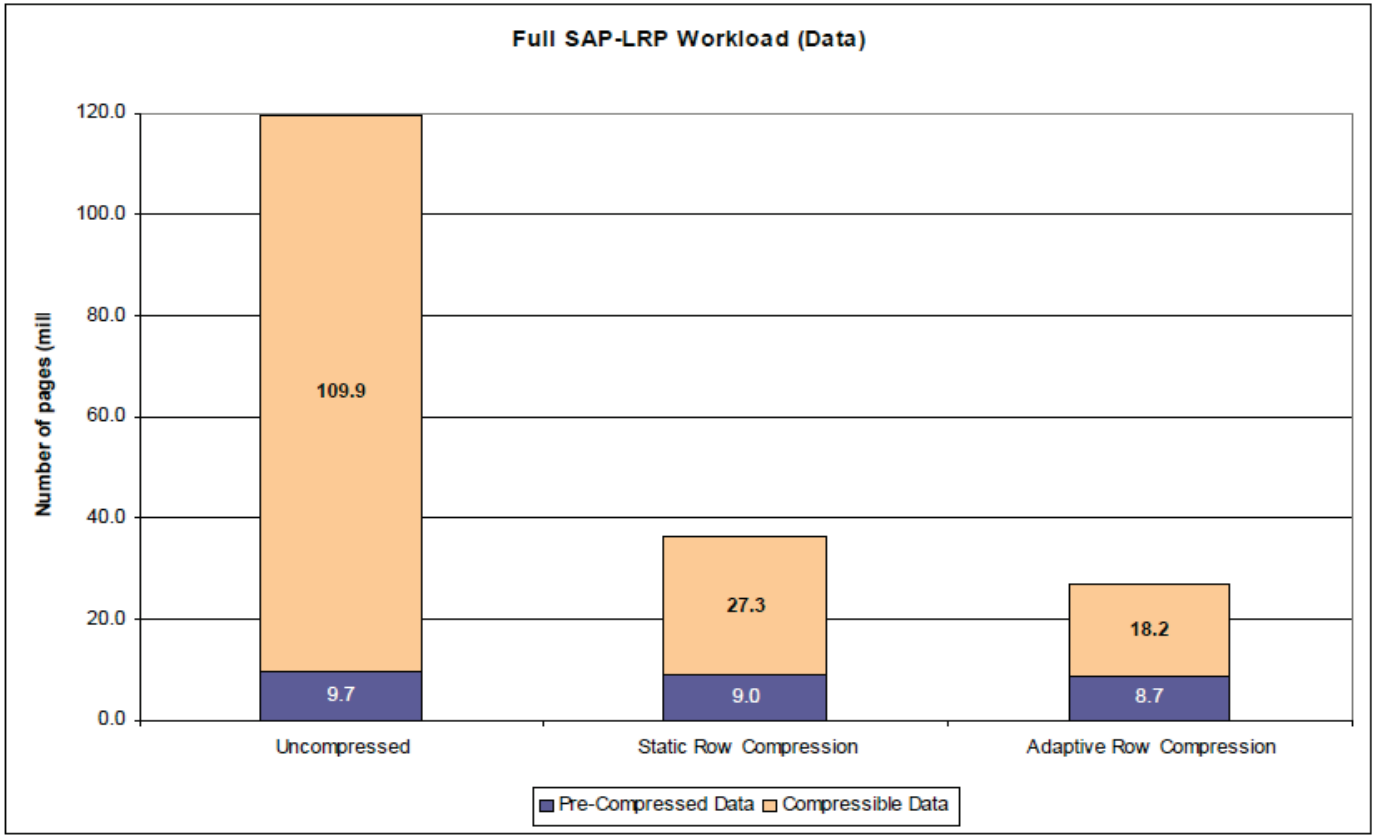
230GB raw size - Most of the data in a single table

- **Graph – Storage Savings**
- **Increase in savings by Adaptive Compression**
  - 3x Compression with Static Compression using reorg
  - 5.6x Compression with Automatic dictionary creation and Adaptive Compression
  - 7.4x Compression with Adaptive Compression and full reorg



Additional Storage Savings

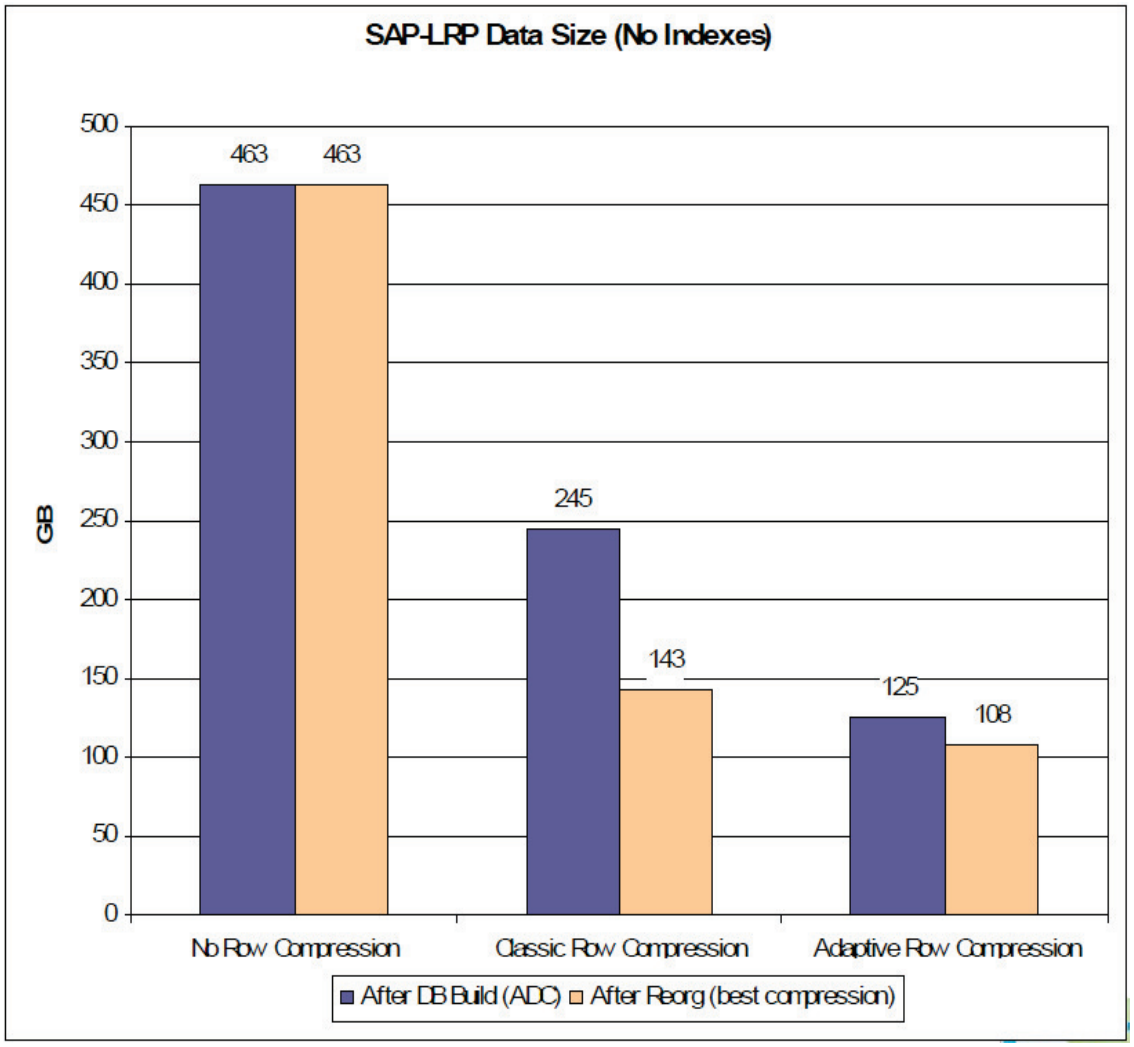
- SAP-LRP workload, ~2TB data size before compression



- Considering the 1425 largest tables (~99.5% of the total data), after Classic Table Reorg
- 33% savings for normal tables, compared to Classic Row Compression
- Almost no storage savings for tables with pre-compressed data (SAP cluster tables, certain tables with LOB fields)

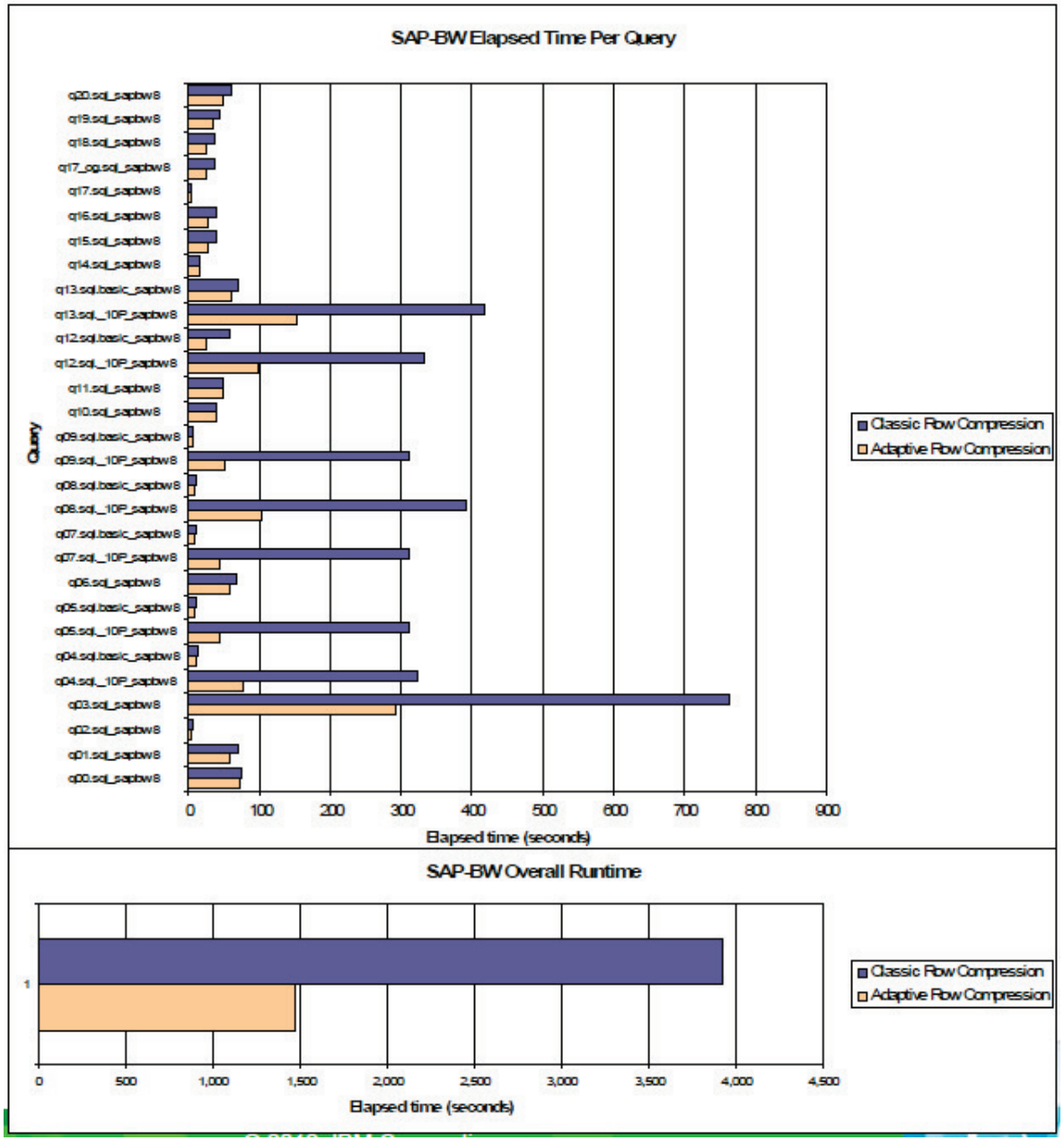
Reorg Avoidance – ADC vs. Classic Table Reorg

- SAP-LRP system with ~450GB of data in uncompressed form
- ~24,000 tables, 99.7% of the data in 1,500 tables
- Comparison between database build (ADC) and full Classic Table Reorg
- Classic Row Compression with ADC yields 71% larger data size than after Reorg
- Adaptive Compression with ADC is only 16% larger than after Reorg
- Adaptive Compression with ADC is 14% smaller than Classic Row Compression after Reorg (i.e. best possible compression)



Query Performance – SAP-BW Benchmark (1TB)

- **Additional CPU consumption for compression & expansion operations**
  - Row compression design paradigm: Compression under logical I/O
  - Rule of thumb: Compression is beneficial in I/O-bound situations
- **Performance measurements done from two angles:**
  - 1. Focus on additional CPU cost (to minimize overhead)
  - 2. Elapsed time studies in benchmark scenarios
- **Most long-running queries experience dramatic speed-up**
- **Overall runtime reduced by 62%**
- **Median query speedup of 43%**



Row Compression – Enablement & Tools

- **How to enable row compression?**
  - Must have DB2 Storage Optimization Feature
  - To enable classic row compression

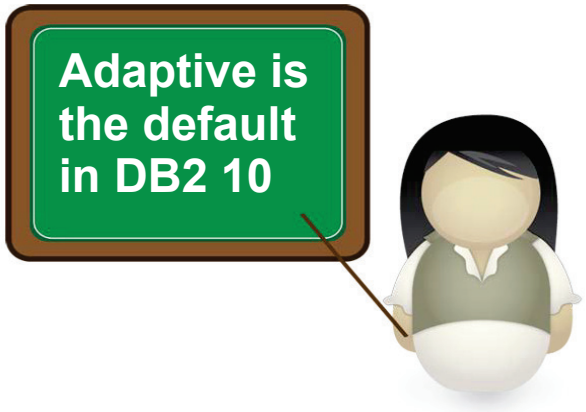
```
CREATE TABLE / ALTER TABLE … COMPRESS YES STATIC
```

  - To enable adaptive row compression

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

  - To disable compression

```
CREATE TABLE / ALTER TABLE … COMPRESS NO
```



- **Data is compressed after the table dictionary is created.**
  - **INSERT/UPDATE/LOAD/IMPORT** can trigger the automatic dictionary creation
  - Classic **REORG** with **RESETDICTIONARY** option will always generate a new dictionary and compress all table data

Row Compression - Example Scenarios

1) Compressing data for new table

```
CREATE TABLE Sales (<columns definition>) COMPRESS YES
```

Load data… Automatic Dictionary Creation (ADC) will kick off and create compression dictionary. Once dictionary is built, new data put into the table is compressed:

```
LOAD FROM file OF DEL REPLACE INTO NewSale
```

2) Compressing data in existing tables

```
ALTER TABLE Sales COMPRESS YES
```

Data is still un-compressed. Explicitly compress data via REORG:

```
REORG TABLE Sales
```

3) Recreating the dictionary to optimize compression

(Classic Row Compression) Data has changed a lot so current dictionary is not so effective anymore. Use REORG to recreate dictionary and re-compress data:

```
REORG TABLE Sales RESETDICTIONARY
```

4) Uncompressing your data

Disable compression:

```
ALTER TABLE Sales COMPRESS NO
```

Uncompress data:

```
REORG TABLE Sales
```

Adaptive Compression greatly reduces the need for REORGs to maintain the compression ratio.

Row Compression – Enablement & Tools

- **Estimating storage savings**
  - ADMIN_GET_TAB_COMPRESS_INFO_V97 ← **Deprecated in DB2 10!**
  - Instead use: ADMIN_GET_TAB_COMPRESS_INFO and ADMIN_GET_TAB_DICTIONARY_INFO

```
SELECT SUBSTR(TABNAME,1 ,10) tabname, OBJECT_TYPE, ROWCOMPMODE,
PCTPAGESSAVED_CURRENT current, PCTPAGESSAVED_STATIC with_static,
PCTPAGESSAVED_ADAPTIVE with_adaptive
FROM TABLE(SYSPROC.ADMIN_GET_TAB_COMPRESS_INFO 'DB2INST1','CUSTOMERS')) AS T;
```

TABNAME	OBJECT_TYPE	ROWCOMPMODE	CURRENT	WITH_STATIC	WITH_ADAPTIVE
CUSTOMERS	DATA	S	60	68	81
CUSTOMERS	XML	S	58	62	62

Temporary Table Compression

- **Enabled automatically with the DB2 Storage Optimization Feature**
- **Only classic row compression is used for temporary tables**
- **System temporary tables**
  - DB2 Optimizer analyzes storage savings and performance impact, if it is worthwhile to compress, classic row compression is used automatically
- **User temporary tables**
  - Created global temporary tables (CGTTs) and declared global temporary tables (DGTTs) are always enabled for compression
- **Use `db2pd` to see whether the optimizer used compression for system temporary tables**

```
db2pd -db <database> -temptable
```

Index Compression

- **Especially useful for large OLTP and data warehouse environments**
- **Index is compressed by default for compressed tables**
- **Can specify `COMPRESS YES` option in `CREATE INDEX`**
- **Different algorithms implemented by the DB2 engine for:**
  - RID list compression
  - Prefix compression
  - Variable slot directory

```
CREATE INDEX <index_name> COMPRESS YES

ALTER INDEX <index_name> COMPRESS YES
REORG INDEXES ALL FOR TABLE <table_name>
```

Index Compression – Monitoring

- **How much space *could I* save by compressing the indexes on table T1 ?**

```
SELECT index_name, pages_saved_percent, compress_attr, index_compressed
FROM TABLE(SYSPROC.ADMIN_GET_INDEX_COMPRESS_INFO ('T', 'myschema', 'T1', NULL, NULL)) AS T
```

```
INDEX_NAME          PERCENT_PAGES_SAVED      COMPRESS_ATTR       INDEX_COMPRESSED
----------------    ----------------------   ------------------  --------------------
INDEX1               57                       N                   N
```

Index Compression – Monitoring

- **How much space *did I* save by compressing the indexes on table T1 ?**

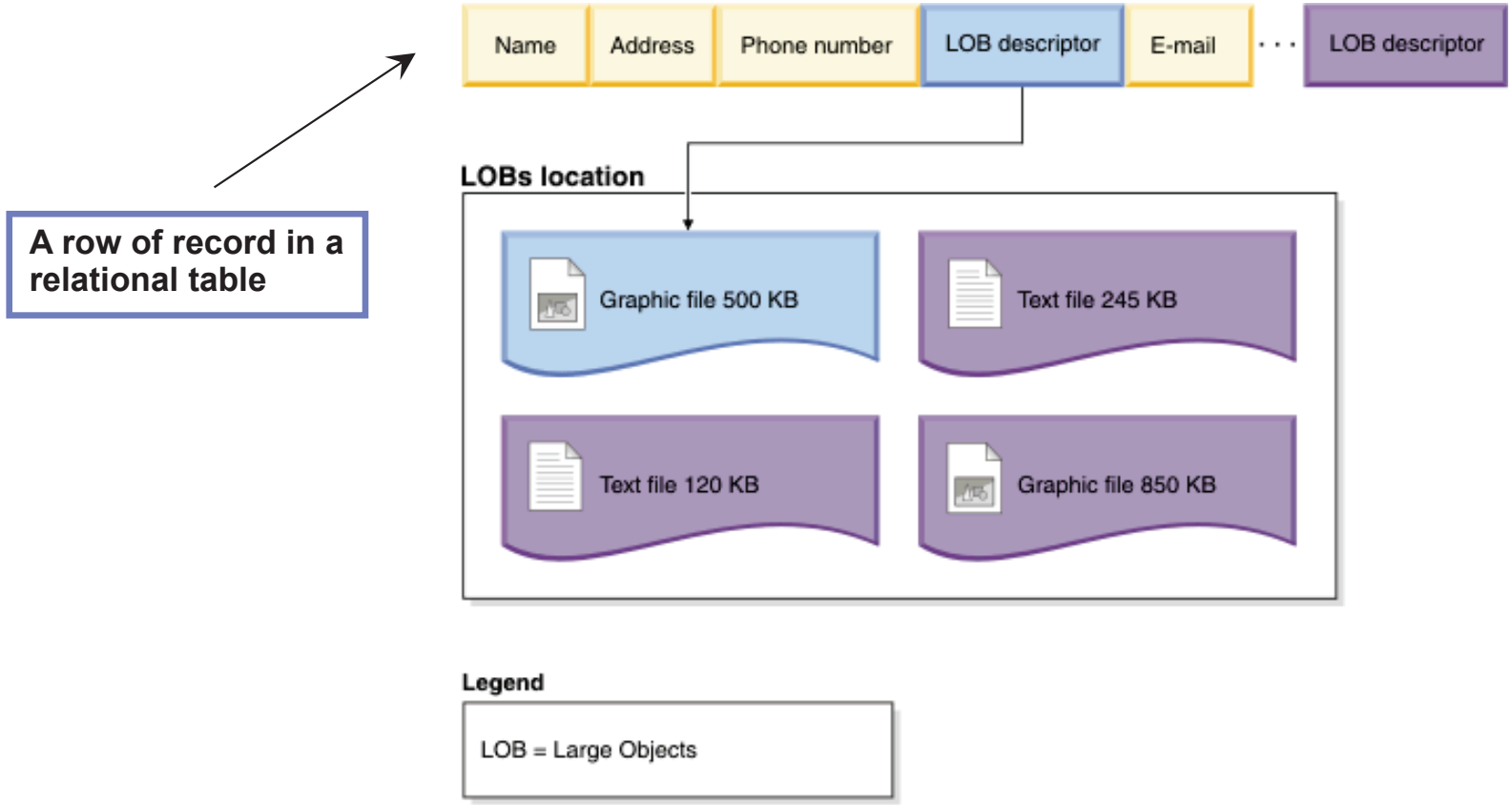
```
ALTER INDEX index1 COMPRESS YES
REORG INDEXES ON TABLE t1
RUNSTATS ON TABLE t1

SELECT index_name, pages_saved_percent, compress_attr, index_compressed
  FROM TABLE (SYSPROC.ADMIN_GET_INDEX_COMPRESS_INFO 'T', 'myschema', 'T1', NULL, NULL)) AS T
```

```
INDEX_NAME       PERCENT_PAGES_SAVED   COMPRESS_ATTR          INDEX_COMPRESSED
---------------- --------------------- ---------------------- --------------------------
INDEX1           58                    Y                      Y
```

actual savings

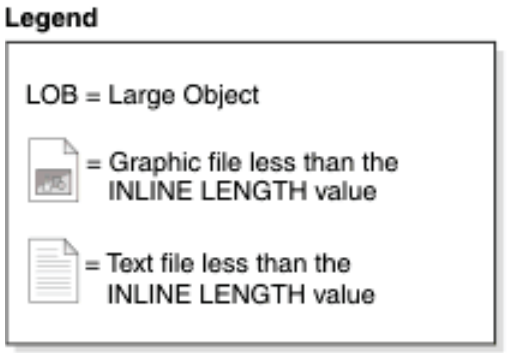
XML and LOB Inlining

- **Default storage behavior**
  - All XML documents stored in XML storage object, also referred to as XDA (XML Data Area)
  - All LOB (Large Objects) stored in LOB storage object
  - Descriptors in rows identify location of each object



XML and LOB Inlining

- **Inlining**
  - Allows small (<32KB for a 32KB page size) XML documents and LOBs to be stored in the base table row
  - Better performance and reduced storage needs




```
CREATE TABLE PROJECTS (PID INTEGER, PLAN XML INLINE LENGTH 300, STARTDATE DATE, …)

ALTER TABLE PROJECTS ALTER COLUMN PLAN SET INLINE LENGTH 1004
```

XML and LOB Compression

- **XML (XDA) Compression**
  - Available since DB2 9.7
  - Same dictionary approach used for Table Compression
  - Enablement is via the table `COMPRESS YES` option
  - Classic/'Offline' REORG table based
    - Use LONGLOBDATA option to compress XML data

```
CREATE TABLE mytab1 COMPRESS YES
ALTER TABLE mytab1 COMPRESS YES
```

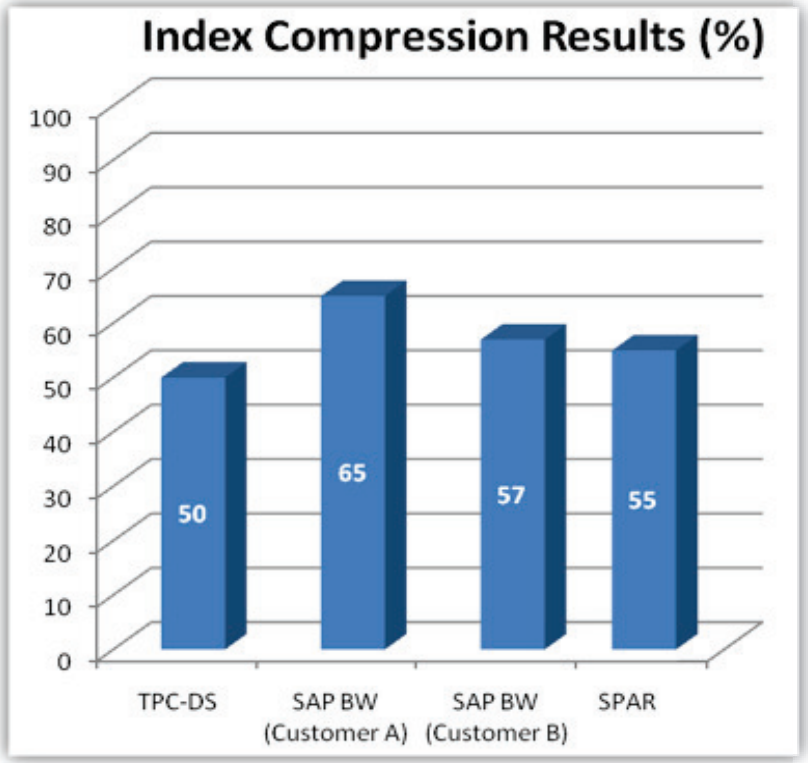
- **Inlining**
  - Inlined XML or LOB data can be compressed with both adaptive and classic row compression

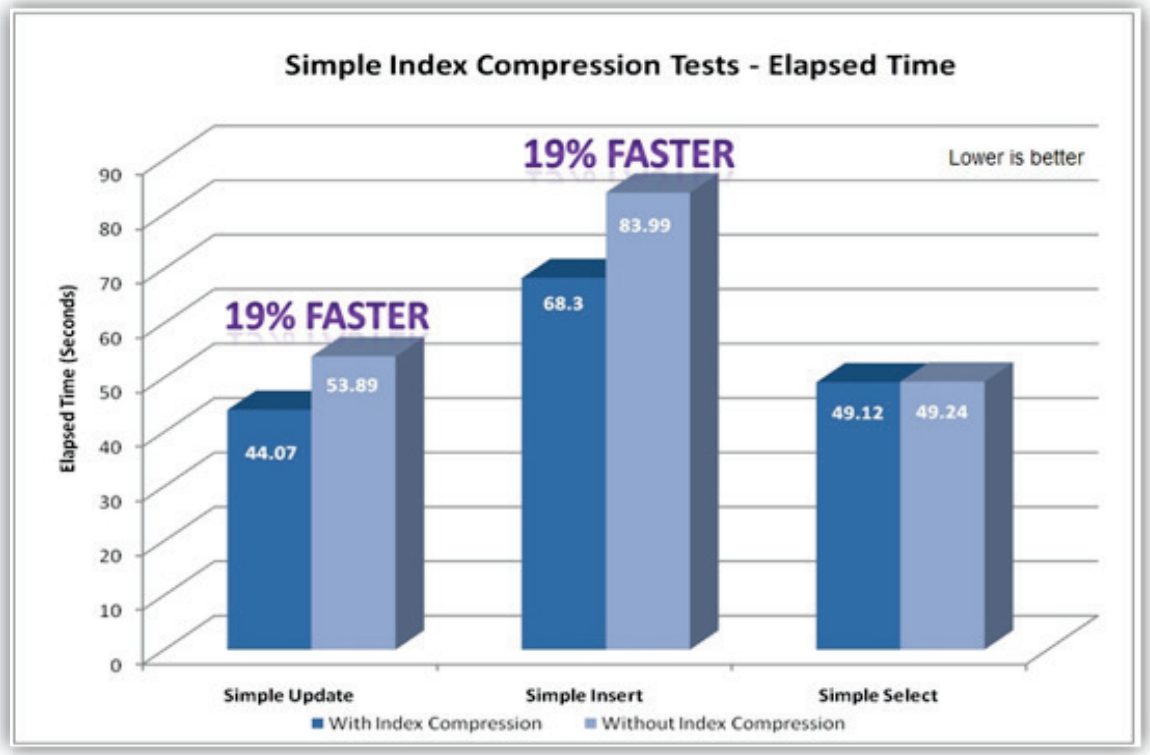
New in DB2 10

Compression Benefits

Index Compression

- Fewer index levels
  - Fewer logical and physical I/Os for key search (insert, delete, select)
  - Better buffer pool hit ratio
- Fewer index leaf pages
  - Fewer logical and physical I/Os for index scans
  - Fewer splits
  - Better buffer pool hit ratio

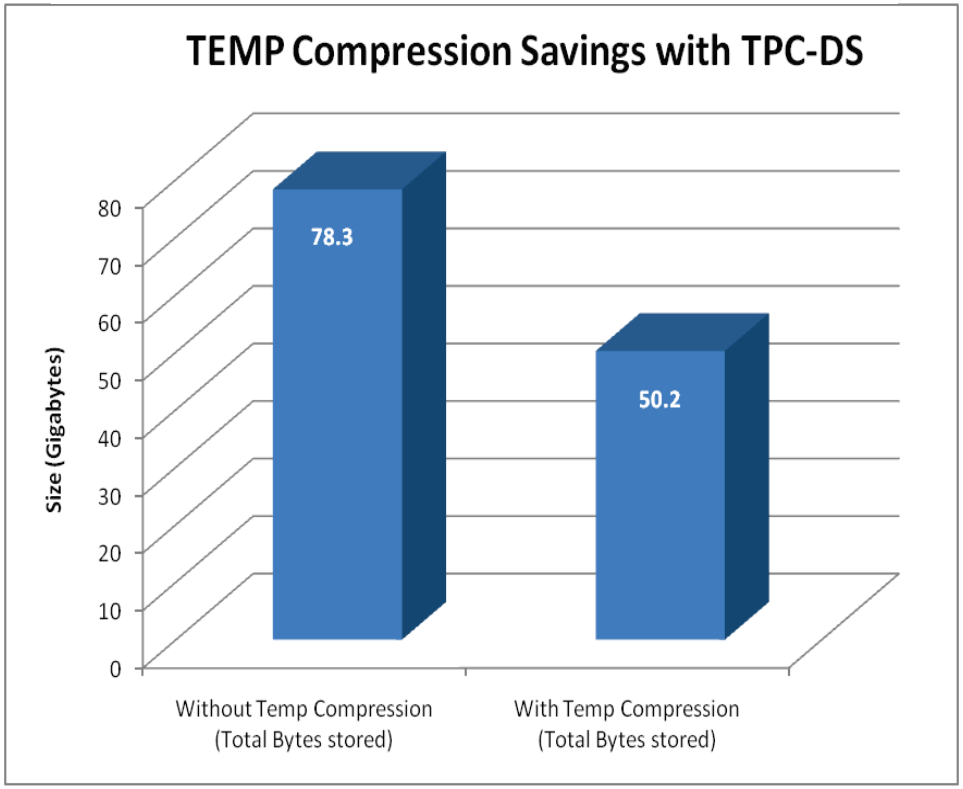


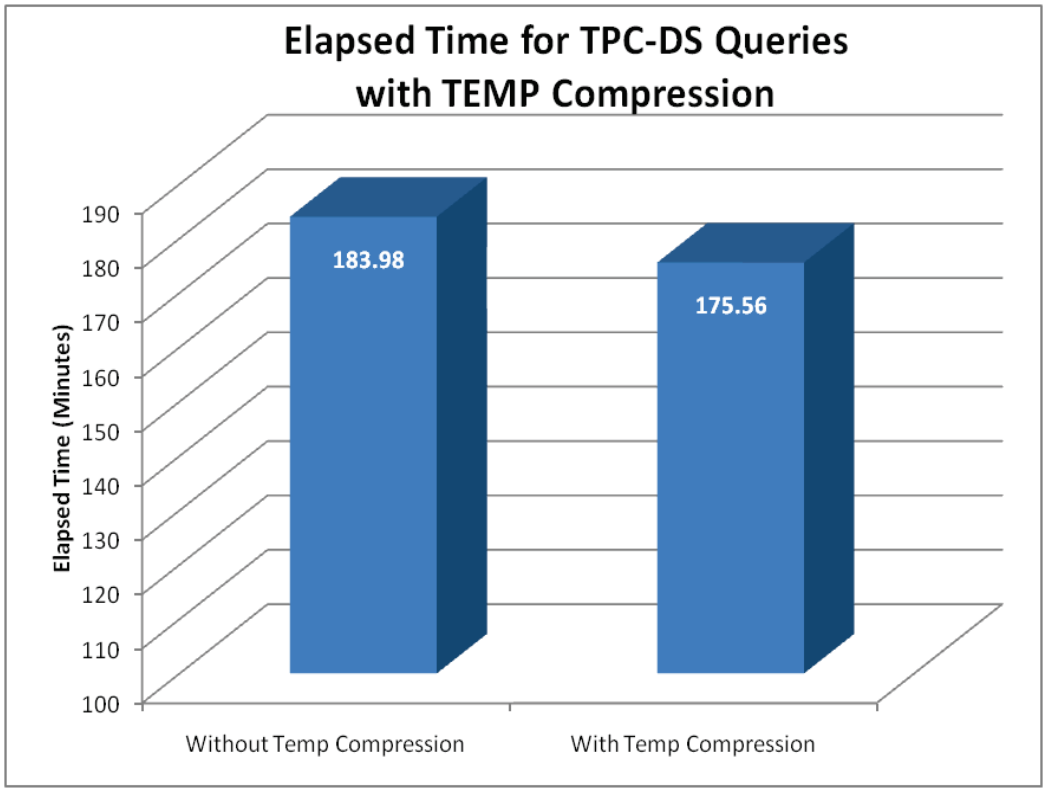


Compression Benefits

Temp Table Compression

- Only vendor in industry to compress temp tables
  - Internal (DB2 utilities, SORT) and external (DGTTs) compressed
- Savings and performance benefits require no DBA intervention





Archive Log Compression

- **New in DB2 10 – archive log compression reduces log archive storage**
  - Automatically compresses log extents on-the-fly
  - Uses the same algorithm as (default) backup compression
- **Simply turn it on and DB2 does the compression for you**
  - Enabled by `Logarchcompr1 and/or Logarchcompr2` db configuration parameter set to ON

Large SAP customer generates 60GB of log per day and they keep 8 weeks of archives

- Storing 3.3TB of archived log files
- Compression of **4x** results in storage of **only 825GB for 8 weeks**

Client Success

What Customers are Saying ...

- **Customer experiences are consistent**
  - Tables will compress in the range of 60% - 80%
  - Indexes compress around 50%
  - Temp space will compress around 35%
  - Overall database storage savings will be between 50% and 65%

Summary

- **Storage optimization through DB2 10 compression can save 55-70% of your database storage requirements**
- **You not only see your online database shrink but often more importantly, your backup storage and disaster recovery storage is cut in half as well**
- **In real customer examples storage savings are realized and performance benefits are apparent**
- **DB2 10 offers better compression ratios, improved ease of use and higher data availability**
  - Adaptive compression provides better compression than classic row compression
  - Adaptive compression adapts to date skews over time – i.e. no REORG needed to maintain compression ratio

DB2® Security

Agenda

- Authentication
- Authorization
- Privileges
- Roles
- Trusted context
- Audit
- Data encryption
- Advanced security
  - Label-based Access Control (LBAC)
  - Row and Column Access Control (RCAC)

Authentication

- **Determines who the user is by identifying with a password**
  - **User authentication is completed by a security facility outside DB2.**
  - DB2 does not store nor maintain user/password information.
  - By default, the operating system's security mechanism is used.
- The following authentication types are available:
  - **CLIENT:**
    - Authentication performed on the client
  - **SERVER:**
    - Authentication occurs on the server
  - **SERVER_ENCRYPT:**
    - Same as SERVER but user/pwd are encrypted
  - **DATA_ENCRYPT:**
    - SERVER authentication and encryption of user data
  - **DATA_ENCRYPT_CMP:**
    - Same as DATA_ENCRYPT, but also allows use of SERVER_ENCRYPT for products that don't support DATA_ENCRYPT
  - **KERBEROS:**
    - Uses Kerberos security protocol
  - **KRB_SERVER_ENCRYPT:**
    - Server accepts KERBEROS or SERVER_ENCRYPT methods
  - **GSSPLUGIN:**
    - Uses a GSS-API plug-in to perform authentication
  - **GSS_SERVER_ENCRYPT**
    - Server accepts GSSPLUGIN or SERVER_ENCRYPT methods

Authorization

- **Determines what database operations users can perform and which data objects users can access**
- A user can acquire permissions in several ways
  - Granted directly to the **user ID**
  - Inherited from a **user group**
  - Inherited through **Roles** assigned to the user ID
  - Permissions acquired through a **trusted context**
  - Permissions assigned to **PUBLIC**.
    - *PUBLIC is a special group that consists of all users, including future users.*
- There are 3 types of permissions that can be granted
  - **Authority Levels**
  - **Privileges**
  - **LBAC Credentials** (*briefly covered in this presentation*)

Authorities

- An **Authority** is a group of privileges and permissions over database manager operations.
  - **Database-specific authorities**
  - **System (or instance level) authorities**

Level	Authorization	Description
System	SYSADM	System administrator
	SYSCTRL	Control over operations that affect system resources
	SYSMAINT	Ability to perform maintenance operations
	SYSMON	Ability to use database system monitor
Database	DBADM	Database administrator (only may be granted by SECADM)
	SECADM	Security administrator
	SQLADM	Authority to monitor and tune SQL within a database
	WLMADM	Ability to manage WLM assets
	EXPLAIN	Explain query plans without access to data
	ACCESSCTRL	Ability to issue limited grant and revoke statements
	DATAACCESS	Data access authority
	LOAD	Use of load utility

Privileges

- A **Privilege** is a single permission enabling the user to create/access a database resource.
  - Stored in the database catalog

Explicit	Implicit	Indirect
•User •Group •Role	•When a database or database object is created	•Inherited through execution of packaged code

- **Explicit**
  - GRANT

```
DB2 GRANT SELECT ON TABLE person TO USER employees
```

  - REVOKE

```
DB2 REVOKE SELECT ON TABLE person TO USER employees
```

- **Implicit**
  - Owner of the table is implicitly granted owner privileges

```
DB2 CREATE TABLE mytable
```

- **Indirect**
  - Execute permission on package that executes a select, indirectly granted select privileges

Roles

- Database object that groups together one or more privileges.
- Similar to User Groups but without the same restrictions
  - Hierarchical: you can assign roles to roles
  - Stored inside the database: changes to roles and its permissions are effective immediately to users
- Can be assigned to users, groups, PUBLIC or to other roles via a GRANT statement
- Step 1 – Create Role

```
CREATE ROLE developer
```

- Step 2 – Assign Privileges to a Role

```
GRANT SELECT ON TABLE server TO ROLE developer
```

- Step 3 – Grant Role to Users

```
GRANT ROLE developer TO USER Bob, USER Alice
```

- Step 4 – Revoke Role as Necessary

```
REVOKE ROLE developer FROM USER Bob
```

Roles – Admin Option and Hierarchies

- ADMIN OPTION
  - Allow for the user to grant or revoke the role when granting the role

```
GRANT ROLE developer TO USER Bob WITH ADMIN OPTION
```

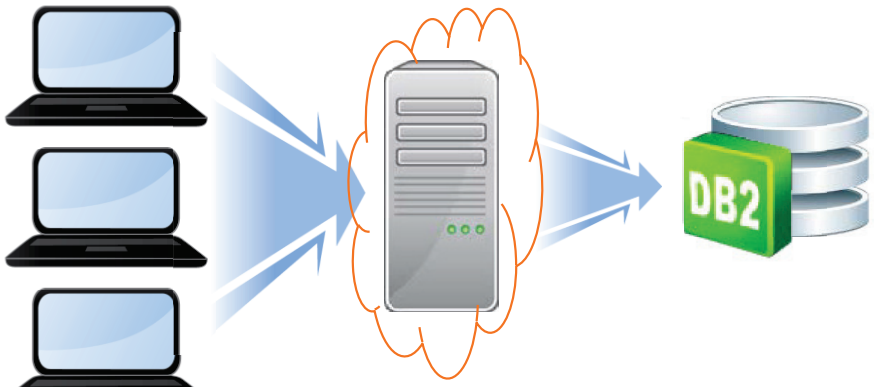
- Hierarchies
  - A role may be granted membership into another role
  - Inherit privileges

```
CREATE ROLE employee;
CREATE ROLE management;
CREATE ROLE executive;

GRANT ROLE employee TO ROLE management;
GRANT ROLE management TO ROLE executive;
```

Trusted Context

- A trusted relationship between the DB and the application that allows extra capabilities:
  - Switch current user ID
  - Acquire additional privileges via role inheritance
- Provide a means whereby the end-user identity in a three-tier environment can be easily and efficiently propagated to the database server
- Relationship identified by connection attributes
  - IP Address, Domain Name, Authorization ID, Data Encryption used

```
CREATE TRUSTED CONTEXT ctxt
BASED UPON CONNECTION USING SYSTEM AUTHID smith
ATTRIBUTES(address '192.168.2.27') DEFAULT ROLE managerRole ENABLE
```



Creating and Applying Auditing

- Step 1: **Create an Audit Policy** to monitor and track activities. You can generate records for specific:
  - Activity Types (eg: All activities, Security Maintenance or all SQL Statements)
  - Activity Type Outcomes (Successes, Failures, Both, None)

```
CREATE AUDIT POLICY telleraction CATEGORIES execute
STATUS both ERROR TYPE audit
```

- Step 2: **Apply the Audit Policy** to any of the following objects or users
  - A whole database, tables, trusted contexts, authorization IDs (users, groups and roles), authorities
    - Example: Create an audit policy to monitor all SQL statements executed by the users in the teller role and all error messages

```
AUDIT ROLE teller USING POLICY telleraction
```

Collecting Audit Records

- Step 3: Allow regular database activities to continue, audit records will be created

- Step 4: Archive audit records
  - Use the provided SYSPROC.AUDIT_ARCHIVE stored procedure/table function to archive the active audit log

```
CALL SYSPROC.AUDIT_ARCHIVE(NULL, NULL)
```

Reading and Loading Audit Records

- Step 5: Put Audit records into delimited file format
  - Use the provided SYSPROC.DELIM_EXTRACT stored procedure to extract data from the archive logs and load into a delimited file format

```
CALL SYSPROC.AUDIT_DELIM_EXTRACT(NULL, '$HOME/AUDIT_DELIM_EXTRACT',
NULL, '%20070618%', 'CATEGORY EXECUTE')
```

- Step 6: Create and load audit tables
  - Run the provided script db2audit.ddl to create the tables necessary to hold the audit data
  - The script creates 8 needed tables for audit files: AUDIT, CHECKING, OBJMAINT, SECMAINT, SYSADMIN, VALIDATE, CONTEXT and EXECUTE
  - Load data into the tables using the LOAD command

```
LOAD from execute.del of DEL MODIFIED BY DELPRIORITYCHAR LOBSINFILE
INSERT INTO schema.EXECUTE
```

- Step 7: Begin querying the tables and creating reports!

Data Encryption

- External Tools
  - **IBM Database Encryption Expert** to encrypt the underlying operating system data and backup files
  - **OS file-level encryption** (eg: AIX's encrypted file system)
- Built-in function can encrypt data based on a provided password
  - **ENCRYPT (<data>, <pwd>, <pwd_hint>)**
    - Encrypts <*data*> based on password <*pwd*>
    - Result is a VARCHAR FOR BIT DATA value

```
INSERT INTO EMP(SSN)
VALUES ENCRYPT('289-46-8832','Pacific','Ocean')
```

- Built-in function to retrieve password hints
  - **GETHINT (<enc_data>)**
    - Returns the password hint if one was provided
  - ***Password management is responsibility of the user***

```
SELECT GETHINT(SSN) FROM EMP;
```

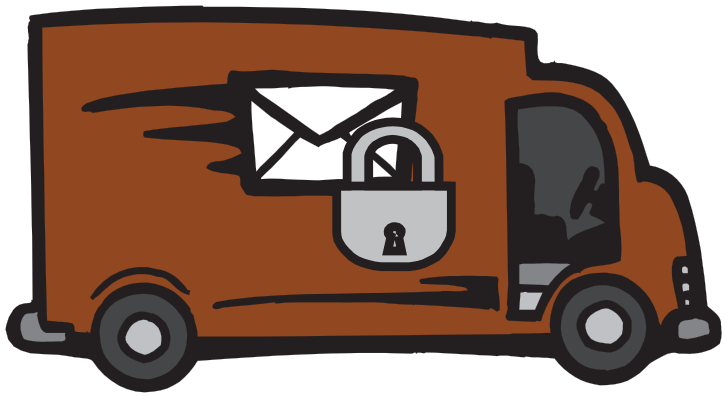
Data Decryption

- Built-in functions can decrypt data based on a provided password
  - **DECRYPT_BIN (<enc_data>, <pwd>)**
    - Decrypts <enc_data> and returns the value as VARCHAR FOR BIT DATA
  - **DECRYPT_CHAR (<enc_data>, <pwd>)**
    - Decrypts <enc_data> and returns the value as VARCHAR

```
SELECT DECRYPT_CHAR(SSN,'Pacific') FROM EMP
```

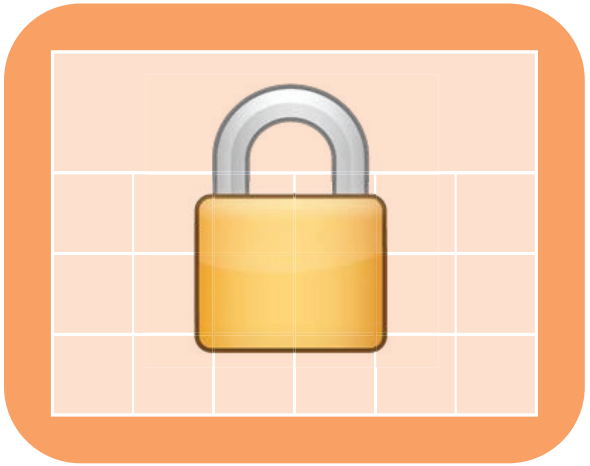
Encrypting Data in Transit

- **DATA_ENCRYPT authentication type**
  - During authentication, user and password are encrypted
  - The following data is encrypted:
    - **SQL** and **XQuery** statements
    - SQL program **variable data**
    - **Output data** from the server processing of an SQL or XQuery statement and including a description of the data
    - Some or all of the **answer set data** resulting from a query
    - Large object (**LOB**) data streaming
    - **SQLDA descriptors**
- **Secure Sockets Layer (SSL)**
  - CLI, CLP, and .Net Data Provider client applications and applications that use the IBM® Data Server Driver for JDBC and SQLJ (type 4 connections) support SSL
  - DB2 also supports SSL's successor, Transport Layer Security (TLS)



Fine Grained Access Control

- Regular SQL privileges can only protect tables as a whole.
- DB2 offers 2 solutions that complement the table privileges model:

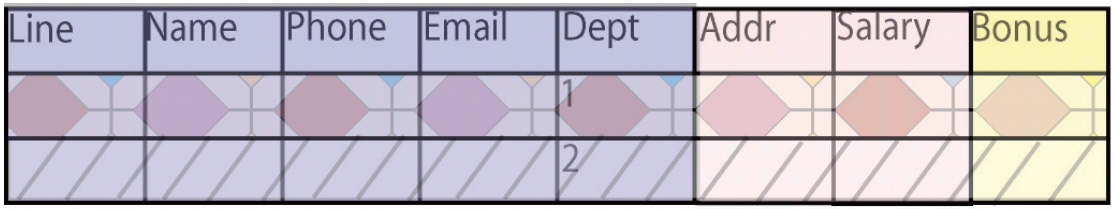


- **LBAC (Label-Based Access Control)** is a fixed label security model designed for environments with classified data
- Hierarchical access scenarios
  - Great for large companies with well defined data and user classifications
  - Suited for such applications as those intelligence and defense communities

- **RCAC (Row and Column Access Control)** is a general purpose security model
  - No data or user classification required
  - Best suited for commercial customers

Label-Based Access Control

- **Label Based Access Control**
  - Sets security labels at the row level, column level or both
- LBAC complements the traditional DB2 Discretionary Access Control (DAC)
- How LBAC works
  - Assign labels to database objects and users
  - DB2 compares the labels whenever an object is accessed
  - Access is granted based on the LBAC security policy defined



Label Name	SENS
EMP_TYPE Component	HR
DEPT Component	Blank

Column Label

Label Name	DEPT_2
EMP_TYPE Component	Blank
DEPT Component	DEPT_2

Row Label

User: Dayna

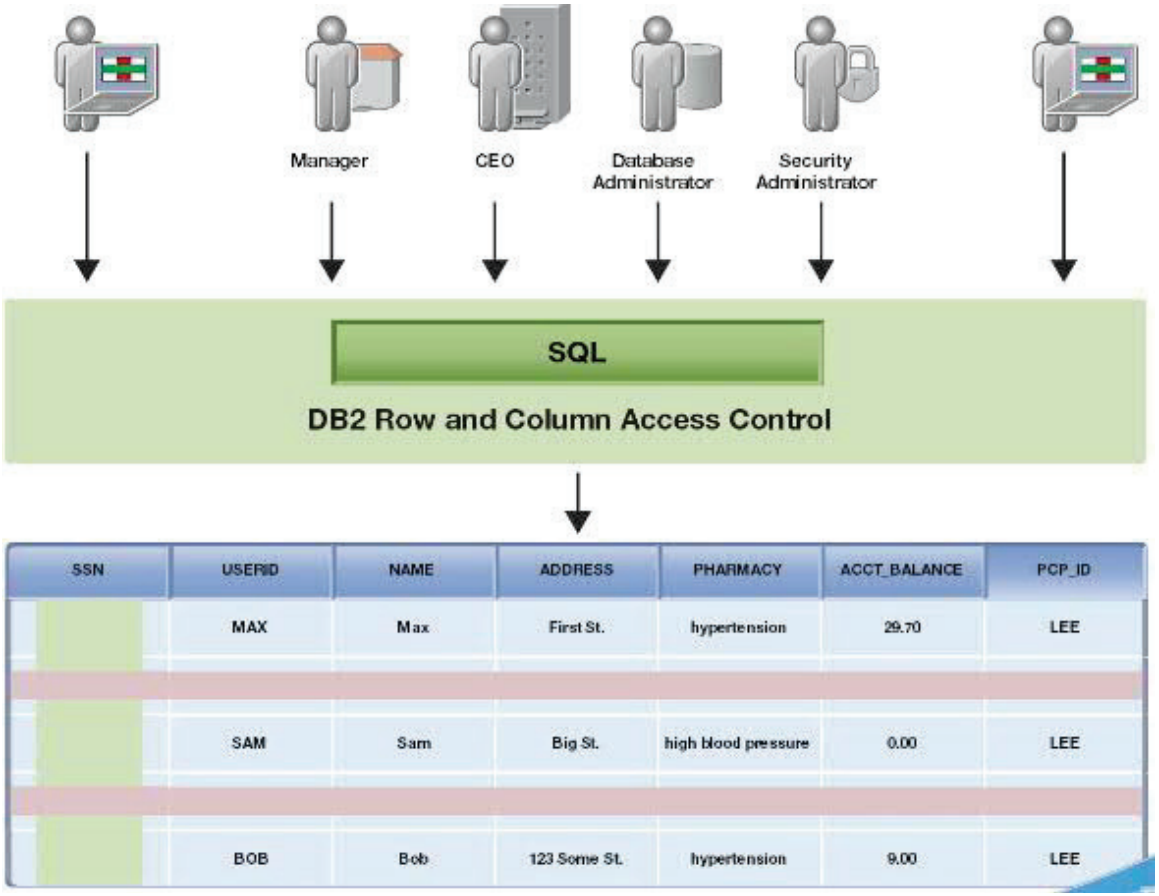
Label Name	MANAGER
EMP_TYPE Component	MANAGER
DEPT Component	DEPT_GEN

User Label

Row and Column Access Control

RCAC is:

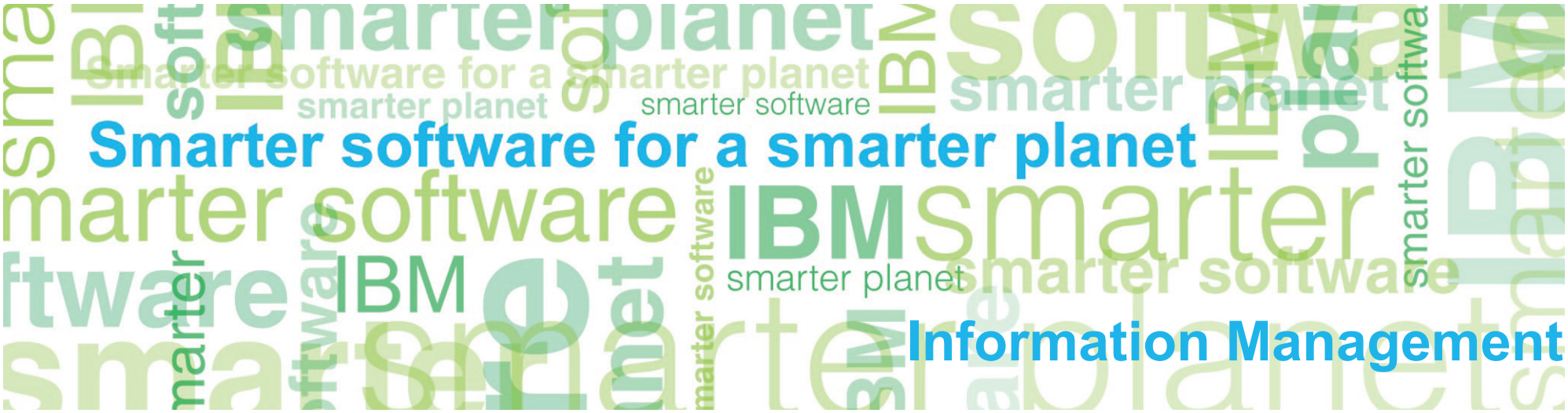
- Additional layer of data security introduced
- Complementary to table level authorization
- Controls access to a table at the row, column or both levels
- Two sets of rules, permissions for rows and masks for columns



Summary

- Identifying the incoming connection is done by authentication, what the connection is able to do is classified by authorizations and privileges
- Roles allow for the grouping of many different privileges for a centralized point
- Trusted context allows for the identification of the incoming connection by defined attributes in a multi-tier environment
- Auditing allows for the monitoring of work
- Built-in functions allow for the encryption and decryption of data stored
- Advanced security mechanisms include LBAC and RCAC
    - **Label-Based Access Control** controls access by assignment of security labels to objects and users
    - **Row and Access Column Control** controls access by definition of row permissions and column masks

DB2® Backup and Recovery

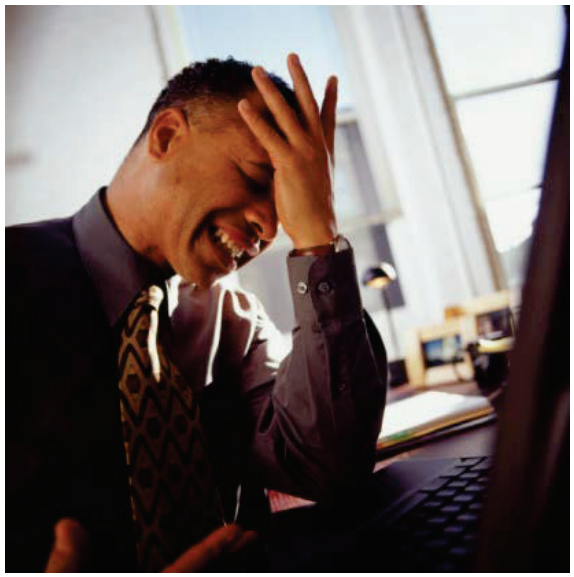


Agenda

- **Why back up data**
- **Basic backup and recovery concept**
- **Logging**
  - Log file states
  - Logging types
  - Infinite logging
  - db2cklog
- **Backup**
  - Backup modes
  - Partitioned database
  - Table spaces
  - Incremental backup
  - db2ckbkp
- **Recovery**
  - Recovery types
- **Restore**
  - Table space restore
  - Incremental restore
  - Redirected restore
  - Recover Utility
- **IBM DB2 Advanced Recovery Feature**

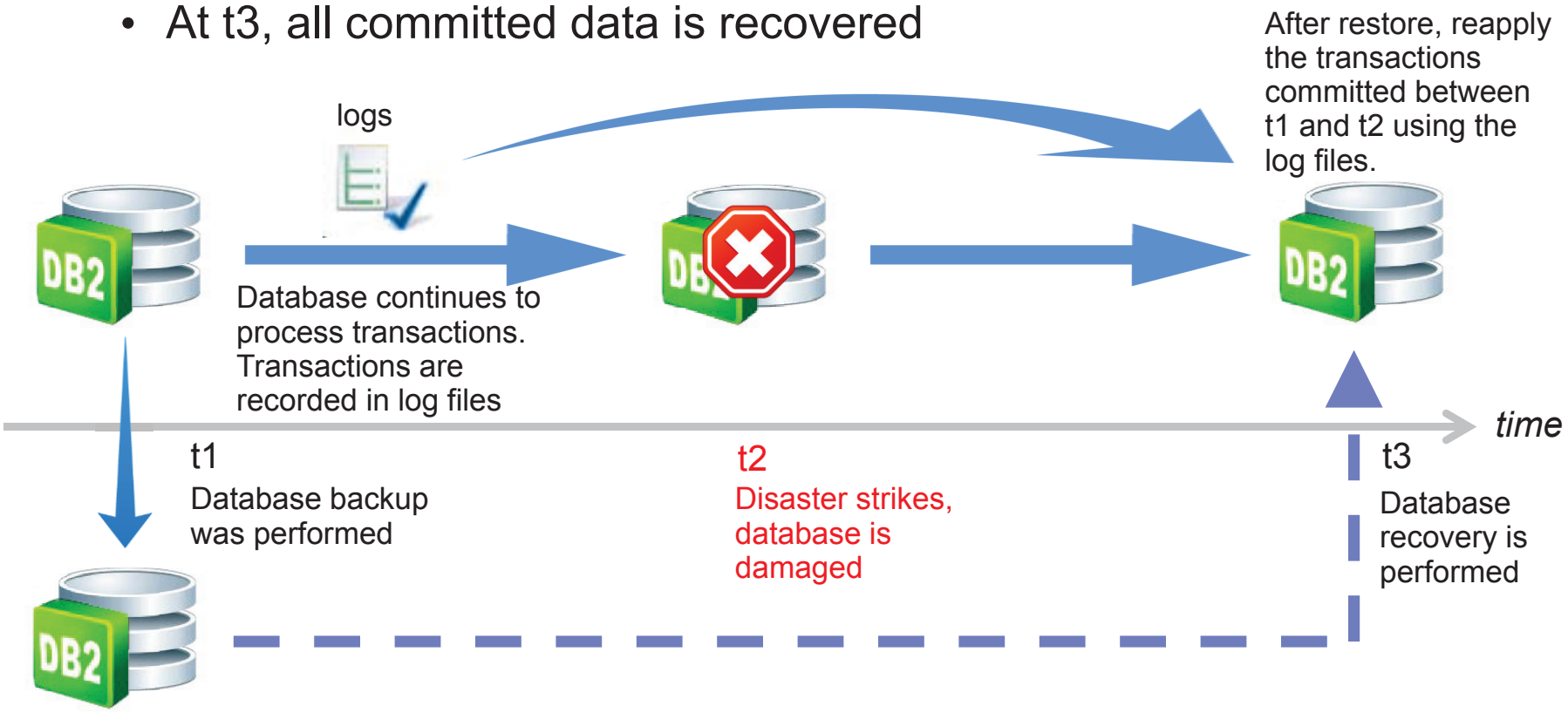
Why Backup Data

- Backing up data is vital for businesses
  - Lost information can cause a major crisis or worse, lead to business failure.
- Common problems:
  - System outage
    - Power failure
    - Hardware failure
  - Transaction failure
    - Users may inadvertently corrupt the database
  - Media failure
    - Disk drive becomes unusable
  - Disaster
    - Database facility damaged by fire, flooding or other catastrophe
- DB2 backup and recovery methods are designed to help you keeping your information safe!



Basic Concept of Backup and Recovery

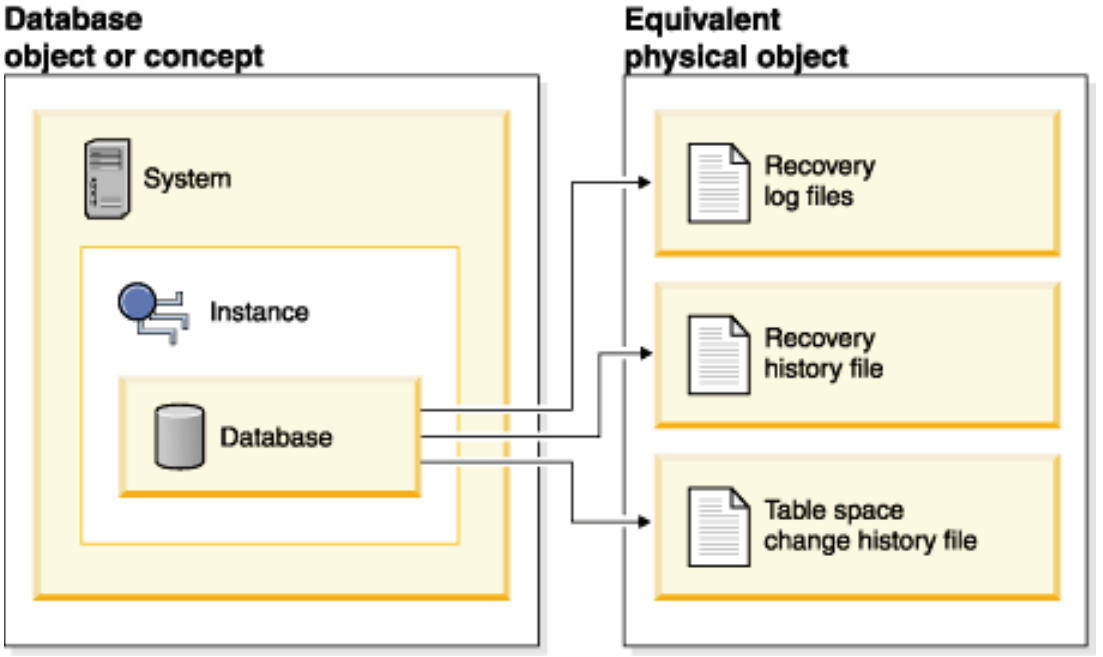
- **Process of making copies of data and which may be used to restore the original in case of failure or loss of data**
  - E.g.
    - At t1, a database backup operation is performed
    - At t2, a problem that damages the database occurs
    - At t3, all committed data is recovered



Database Logging

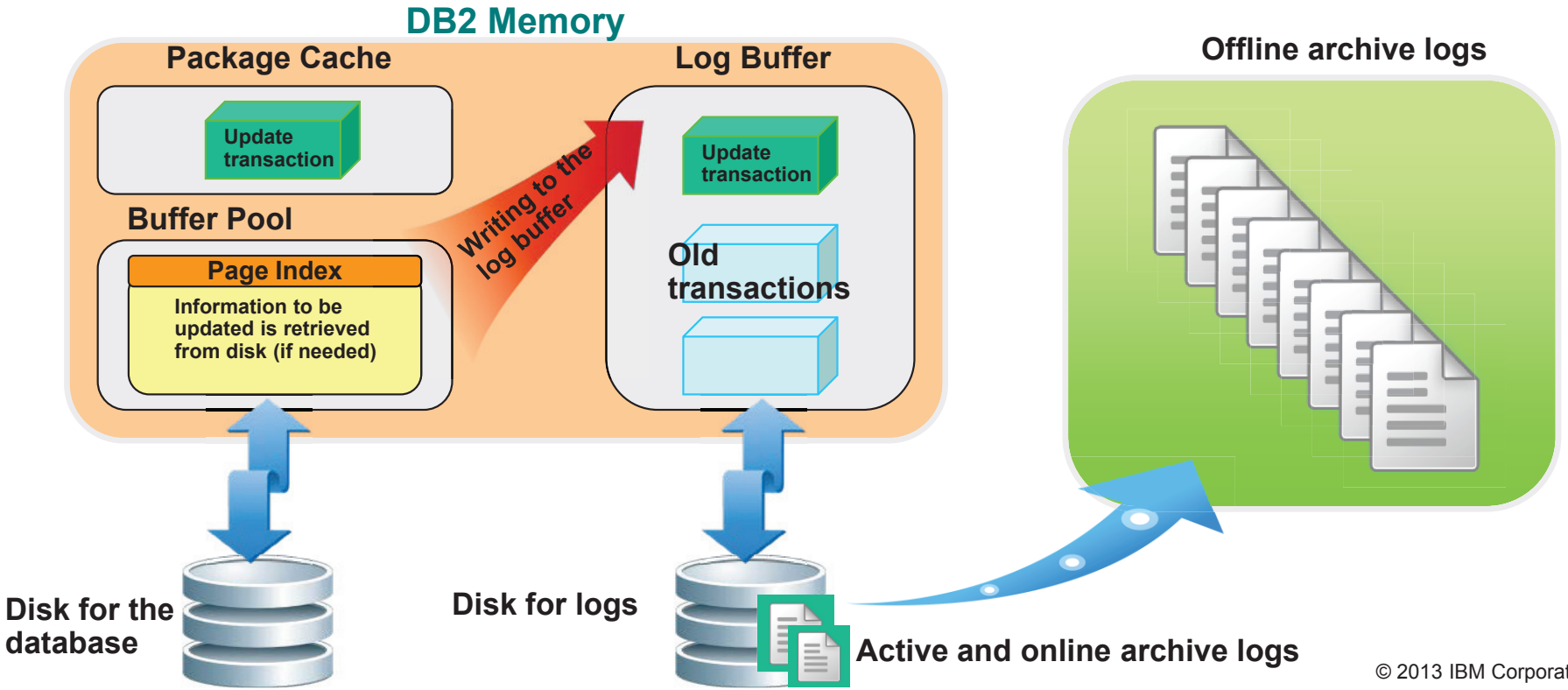
- **Keep track of changes made to database objects and data**
- **During the recovery process, DB2 examines these logs and decides which changes to redo or undo**
- **Key element of any high availability strategy**
- **Can be stored in files or on devices**
- **Log records are written to disk when one of the**
- **following occurs:**
  - Log buffer is full;
  - A transaction commits;
  - A group of transactions
  - commit, as defined by the
  - `mincommit` value

Deprecated in DB2 10



Log File States

- **Active logs**
  - Contain at least 1 transaction that has not been committed or rolled back
- **Online archive logs**
  - Contain committed and externalized transactions in the active log directory
- **Offline archive logs**
  - Contain committed and externalized transactions in a separate repository

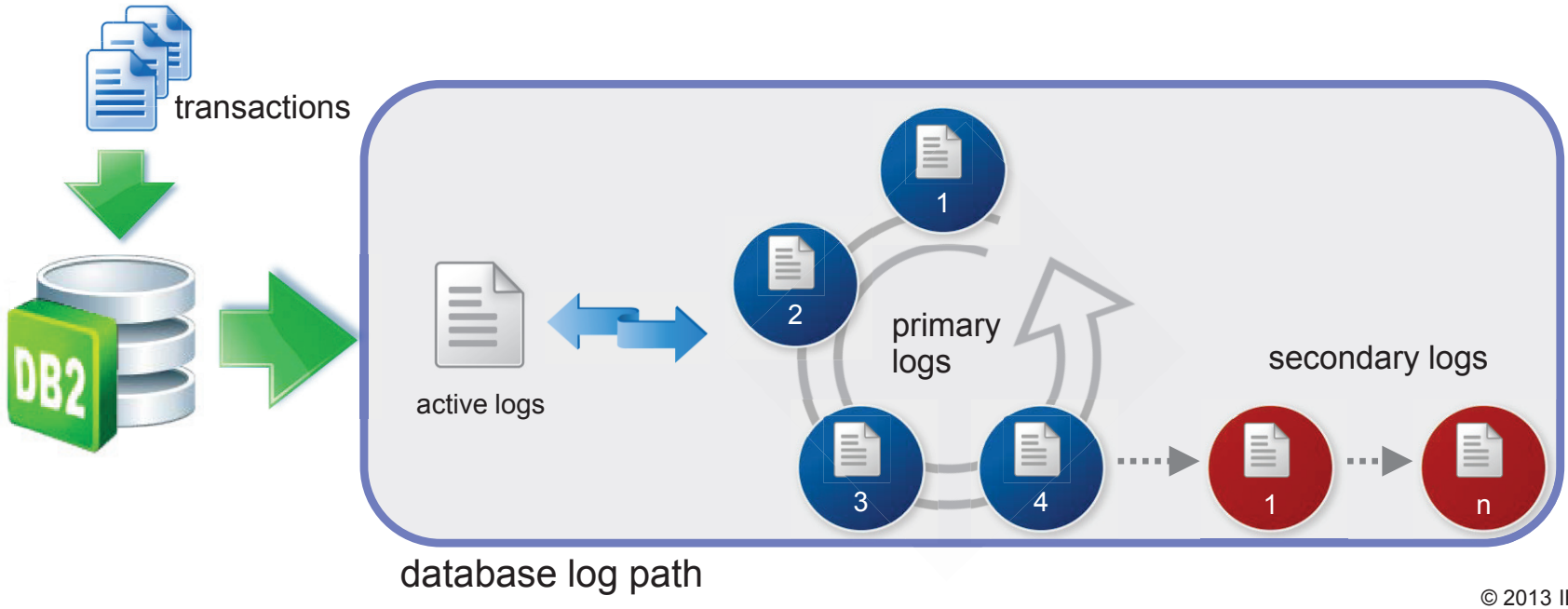


Key Logging Configuration Parameters

- **LOGPRIMARY**
  - number of primary log files to be pre-allocated in the active logs directory
- **LOGSECOND**
  - number of secondary log files that are created and used for recovery log files (only as needed).
- **LOGBUFSZ (Log Buffer Size)**
  - amount of the database heap (defined by DBHEAP parameter) to use as a buffer for log records before writing these records to disk
- **LOGFILSIZ (Log File Size)**
  - size of each primary and secondary log file in unit of 4KB pages.
- **LOGPATH and NEWLOGPATH**
  - LOGPATH is the default active log directory
  - Changed to a user defined location using NEWLOGPATH.
- **FAILARCHPATH (Failover log archive path)**
  - Specifies a third target to archive log files if the primary and secondary archival paths fail

Logging Types - Circular

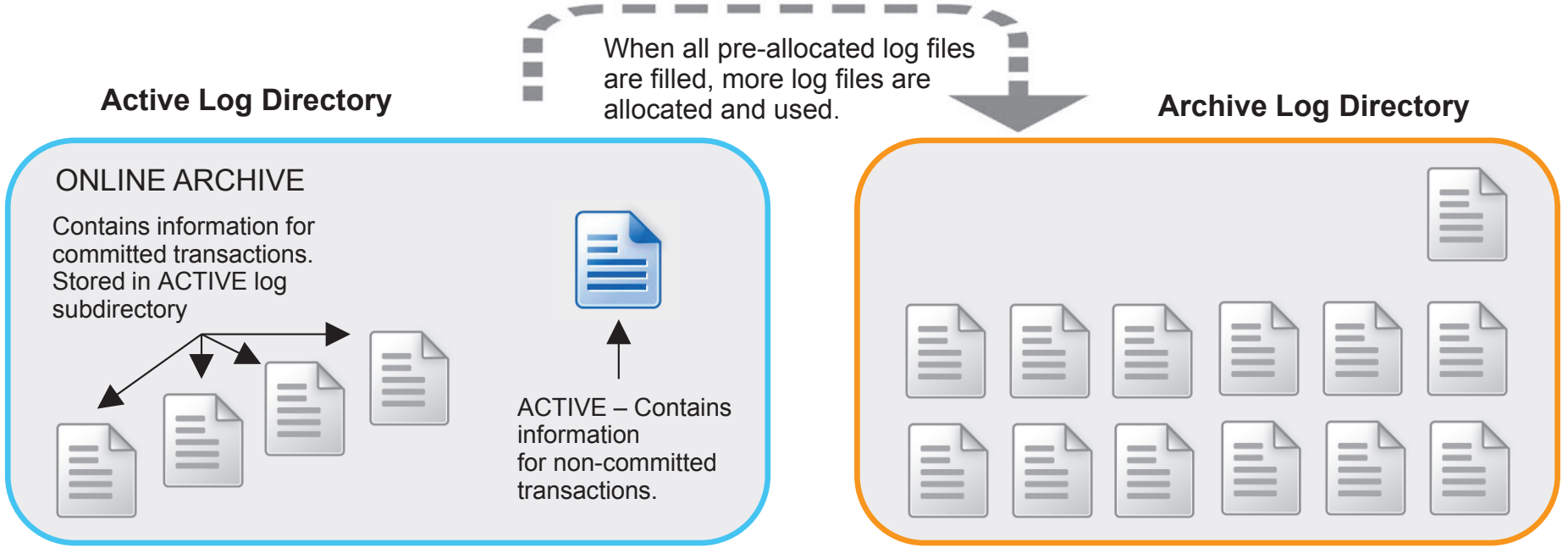
- ***Ring* of online primary logs provide version recovery in case of failure**
  - Secondary logs used when next primary log is not available
- **Default logging for DB2**
  - DB configuration parameters `logarchmeth1` and `logarchmeth2` are set to OFF
- **Only full, offline database backups are allowed**
  - Only ensures integrity of current transactions



Logging Types - Archive

- **Maintain a history of log files**
  - Enable with **LOGARCHMETH1** DB configuration parameter
    - **LOGRETAIN** and **USEREXIT** have been discontinued in DB2 10. They have been replaced with **LOGARCHMETH1**.
  - Allows roll-forward recovery or online backup
- **Logs can be archived externally when no longer active to avoid exhaustion of log directory**
- **As of DB2 10.1, archived log files can be compressed**

New in DB2 10.1



Infinite Logging

- **Issue with limited number of logs**
  - A long running transaction can exhaust logs allocation, even after secondary log files are allocated
  - The number of primary and secondary log files must comply:
    - If **`logsecond`** has a value of -1, **`logprimary`** **<= 256**.
    - If **`logsecond`** does not have a value of -1, **(`logprimary` + `logsecond`) <= 256**.

- **Solution: Infinite Logging**
- **No limit on the size or the number of in-flight transactions running**
- **Enabled by setting `logsecond` to -1**
- **Database must be configured to use archive logging**
  - Can hinder performance for rollback and crash recovery
- **Other control parameters**
  - **`num_log_span`**: number of log files an active transaction can span
  - **`max_log`**: percentage of the primary log space that a transaction can consume

DB2CKLOG - DB2 CHECK LOG

- **Check the validity of archive log files**
  - Determine whether the log files can be used during roll-forward recovery
  - A single archive log file or a range of archive log files can be checked

```
DB2CKLOG log_num ARCHLOGPATH path
DB2CKLOG log_num to log_num2
```

Validating a range of logs:

```
$ db2cklog 3 to 5

                    D B 2 C K L O G

                DB2 Check Log File tool

...

"db2cklog": Finished processing log file "S0000003.LOG". Return code: "0".

...

"db2cklog": Finished processing log file "S0000004.LOG". Return code: "0".

...

"db2cklog": Finished processing log file "S0000005.LOG". Return code: "0".
```

Successful Validation

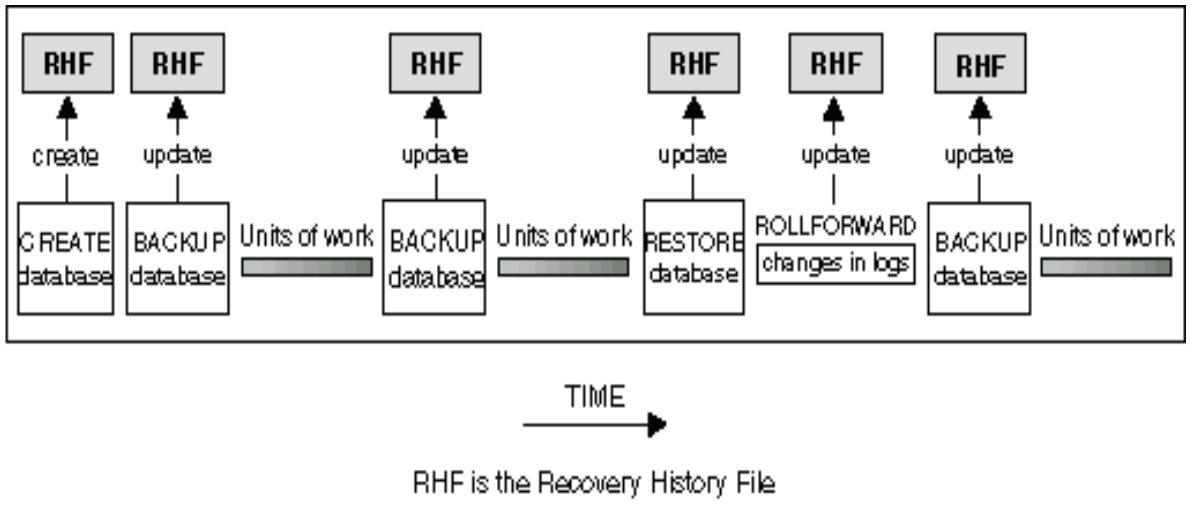
Recovery History File

- **Purpose**
  - In addition to recovery log files, the recovery history file is also created automatically when a database is created.
  - Used to recover all or part of a database to a point in time using the summarized backup information in this file
- **View**
  - To see the entries in the recovery history file:

```
LIST HISTORY
```

OR

```
SELECT * FROM SYSIBMADM.DB_HISTORY
```



RHF is the Recovery History File

- **Pruning**
  - Recovery history file cannot be directly modified, but entries that are no longer relevant can be pruned from the file using command `PRUNE HISTORY`

Database Backup

- **Copy of a database or table space**
  - User data
  - DB2 catalogs
  - All control files, e.g. buffer pool files, table space file, database configuration file

- **Backup modes:**
  - **Offline Backup**
    - Does not allow other applications or processes to access the database
    - Only option when using circular logging
  - **Online Backup**
    - Allows other applications or processes to access the database
    - Available to users during backup
    - Can backup to disk, tape, TSM and other storage vendors

Database Backup

- **Command-line:**

```
db2 backup database <db_name> <online> to <dest_path>
```

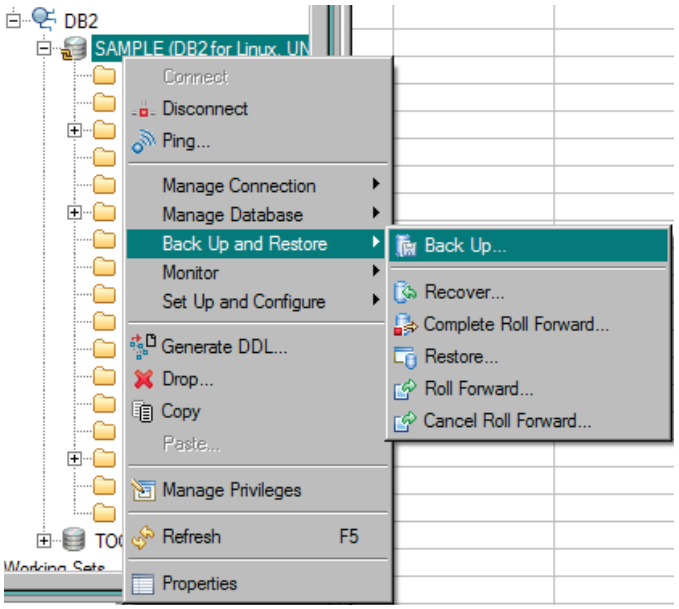
  - Online backup example

```
db2 backup database mydb online to /home/db2inst1/backups
```

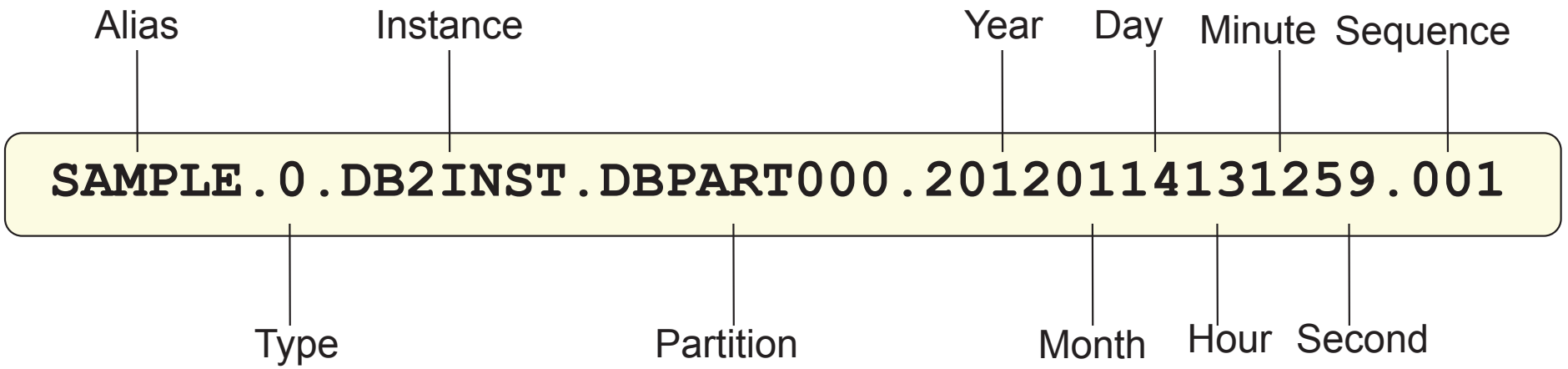
  - Offline backup example

```
db2 backup database mydb to /home/db2inst1/backups
```

- **IBM Data Studio:**
  1. Connect to target database
  2. Right click on database
  3. Select “Back Up and Restore” then “Back Up”



Database Backup – File Naming Convention



Backup Type:
- 0 = Full Backup
- 3 = Table space Backup
- 4 = backup image generated by the LOAD COPY TO command

Backing Up Partitioned Databases

- **There are four possible ways:**
  - Back up each database partition one at a time
  - Use the db2_all command
    - Backup specified partitions
  - Run a single system view – SSV (Recommended)
    - Backup some or all of the database partitions simultaneously
    - Database can be online or offline
    - Use the `ON DBPARTITIONNUMS` parameter to specify the partitions
    - Log files are included by default
    - Old backup images can be deleted
  - Use backup task assistant in IBM Data Studio

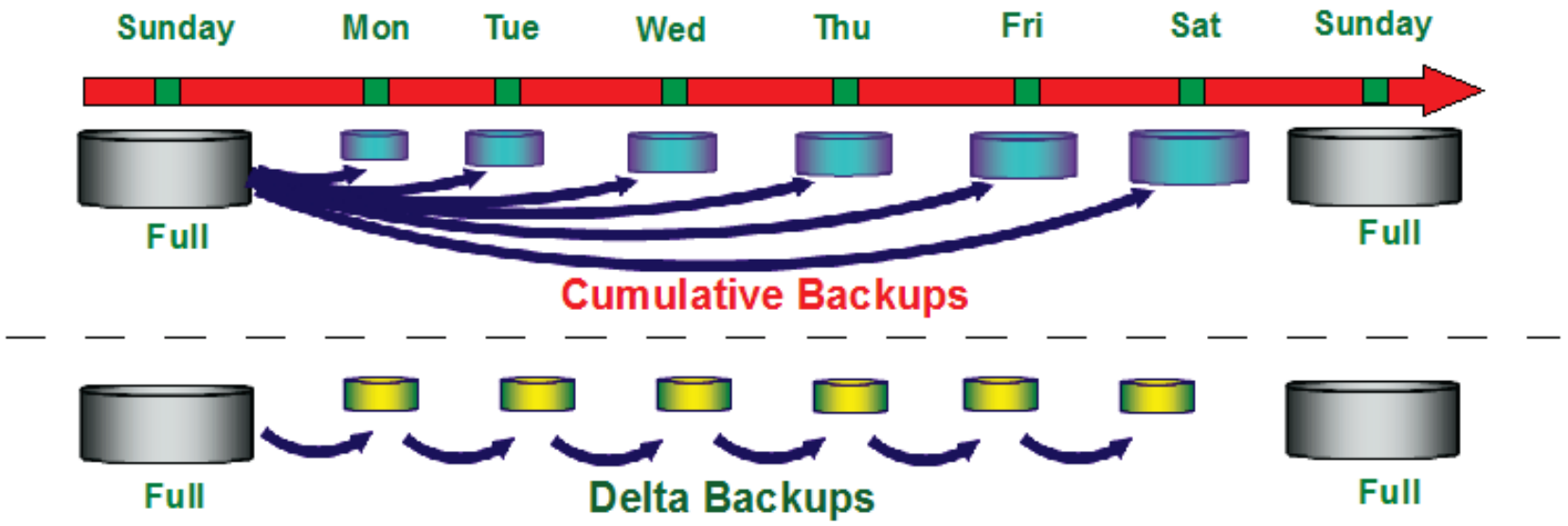
```
db2 backup database mydb1 ON ALL DBPARTITIONNUMS
db2 backup database mydb1 ON DBPARTITIONNUMS (2,4)
```

Table Space Backup

- Enables user to backup a subset of database
- Multiple table spaces can be specified
- Database must be using archive logging
- Table space backup can run in both online and offline backup
- Table space can be restored from either a database backup or table space backup of the given table space
- Use the keyword TABLESPACE to specify table spaces
- Supported in DB2 pureScale environment

```
db2 backup database mydb1 TABLESPACE (TBSP1) ONLINE to
/home/db2inst1/backup
```

Incremental Backups



- **Incremental (a.k.a. cumulative)** - Backup of all database data that has changed since the most recent, successful, full backup operation
- **Incremental Delta** - Backup of all database data that has changed since the last successful backup (full, incremental, or delta) operation.
- Requires `TRACKMOD` database configuration parameter set to **ON**
  - Supports both database and table space backups.
- Suitable for large databases, considerable savings by only backing up incremental changes.

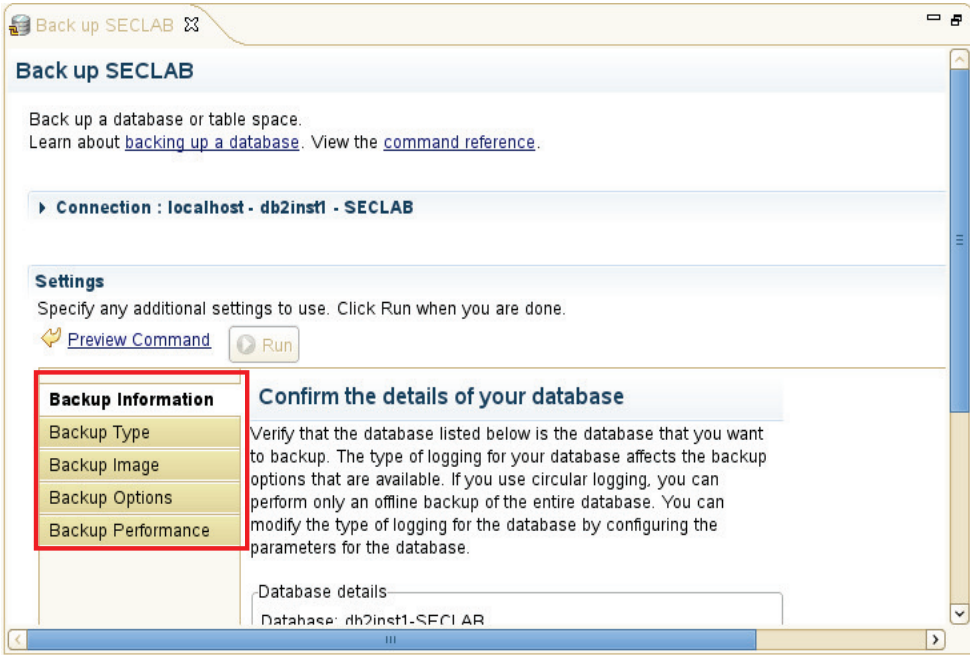
Database Backup – Compression

- **DB2 backups can be automatically compressed**
  - 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
- **E.g.**

```
db2 backup database DS2 to /home/db2inst1/backups
compress
```

Automatic Database Backup

- **Simplifies database backup management tasks for the DBA**
- **Ensures that the database is backed up both properly and regularly**
- **To configure automatic backup**
  - **DB configuration parameters**
    - AUTO_DB_BACKUP
    - AUTO_MAINT
  - **Graphical user interface tools**
    - Data Studio's "Backup Task
    - Assistant"
  - **System stored procedure**
    - AUTOMAINT_SET_POLICY



Optimizing Backup Performance

- **During a backup operation, DB2 automatically chooses an optimal value for:**
  - **`PARALLELISM` *`n`***
    - Number of table spaces backed up in parallel
  - **`WITH` *`num_buffers`* `BUFFERS`**
    - Number of buffers used
    - Use at least **twice as many buffers** as backup targets (or sessions) to ensure that the backup target devices do not have to wait for data.
  - **`Buffer` *`buffer-size`***
    - Backup buffer size
- **Allocate more memory to backup utility by increasing utility heap size (`UTIL_HEAP_SZ`) configuration parameter.**
- **Backup subset of data where possible:**
  - Table space backups
  - Incremental backups
- **Use multiple target devices**

DB2CKBKP – Check Backup

- **This utility can be used to test the integrity of a backup image**
  - determine whether the image can be restored.
  - display the meta-data stored in the backup header.

```
$ db2ckbkp -h SAMPLE.0.moba.NODE0000.CATN0000.20041008013428.001

=====================
MEDIA HEADER REACHED:
=====================
        Server Database Name           -- SAMPLE
        Server Database Alias          -- SAMPLE
        Client Database Alias          -- SAMPLE
        Timestamp                      -- 20041008013428
        Database Partition Number      -- 0
        Instance                       -- moba
        Sequence Number                -- 1
        Release ID                     -- A00
        Database Seed                  -- 92DBF20F
        DB Comment's Codepage (Volume)-- 0
        DB Comment (Volume)            --
        DB Comment's Codepage (System)-- 0
        DB Comment (System)            --
        Authentication Value           -- 255
        Backup Mode                    -- 1
        Includes Logs                  -- 1
        Compression                    -- 0
 ・・・（略）・・・
```

This backup is an online backup with INCLUDE LOGS option
0: Not included in the log file
1: contains log file

Backup is not compressed
0: not compressed
1: compressed

Database Recovery

- **Recovery is the rebuilding of a database or table space after a problem such as media or storage failure, power interruption, or application failure.**
- **Types of Recovery**
  - **Crash or restart recovery**
    - Protects the database from being left inconsistent
  - **Version recovery**
    - Restores a snapshot of the database
  - **Roll forward recovery**
    - Extends version recovery by using full database and table space backup in conjunction with the database log files
- **Crash recovery and Version recovery are enabled in DB2 by default**

DB2 Restore Utility

- **Restores database or table space from a previously taken backup**
- **Invoked using:**
  - The **Restore Database** command
  - The **db2Restore** API
  - IBM Data Studio **Restore** task assistant
- **`TAKEN AT` - Specify the time stamp of the database backup image**
- **Without prompting – Overrides any warnings**

– **Example:**

– `SAMPLE.0.DB2INST.DBPART000.20120114131259.001`

Table Space Restore Operation

- **Table space restore occurs in offline and online mode**
  - Other table spaces can be used concurrently while restore in progress
- **Restored table space is in Roll Forward Pending state**
  - can be either rolled forward to **End of Logs** or a **Point In Time**.
- **Minimum recovery time can be checked using**
  - `db2 list tablespaces show detail`
- **User table space must be in line with catalog table space**
  - E.g.: if catalog indicates table T1 exists in table space TSP1, table T1 must exist in the TSP1 table space, otherwise database becomes inconsistent
- **Recommended to take a table space backup after restore to a Point In Time**
  - Transactions that come after the point in time are lost
  - backup required as new point of reference for future recoveries

Incremental Restore

- **Restore a database with incremental backup images**
- **AUTOMATIC (recommended) - All required backup images will be applied automatically by restore utility**
- **MANUAL – User applies the required backups manually**
  - **db2ckrst** can provide the sequence for applying backups
- **ABORT - aborts an in-progress manual cumulative restore**

```
RESTORE DATABASE sample INCREMENTAL AUTOMATIC FROM /db2backup/dir1;

ROLLFORWARD DATABASE sample TO END OF LOGS AND COMPLETE;
```

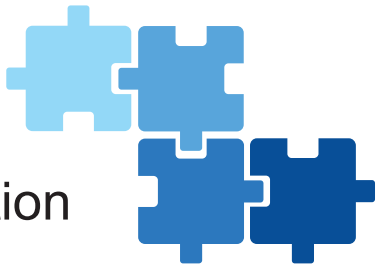
Redirected Restore

- **A redirected restore operation is performed when:**
  - Restore a backup image to a machine that is different than the source machine
  - Restore table space containers into a different physical location
  - One or more containers is inaccessible
  - Redefine the paths of a defined storage group
- **Restrictions**
  - Cannot use a redirected restore to move data from one operating system to another
  - Cannot create or drop a storage group during the restore process
  - Cannot modify storage group paths during a table space restore process
- **Two-step database restore process with an intervening table space container definition step**

  1) Issue `RESTORE DATABASE` command with `REDIRECT` option

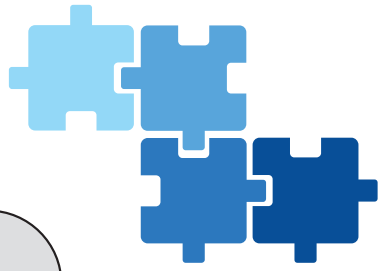
  2) Take one of the following steps:
  - `SET TABLESPACES CONTAINERS`
  - `SET STOGROUP PATHS`

  3) Issue `RESTORE DATABASE` command with `CONTINUE` option



Restore with Transport

- **The `RESTORE DATABASE` command can transport a set of table spaces or schemas**
  - The **`TRANSPORT`** option of the **`RESTORE`** command copies objects from a backup image to a database
  - Database objects are re-created to reference the new database and the data is restored
- **Transporting a database schema involves:**
  - Backup valid transportable sets from the source database
  - Restore transportable sets on the target db using the TRANSPORT option
  - Optionally, you can redirect the container paths for the table spaces being transported
- **Restrictions**
  - Database Schema must be transported in its entirety.
  - Source and Target should not share the same schema names



```
db2 restore db originaldb tablespace (mydata1)
schema(schema1,schema2) from <Media_Target_clause> taken
at <date-time> transport into targetdb redirect

db2 list tablespaces db2 set tablespace containers for
<tablespace ID for mydata1> using (path '/db2DB/data1')

db2 restore db originaldb continue
```

DB2 Recover Utility

- **DB2 Recover Utility:**
  - Performs restore and rollforward operations to recover a database to a specified time, based on information found in the recovery history file or to the end of logs
  - Can be issued for an incomplete recover operation that ended during the rollforward phase, the recover utility attempts to continue the previous recover operation without redoing the restore phase
  - Automatically establishes connection to the database
- **In IBM® Data Studio Version 3.1 or later, you can use the task assistant for recovering databases**
- **To invoke the recover utility use**
  - RECOVER DATABASE command
  - db2Recover application programming interface (API)
- **Restrictions**
  - Do not support Tablespace & Incremental restore operations
  - If it is interrupted during the restore phase, it cannot be continued. RECOVER command has to be re-issued

```
db2 RECOVER DATABASE SAMPLE TO 2001-12-31-04.00.00
db2 RECOVER DATABASE SAMPLE to END OF LOGS
```

IBM DB2 Advanced Recovery Feature

*New – Buy all 3 products in one bundle

DB2 Merge Backup

- Improve speed of your backup & recovery processes
- Minimize application impact

DB2 Recovery Expert

- Recover faster with greater granularity while protecting your critical business data
- Eliminate data errors before they compound into costly business mistakes
- Track and report data changes in response to auditing requests

Optim High Performance Unload

- Extract large amounts of data quickly and with minimal impact on system productivity
- Perform full data and system migrations from one DB2 instance to another

Backup

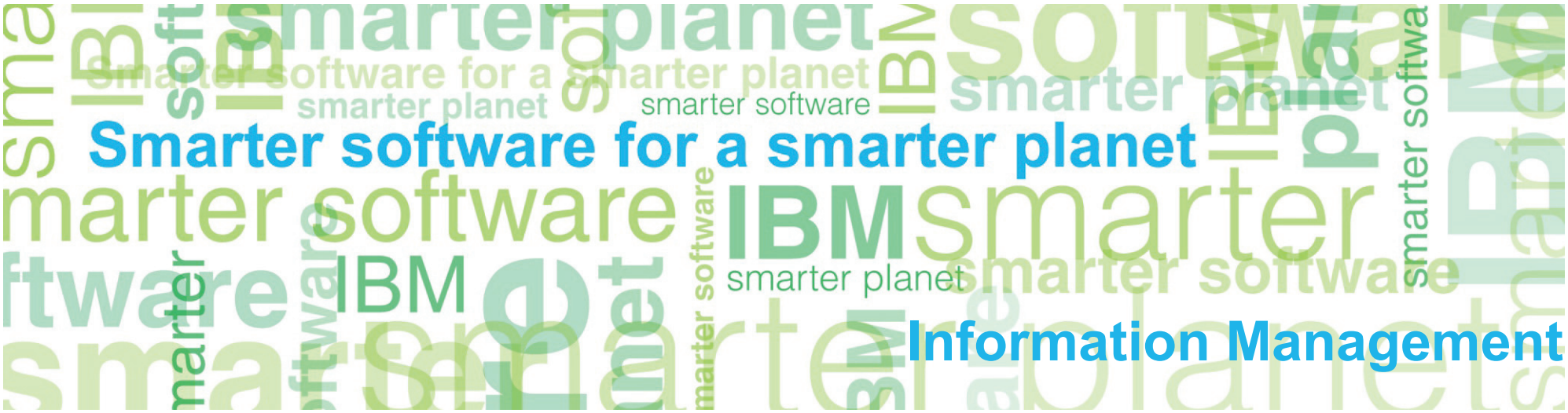
Recover

Unload

IBM DB2 Advanced Recovery Feature can be purchased separately and used with various DB2 Editions (except DB2 Express-C)

All products support DB2 10.5 for LUW

Storage Design and Multi-temperature Storage

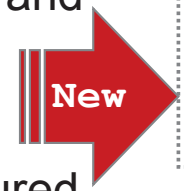


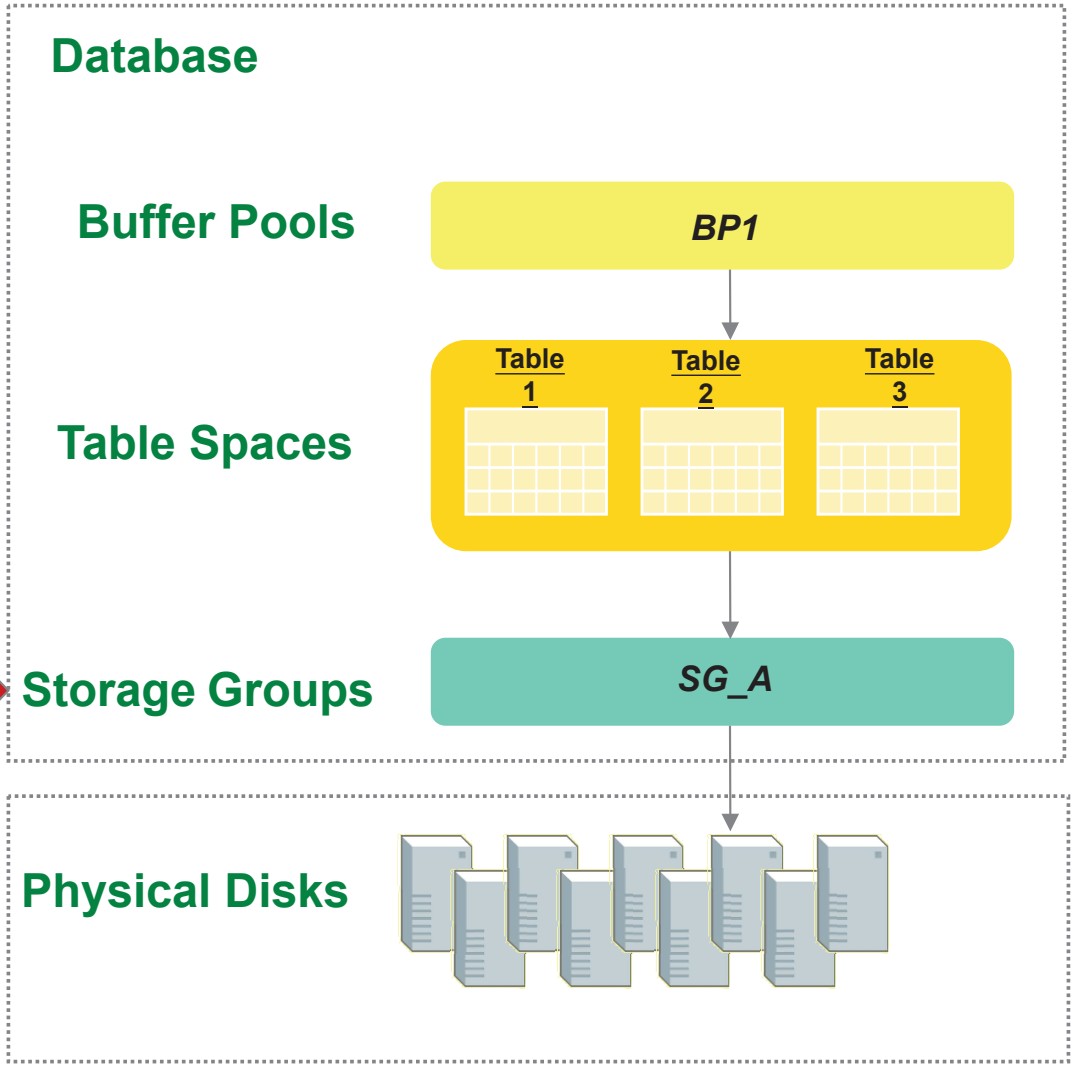
Agenda

- DB2 Storage Model
- Auto-resize Feature
- Automatic Storage
- Multi-Temperature Data Management
- WLM Integration
- Summary

DB2 Storage Model

- **Database**
  - Contains a set of objects used to store, manage, and access data
- **Buffer Pool**
  - Area of main memory for the purpose of caching data as it is read from disk
- **Table Space**
  - Logical space used to store data objects such as tables and indexes
- **Storage Group**
  - Set of storage paths configured to represent different classes of storage in the database system, where table spaces are stored
- **Physical Disk**
  - Physical location used to store data





Database Managed Space – Auto-Resize Feature

- **By default, the auto-resize feature is not enabled for DMS table spaces**
- **To enable the auto-resize feature, specify AUTORESIZE YES clause**

```
CREATE TABLESPACE DMST1 MANAGED BY DATABASE USING
(FILE '/db2files/DMS1' 10 M) AUTORESIZE YES
```

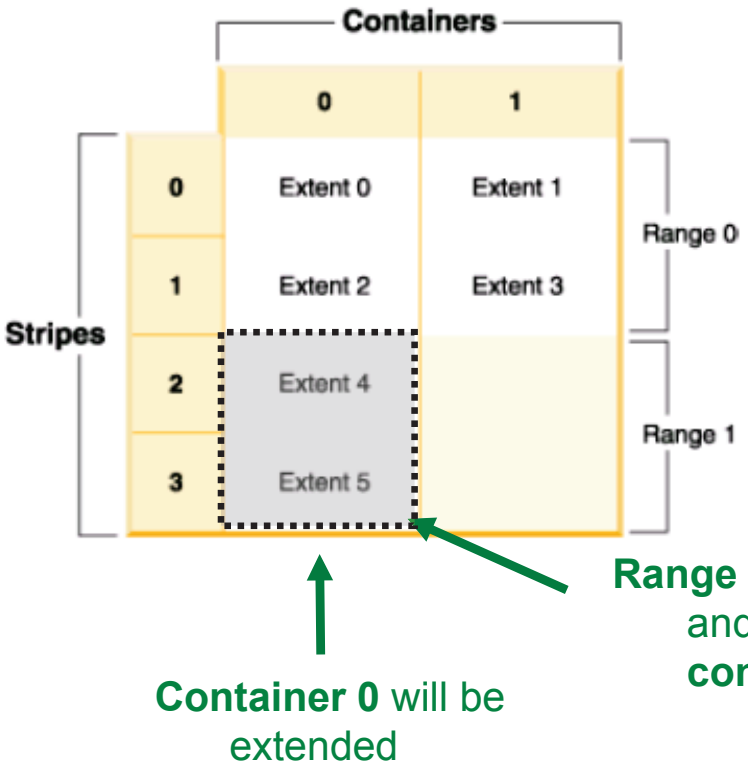
```
ALTER TABLESPACE DMTS1 AUTORESIZE YES
```

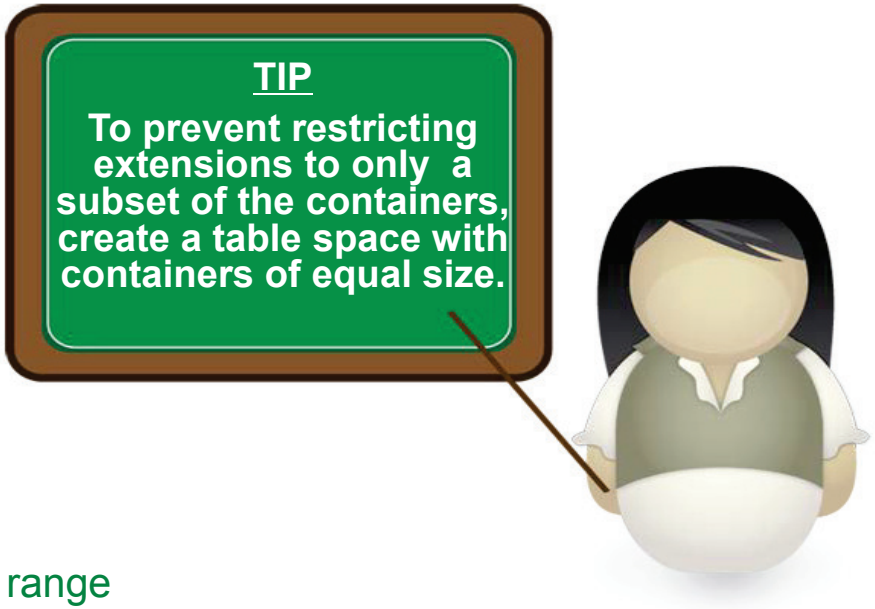
- **Options to be used with AUTORESIZE:**
  - **MAXSIZE** defines the maximum size for the table space
  - **INCREASESIZE** defines the amount of space to increase the table space when there are no free more free extents available

```
CREATE TABLESPACE DMS1 MANAGED BY DATABASE USING
(FILE '/db2files/DMS1' 50 M)
AUTORESIZE YES
INCREASESIZE 50 PERCENT
MAXSIZE 1G
```

Database Managed Space – Auto-Resize Feature

- **Determining how to extend table space**
  - DB2 decides which containers can be extended so that no rebalancing occurs
  - Only containers present within the last range of the table space map are extended
  - All those containers are extended by an equal amount



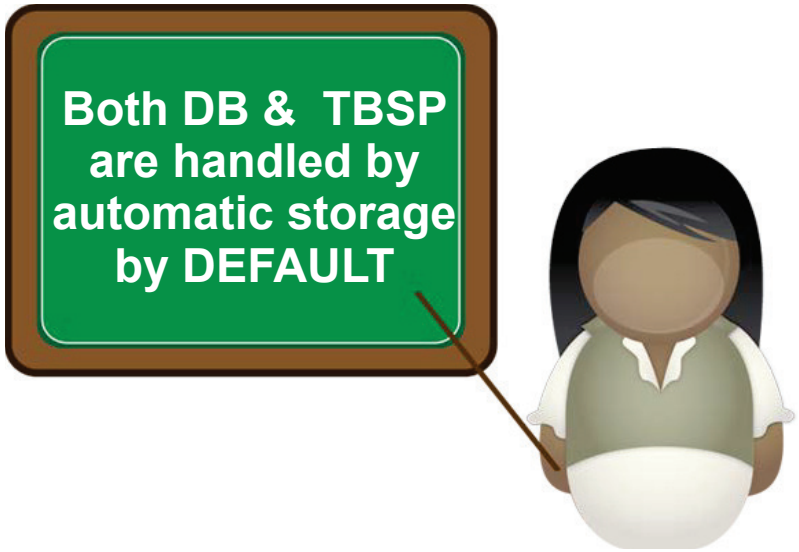


Automatic Storage

- **Highlights**
  - Low maintenance, containers are created and extended as needed
  - Creates a **DMS** table space for **regular/large table spaces**
  - Creates a **SMS** table space for **temporary table spaces**
  - **Leverage the new storage groups – a new layer of abstraction between logical (table spaces) and physical storage (containers) configured to represent different classes of storage available to your database**

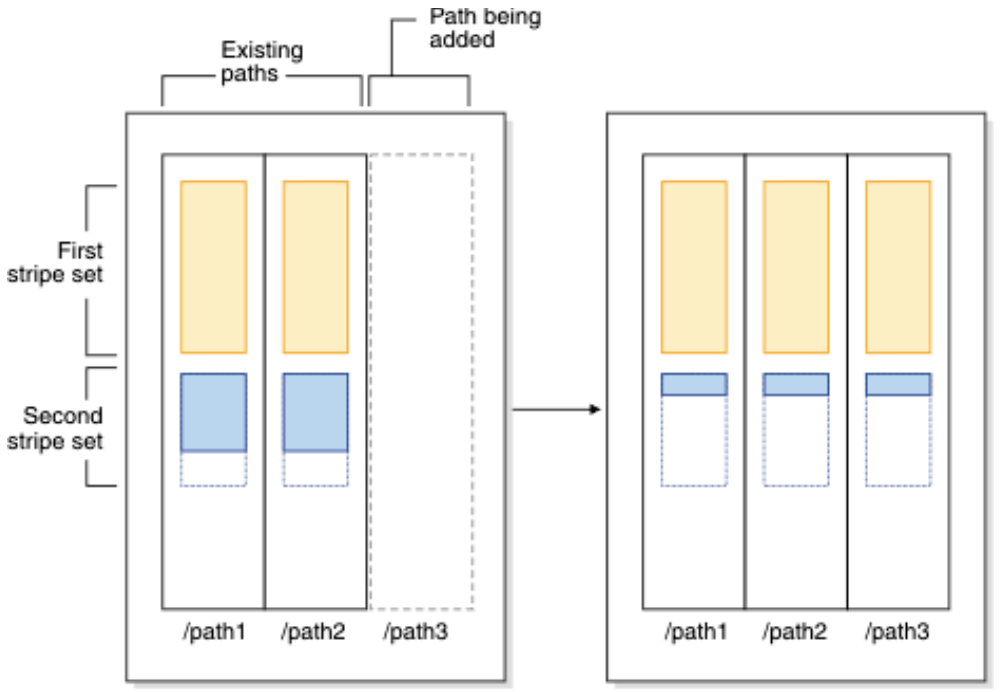
- **How to create**

```
CREATE DATABASE db1
CREATE TABLESPACE tbsp1
```



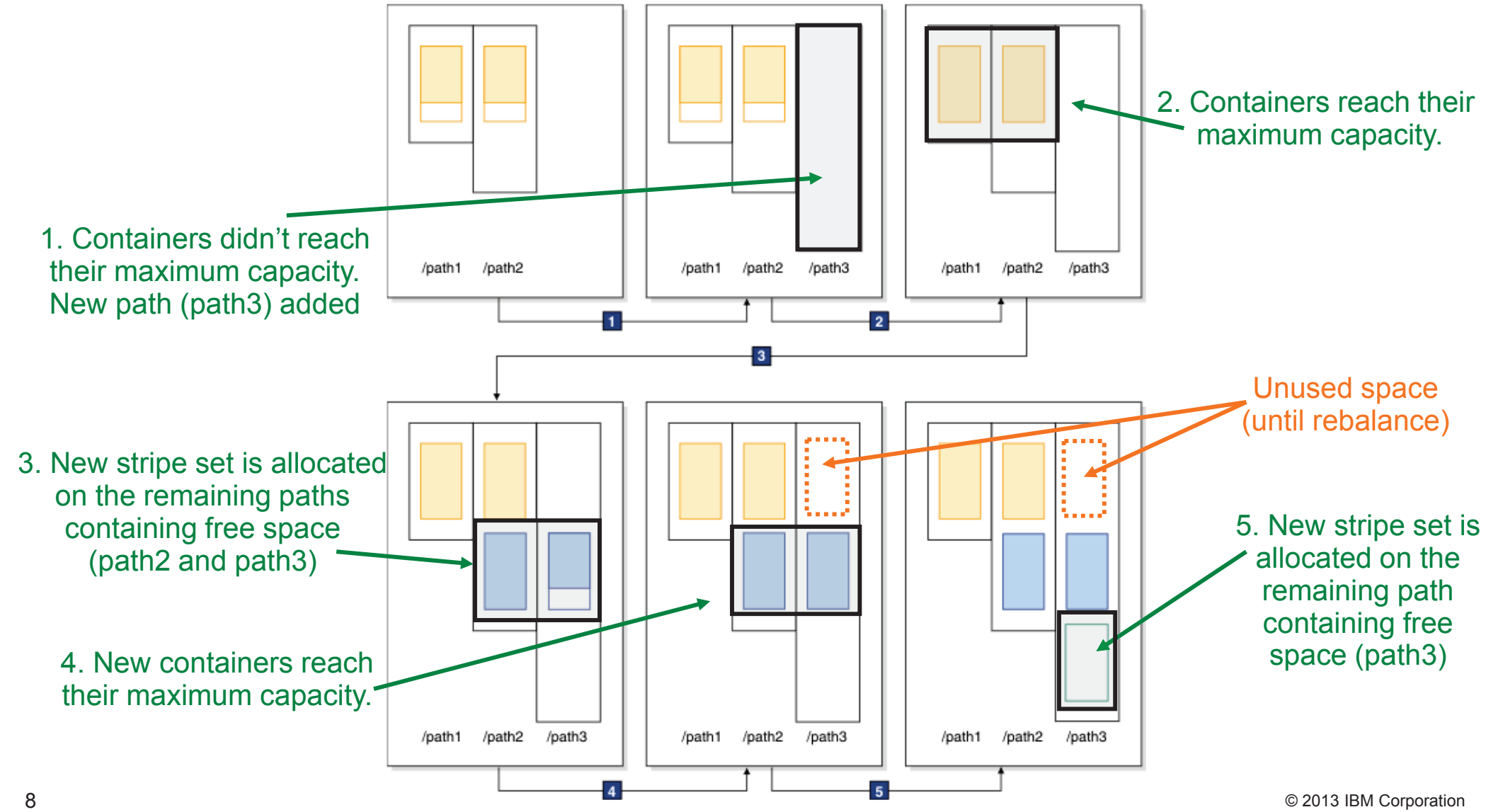
Automatic Storage

- **Storage for automatic table spaces is managed at the storage group level**
  - Storage is added to the storage group used by the table spaces
- **DB2 automatically creates and grows the underlying containers**
  - By extending existing containers
  - By adding a new stripe set of containers
- **For added storage, table spaces that already exist do not start consuming storage on the new paths immediately**
  - Rebalance Operation extends the existing stripe set and rebalances the data along all containers



Automatic Storage

- **Adding a new stripe set of containers (without REBALANCING)**



Automatic Storage – Storage Groups

```
CREATE DATABASE TESTDB7 ON 'C:','D:';
```

IBMSTOGROUP is created by DEFAULT and storage paths C: D: are added to it

- SYSCATSPACE, TEMPSPACE1, USERSPACE1 **created in IBMSTOGROUP**
- **Leveraging storage groups**

```
CREATE STOGROUP sg_1 ON '/u/ssd1', '/u/ssd2'
```

```
ALTER STOGROUP sg_1 ADD '/u/ssd3'
```

- **To specify another storage group as DEFAULT**

```
CREATE STOGROUP sg_1 ON 'E:' SET AS DEFAULT
ALTER STOGROUP sg_1 SET AS DEFAULT
```

  - NO impact on existing table spaces, used for newly created table spaces
  - Can move existing table spaces using **ALTER TABLESPACE**

- **Storage Path**
  - Can be added to 1+ storage group
  - Available across all database partitions

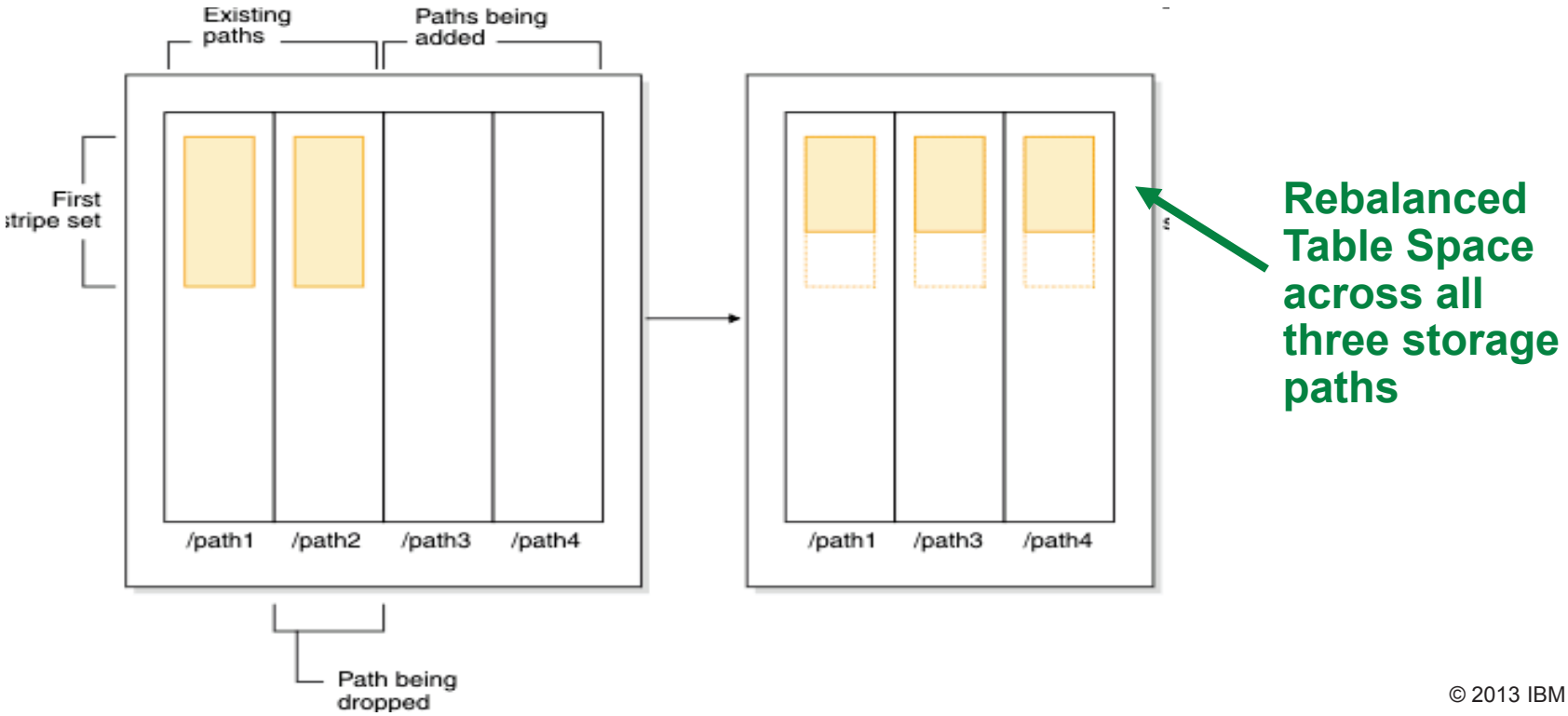
Add / Drop Storage Paths to Storage Group

- **Scenario: Add '/path3', '/path4', and drop '/path2' from storage group sg**

```
ALTER STOGROUP sg ADD '/path3', '/path4'
ALTER STOGROUP sg DROP '/path2'
```

- **Explicitly execute REBALANCE operation**
  - Otherwise, the new storage paths are used only when there is no existing space left

```
ALTER TABLESPACE tpsp1 REBALANCE
```



Dropping a Storage Group

Step 1: Find the table spaces using the storage group

```
SELECT TBSP_NAME, TBSP_CONTENT_TYPE FROM table (MON_GET_TABLESPACE(' ', -2))
WHERE TBSP_USING_AUTO_STORAGE = 1 AND STORAGE_GROUP_NAME = 'sto_group_old'
ORDER BY TBSP_ID
```

Step 2: Assign table spaces to another storage group

```
ALTER TABLESPACE tbsp_2009 USING STOGROUP STO_GROUP_NEW
```

Step 3: Monitor rebalance activity for the storage group to be dropped. Then drop the storage group

```
SELECT * from table (MON_GET_REBALANCE_STATUS( ' ', -2))
WHERE REBALANCER_SOURCE_STORAGE_GROUP_NAME = 'sto_group_old'

DROP STOGROUP STO_GROUP_OLD
```

Empty result state indicates move complete

Automatic Storage – Table Space

- **CREATE TABLESPACE command is extended to be able to specify a storage group**

```
CREATE TABLESPACE mytbspc USING STOGROUP sg_1
```

- **Dynamically inherits media attributes from the storage group**
  - using the **INHERIT** clause (***data tag, transfer rate, and overhead***)
  - Can **override** inherited media attributes from storage group

- **ALTER TABLESPACE** command is being extended to allow table spaces to change which storage group they are associated to

```
ALTER TABLESPACE mytbspc USING STOGROUP sg_2
```

  - By **default**, when changing the storage group, the table space inherits attributes from the target storage group by default
  - **DMS table spaces can be converted to Automatic Storage** in order to use storage groups

Convert DMS Table Spaces to use Automatic Storage

- **Use ALTER TABLESPACE to modify the DMS table space to use automatic storage**
  - 1. Specify the **new table space** to convert and indicate the **storage group** to be used

```
ALTER TABLESPACE tbsp_2011q1 MANAGED BY
AUTOMATIC STORAGE USING STOGROUP sg_2
```

  - 2. **Move the data** from the old containers to the storage paths **in the storage group**

```
ALTER TABLESPACE tbsp_2011q1 REBALANCE
```

    - Asynchronous operation ✅
    - Does not affect data availability ✅

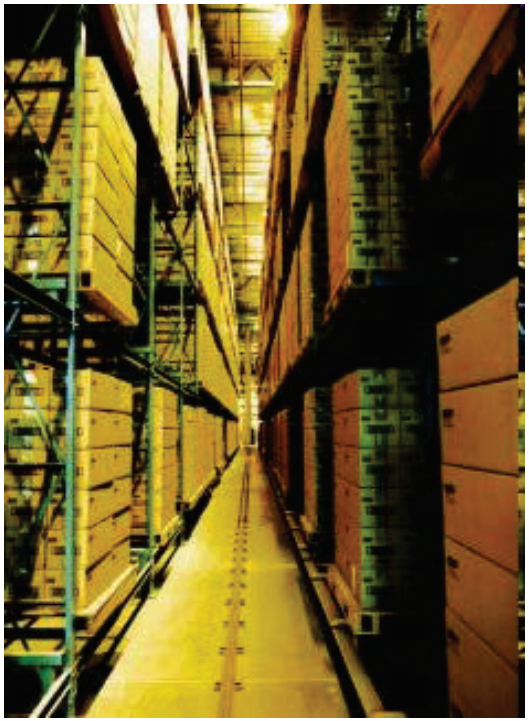
  - 3. Monitor the progress using the monitoring table function **MON_GET_REBALANCE_STATUS**

DB2 Storage – Summary

- **System Managed Space (SMS)**
  - Data stored in files, space is allocated on demand
  - Access to data controlled using standard I/O functions of the OS
  - Low maintenance, Ideal for small, personal databases
- **Database Managed Space (DMS)**
  - Data stored in files or on raw devices
  - Storage space pre-allocated in file system, typically contiguous physically
  - Ideal for performance-sensitive applications
  - Increased maintenance and monitoring
  - Auto-resize feature handles the table space by automatically extending existing containers
- **Automatic Storage Table Space**
  - Containers are created and extended as needed
  - Automatically handles resizing table spaces
  - Creates a DMS table space for regular/large table spaces
  - Creates a SMS table space for user or system temporary table spaces
  - Leverage the **Storage Groups** – a new layer of abstraction between logical (table spaces) and physical storage (containers)

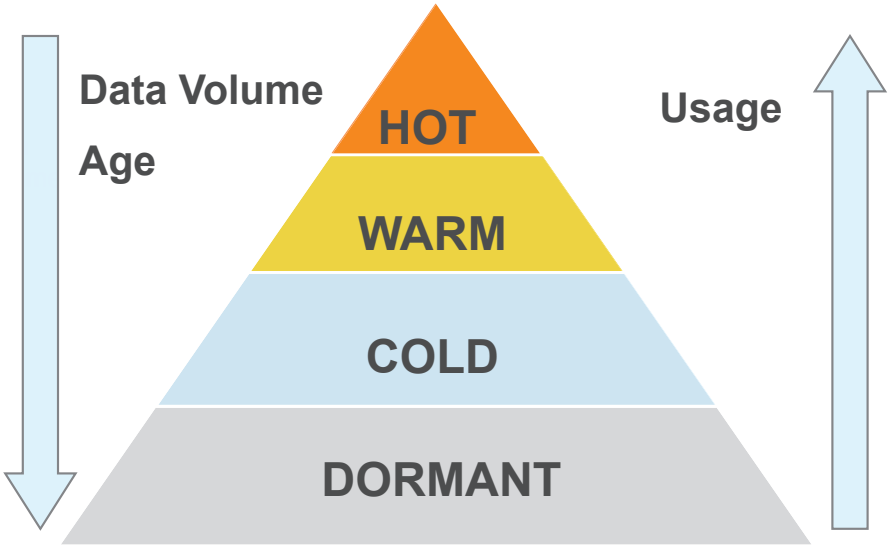
Multi-Temperature Storage – Business Motivation

- **There is an explosive growth in storage requirements due to**
  - technology shifts, unstructured data, and regulatory mandates
- **Data Warehouses retain large volumes of data for real-time business analytics**
  - 100+ TB of data stored
  - Require expensive high speed storage for BI + DSS
- **IT budgets cannot support the growth necessary without a change in storage practices.**
- **Difficult tradeoff between performance and cost**
  - Data in frequent use vs. less frequent/ historical data
  - Small subset of data benefits from high-speed storage
  - High-speed storage (e.g. SSD) **=** Higher costs
- **Access to various groups of data can have different performance and reliability requirements – Tiered storage is needed**
- Multi-temperature Storage **provides a solution**



Multi-Temperature Data Management

- **Provides the ability to assign priority to data (hot, warm, cool, cold) and dynamically assign it to different classes of storage**
  - Data temperature signifies priority of the data defined by business
  - Data temperature is inversely proportional to volume
    - Small portion of **hot** data vs. large portion of **warm**/**cold** data
- **Data can change temperature**
  - As data ages
  - As business criteria behind temperature changes



Sales data of this month = **most frequent**

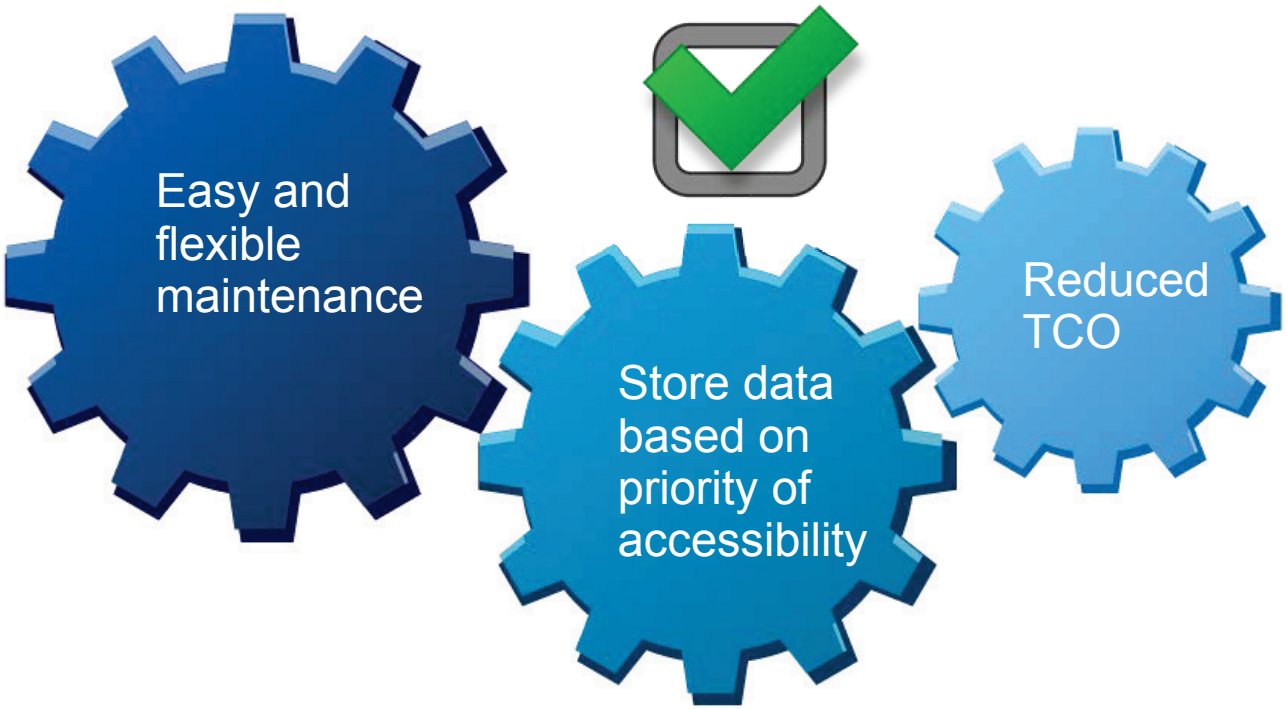
Sales data of this quarter = **less frequent**

Sales data of previous quarters = **rarely accessed**

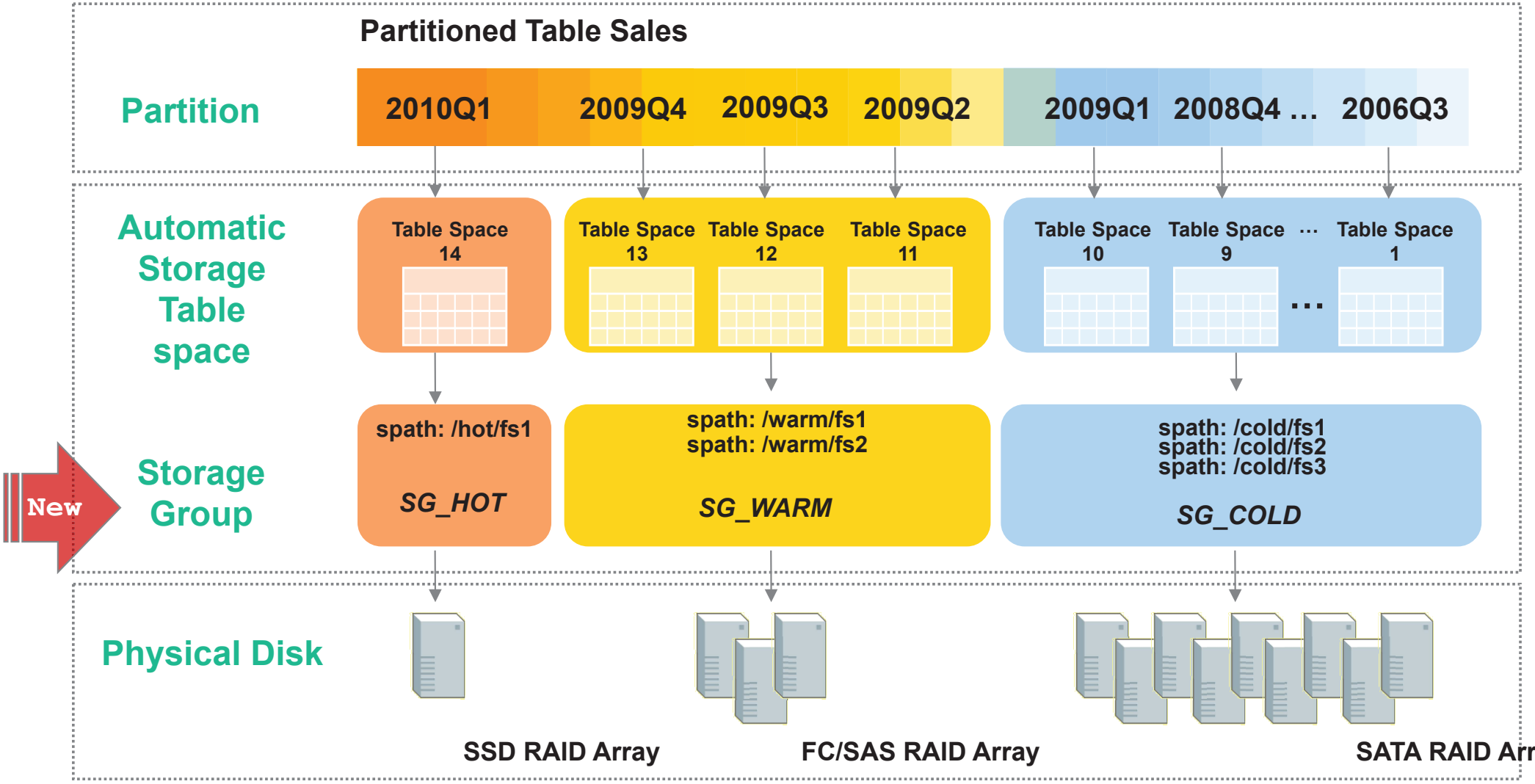
Sales data of past years = **historical data**

Storage Groups

- **Storage Groups allow the flexibility to implement Multi-temperature Data Management in Automatic Storage table spaces**
- **Different Storage Groups can represent different classes of storage**
  - **Hot** data assigned to storage groups with fast devices
  - **Warm** or **Cold** data assigned to slower devices
- **Easy maintenance when data ages and needs to be moved to a different storage class**



Leveraging Storage Groups

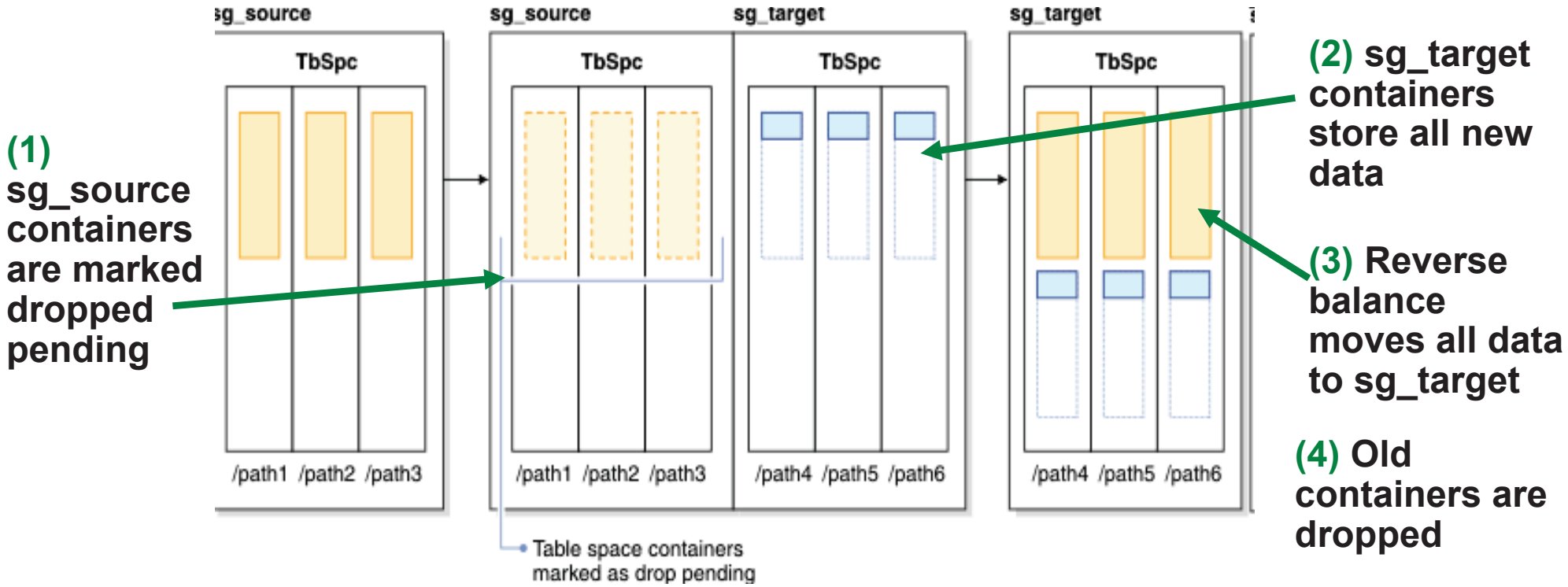


How to Change the Temperature of Your Data

- Use **ALTER TABLESPACE** to change the data temperature of table space

```
ALTER TABLESPACE tbSpc USING STOGROUP sg_target
```

- **Implicit REBALANCE** occurs when a table space moved between storage groups



- Monitor the progress of REBALANCE by using the new monitoring table function **MON_GET_REBALANCE_STATUS**

Multi-temperature Storage Integrates with DB2 WLM

- **Existing WLM perspectives are user-centric (who) and request-centric (what)**
- **Introducing a new perspective** "data-centric" **(where)**
  - New **data tag** attribute
    - For storage group or table space
    - Priority can be given to requests based on what data is accessed [Values 1 (high) – 9 (low)]
- **WLM work class and threshold DDL have been extended to support the new data tag attribute**
  - DB2 optimizer can provide an estimated list of data tags for data touched by a query at compile
    - The data tag can influence the initial placement of the activity into a service class
- *New* Data tag threshold DATATAGINSC **uses information that is available at runtime to remap an activity to a different service subclass**

Multi-temperature Storage – A Sample Scenario

- **GOAL: Reduce warehouse storage costs while meeting the desired Quality of Service requirements for access to last 3 quarters of data**
- **Step 1: Create two storage groups to reflect the 2 tiers of storage This would result in transfer rate, overhead, etc being programmatically computed at the storage group level.**

```
CREATE STOGROUP sg_hot ON '/ssd/path1', '/ssd/path2' DATA TAG 1
CREATE STOGROUP sg_warm ON '/hdd/path1', '/hdd/path2' DATA TAG 5
```

Data tags represent business priority of the data and is used by the optimizer

- **Step 2: Assign table spaces to storage groups**

```
CREATE TABLESPACE q1_2011_tbsp USING STOGROUP sg_warm
CREATE TABLESPACE q2_2011_tbsp USING STOGROUP sg_warm  DATA TAG 3
CREATE TABLESPACE q3_2011_tbsp USING STOGROUP sg_hot
```

Multi-temperature Storage – A Sample Scenario

… A New Quarter Begins

- **Create a new table space and change storage group for Q3 table space**
  - Q4 table space will reside on hot storage
  - Q3 data will be moved and rebalanced across slower storage

```
CREATE TABLESPACE q4_2011_tbsp USING STOGROUP sg_hot
ALTER TABLESPACE q3_2011_tbsp USING STOGROUP sg_warm DATA TAG 3
```

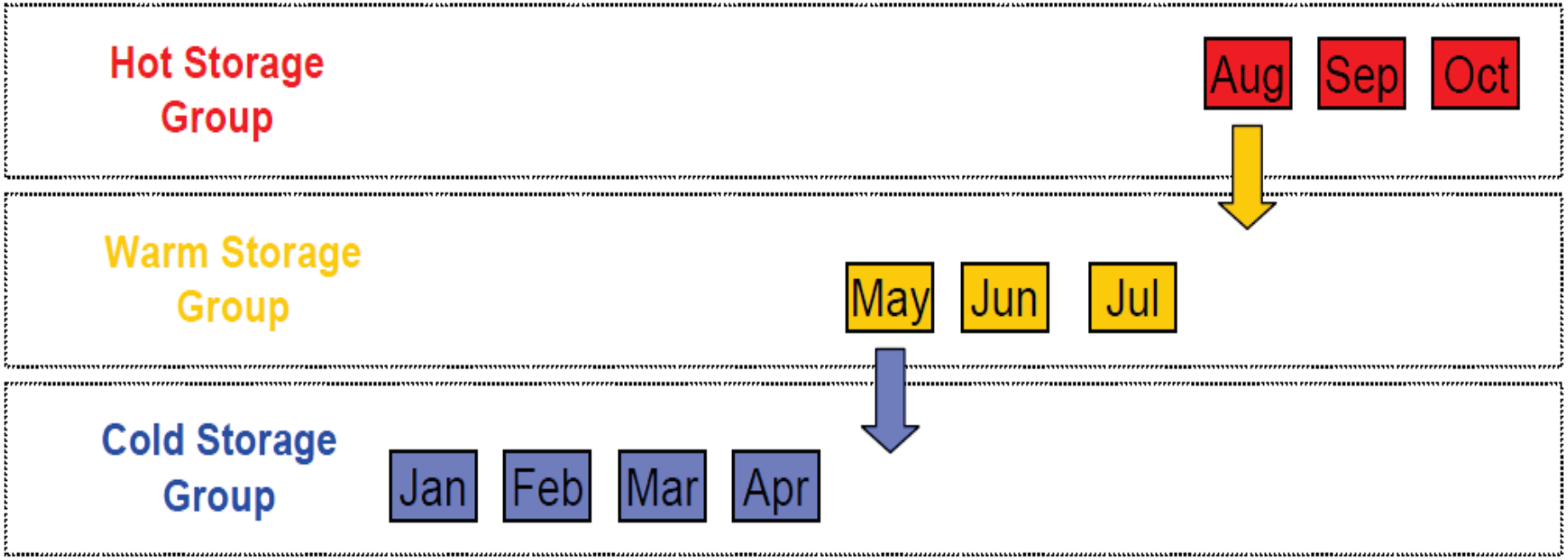
- **Data Tag changed to allow optimizer to consider the changed data priority**

```
ALTER TABLESPACE q2_2011_tbsp DATA TAG 5
```

- Only the most frequently accessed data resides on high-end expensive storage and meets the QoS requirements for that data access
- The bulk of the data resides on less expensive storage.
- Provides easy management by DBA's

Automate Data Aging Across Storage Tiers

- **Using Optim Configuration Manager you can define a data migration job for your multiple storage tiers**
  - Optimize the use of your storage by configuring a multi-temperature storage scheme
  - Data migration aging policies can be designed for one, some or all your partitioned tables with range partitions.



Oracle Storage Comparison

- **DB2**
  - Utilizes Automatic Storage and Table Partitioning to easily move table partitions between storage groups. (Free in Enterprise and AE Edition)
  - Simple, on-line operation to move data from one temperature storage to another
  - Utilizes standard OS file systems
  - Integrated with WLM to provide work load priority based on the data being accessed
- **Oracle**
  - Requires Oracle Partitioning (extra cost option for Enterprise Edition)
  - More script driven versus statement driven, not as simple as DB2 Multi-Temperature Storage
  - Requires Automatic Storage Management (ASM), does not use standard OS file systems
  - Oracle Database has no integration with Database Resource Manager and **NO** ability within it to prioritize queries based on data temperature

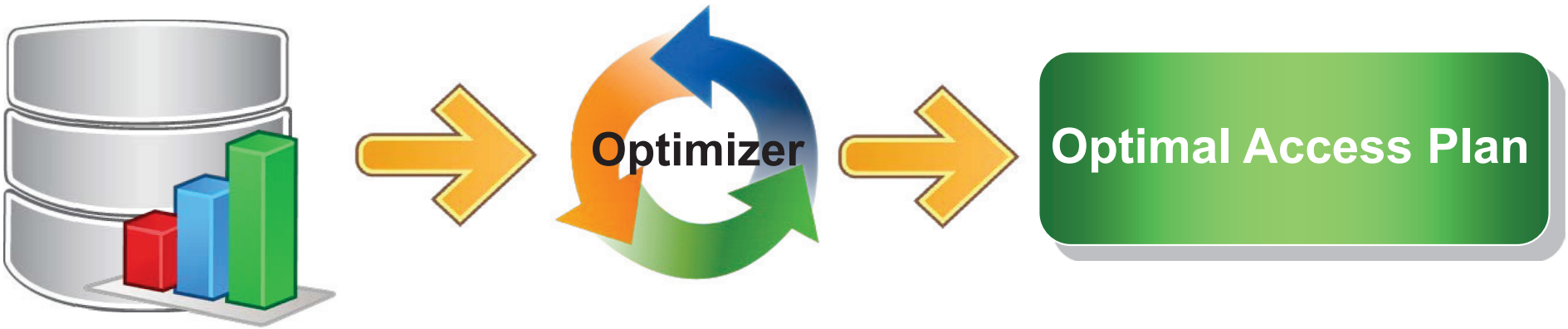
Summary

- **Each database must have a minimal set of storage areas that are used for storing system, user and temporary data.**
- **A database must contain at least three storage areas:**
  - A catalog area
  - One or more user areas
  - One or more temporary areas
- **User table spaces managed by system are deprecated**
  - You can still specify the SMS type for catalog table spaces and temporary table spaces
- **New logical grouping introduced in DB2 10.1 – Storage Groups**
- **Multi-temperature data management with storage groups**
  - **Improves manageability of storage**
    - Saves on storage costs while satisfying performance requirements
  - **Tiered storage system**
    - Degree of data access can correspond to the relative speed of the storage device
  - **Ease of use**
    - Storage groups as database objects are easy to implement and maintain
  - **Improve performance**
    - Fine tune the workload environment using data tag attribute for business priorities

Basic Maintenance & Autonomic Features

Statistics Collection

- DB2 Optimizer needs to intelligently determine the most efficient way of servicing a SQL query. Its decisions are heavily influenced by statistical information from DB objects.
- DB2 **RUNSTATS** command updates statistics that profile the physical characteristics of a database table, along with its associated indexes.
  - Use after many DML changes or a reorganization of a table
  - Statistics are stored in the catalog tables (E.g.: `SYSCAT.TABLES, SYSSTAT.INDEXES`)
- After running **RUNSTATS:**
  - **Static SQL**
    - Requires explicit REBIND after statistics are updated using RUNSTATS
  - **Dynamic SQL**
    - Statements are prepared and executed at run time. They will automatically use updated statistics



RUNSTATS Command

- Executing the basic RUNSTATS command

```
db2 RUNSTATS ON TABLE FOR DB2INST1.TABLE1 AND INDEXES ALL
```

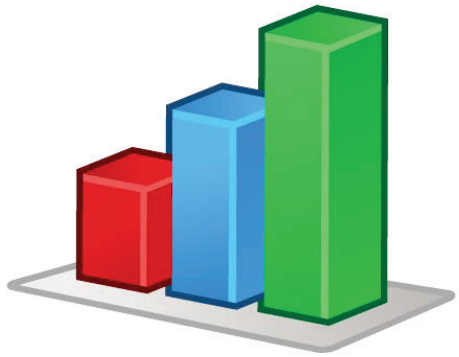
- Sample some data rather than evaluating the entire table

```
db2 RUNSTATS ON TABLE DB2INST1.TABLE1 AND INDEXES ALL TABLESAMPLE SYSTEM (10)
```

- Some situations when updating statistics would be beneficial
  - After data has been loaded into a table and appropriate indexes have been created
  - After creating a new index on a table
  - After a table has been reorganized with the REORG utility
  - After a table and its indexes have been significantly modified through update, insert, or delete operations

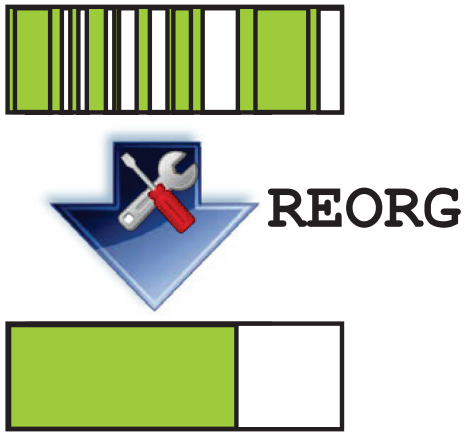
Guidelines for Collecting Statistics

- It's important to keep statistics updated in order to provide the optimizer with accurate information for access plan selection.
- Collecting statistics of very large tables is challenging and can affect workload performance of the system.
- Improving the performance of RUNSTATS
  - Collect basic statistics only for relevant columns. E.g.: columns used to join tables
  - Consider specifying only those columns for which data distribution statistics should be collected.
  - Use row-level or page-level sampling when running RUNSTATS
  - Use throttling option to limit the performance impact of RUNSTATS execution



Data Reorganization

- Over time, data can become fragmented resulting in
  - Increased size of tables/indexes
  - Degraded performance as more pages need to be read
- **Table reorganization**
  - Eliminates fragmentation of table data
  - Reduces number of read operations to access data
  - Reorganizes table data to match index
  - Reclaim wasted space
- **Index reorganization**
  - Rebuilds the index data into un-fragmented, physically contiguous pages
  - Reduces I/O costs because of fragmented leaf pages, badly clustered index (which affects sequential prefetching) and indexes with too many leaves



Data Reorganization

- **Types of Reorg**
  - **Classic** (**offline**) → Sort, build, replace, recreate all indexes
    (+) fastest, perfectly clustered tables and indexes
    (-) limited table access, less control (can't be paused), more space required
  - **In-place** (**online**) → Select *n* pages, vacate the range, fill the range, truncate the table
    (+) full table access, recoverable, less space required
    (-) slower, potentially high logging requirements, subsequent index reorg might be needed
- **How to determine if tables, indexes, or both, need to be reorganized?**
  - Run the REORGCHK command
  - It calculates statistics on the database to determine if REORG is required.

```
db2 REORGCHK CURRENT STATISTICS ON TABLE db2inst1.orders
```

- **Options for REORGCHK Command:**

```
CURRENT STATISTICS or UPDATE STATISTICS
ON SCHEMA schemaname
or ON TABLE ALL or SYSTEM or USER tablename
```

- In addition to the REORGCHK command, you can use these stored procedures:
  - `SYSPROC.REORGCHK_TB_STATS` for tables
  - `SYSPROC.REORGCHK_IX_STATS` for indexes

Data Reorganization

- Perform reorganization of objects
  - Re-org a **Table**

indicates an online reorg

```
db2 REORG TABLE TABLE1 INDEX INDEX_TS1 INPLACE
```

  - Re-org an **Index**

```
db2 REORG INDEXES ALL FOR TABLE TABLE1
```

- **Optional Step:** monitor progress in case of an online reorg
- **Post-tasks**:
  - Update statistics on reorganized objects
  - Rebind applications that access reorganized objects

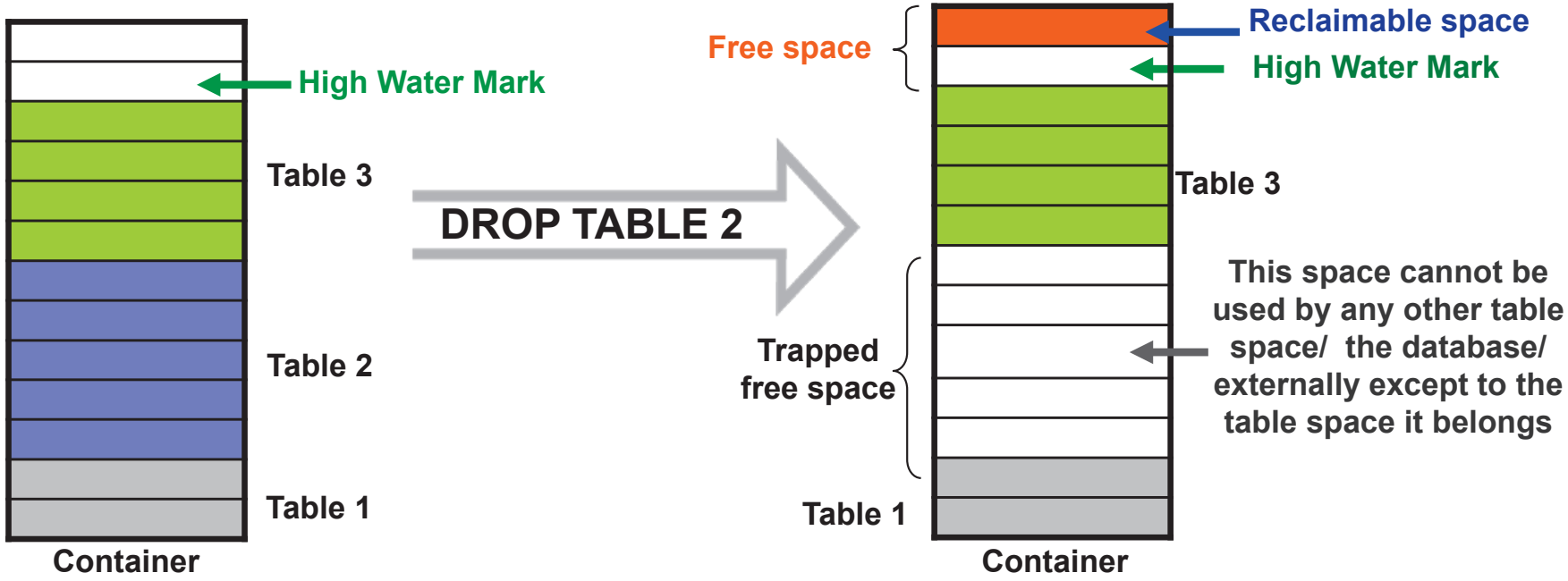
Table space and Storage Management

	System Managed Table space	Database Managed Table space	Automatic Storage Managed Table space
Initial container definition and location	Requires that containers be defined as a directory name.	▪Requires that containers be defined as files or devices. ▪Must specify the initial size for each container.	▪No list of containers specified at creation time. ▪The database manager automatically creates containers on the storage paths specified at creation. ▪Data is striped evenly across all paths.
Initial allocation of space	Done as needed. The file system controls the allocation of storage.	Done when the table space is created. Extents more likely to be contiguous than SMS.	▪Space is allocated on table space creation; can specify the initial size for table space; except temporary table spaces
Changes to table space containers	No changes once created, other than to add containers for new data partitions.	▪Containers can be extended or added, reduced or dropped.	▪Containers can be reduced or dropped. ▪Data can be rebalanced across containers when change occurs.
Handling of demands for increased storage	Containers will grow until they reach capacity imposed by the file system. The table space is considered to be full when any one container reaches it's maximum capacity.	Containers can be extended beyond the initially-allocated size up to constraints imposed by file system.	▪Containers are extended automatically up to constraints imposed by file system. ▪If storage paths are added, containers are extended/ created automatically.

Table Space Maintenance: Optimizing Table Space Usage

Since DB2 9.7

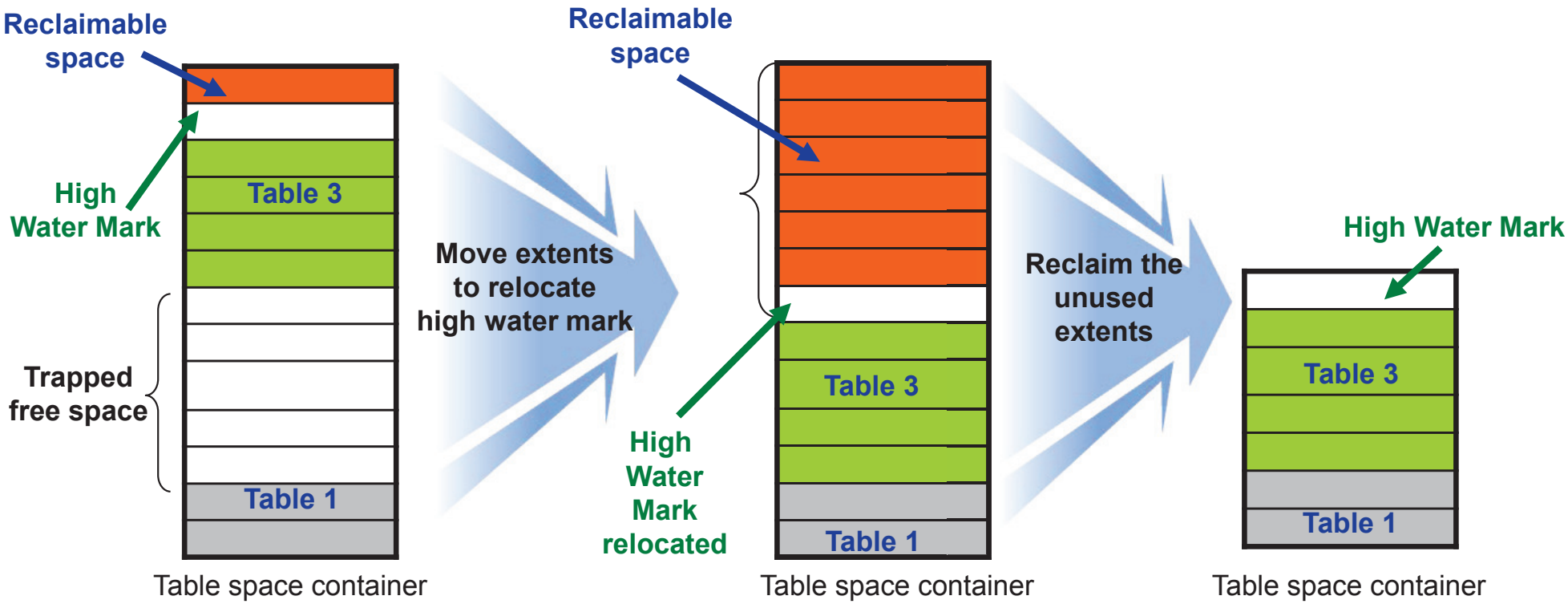
- When an object is dropped, space is freed within the table space.
  - The new free space cannot be shared among different table spaces
  - Only unused extents **above the high water mark** are reclaimable
- **High water mark** refers to the page number of the first page in the extent following the last allocated extent.



- How can the trapped free space below the high water mark be reclaimed?
  - To reclaim unused space, the high water mark needs to be lowered.

Optimizing Table Space Usage

- Re-arrange the extents to lower the high water mark, then reclaim the unused extents
  - Use **ALTER TABLESPACE** command
    - Automatic storage table spaces ➔ **REDUCE** option
    - DMS table spaces ➔ **LOW WATER MARK** + **REDUCE** options
    - SMS table spaces ➔ **Not Available**



Reduce Size: Automatic Storage Table Space

- The DB manager attempts to lower the *high water mark and* reduce size of table space containers
  - Empty containers are dropped or resized, extents are moved to free space
  - Size can be reduced by a specific amount of pages, bytes (K,M,G),% or max. value

Syntax

```
ALTER TABLESPACE <tsname> REDUCE <size>
```

Example 1

```
ALTER TABLESPACE tsname REDUCE MAX
```

Example 2

```
ALTER TABLESPACE tsname REDUCE 25 PERCENT
```

- Example 1, the keyword MAX suggests to move the maximum number of extents possible to the beginning of the table space.
- Example 2, attempts to reduce the size of the table space TS1 to 75% of it's original size, if possible.

Reduce Size: DMS Table Space

- To reclaim unused storage, explicitly lower High Water Mark:
  1. Re-arrange extents in the table to make use of the free extents lower in the table space.
  2. Reduce the size of the containers in the table space by a specified amount.

```
ALTER TABLESPACE <tsname> LOWER HIGH WATER MARK
```

Example

Alternatively, specify a container name

```
ALTER TABLESPACE ts LOWER HIGH WATER MARK
ALTER TABLESPACE ts REDUCE (ALL CONTAINERS 5 M)
```

- To reclaim unused index space on tables that reside in DMS table spaces
  - New online index reorg functionality **New in DB2 v10.1**

```
REORG INDEX ALL FOR TABLE tbs RECLAIM EXTENTS
```

- Moves index pages around to create empty extents
- Frees pages from exclusive use by the index object

Determining Free Space to Reclaim

- **Check Table Space for Free Space**

```
db2 "select varchar(tbsp_name, 15) as tbsp_name, tbsp_type,
   reclaimable_space_enabled, tbsp_free_pages from
   table(mon_get_tablespace('EXTENTREMAP',-2)) as t"
```

TBSP_NAME	TBSP_TYPE	RECLAIMABLE_SPACE_ENABLED	TBSP_FREE_PAGES
EXTENTREMAP	DMS	1	6526

- **Determine Free Pages below/ above High Water Mark**

```
db2 "select varchar(tbsp_name, 15) as tbsp_name, tbsp_free_pages,
tbsp_total_pages, tbsp_page_top from table
(mon_get_tablespace('EXTENTREMAP',-2)) as t"
```

TBSP_NAME	TBSP_FREE_PAGES	TBSP_TOTAL_PAGES	TBSP_PAGE_TOP
EXTENTREMAP	6526	3538280	3536394

Free pages above HWM = TBSP_TOTAL_PAGES – TBSP_PAGE_TOP

Free pages below HWM = TBSP_FREE_PAGES – (free pages above HWM)

Autonomic Computing in DB2

- By sensing and responding to changes in the environment, DB2's autonomic computing features **automatically adjust the system to optimize its operation**
- Included in ALL DB2 editions

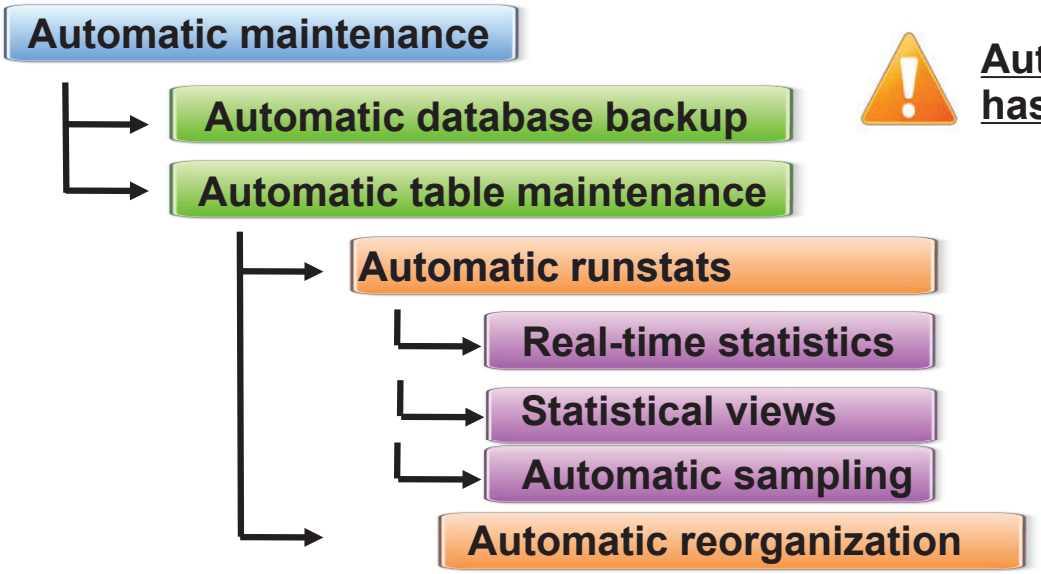
Self-Healing	Self-Configuring	Self-Optimizing	Self-Managing
• Data Studio web console	• Configuration Advisor • Design Advisor	• Utility Throttling • Self-tuning Memory Manager	• Automatic Storage • Automatic Maintenance • Automatic Dictionary Creation

- **Time and cost savings**
  - It shifts the burden of managing a computing environment from DBAs to technology
  - Over 70% of IT budgets are consumed by labor costs

Automatic Object Maintenance

- Automatic Maintenance for statistics, reorganizations and other tasks
  - **AUTO_MAINT** parameter is the **master on/off switch**
    - Individual Child parameters can be set to ON/OFF and the settings are persisted in the database configuration file
- Set maintenance policy via
  - Database configuration
  - Data Studio
  - Stored Procedures
    - SYSPROC.AUTOMAINT_SET_POLICY
    - SYSPROC.AUTOMAINT_SET_POLICYFILE

} Sample XML configuration samples located in: *$YOURINSTANCEHOME/sqllib/samples/automaintcfg*



Automatic statistics profiling (auto_stats_prof) has been deprecated

Automatic Object Maintenance Parameters

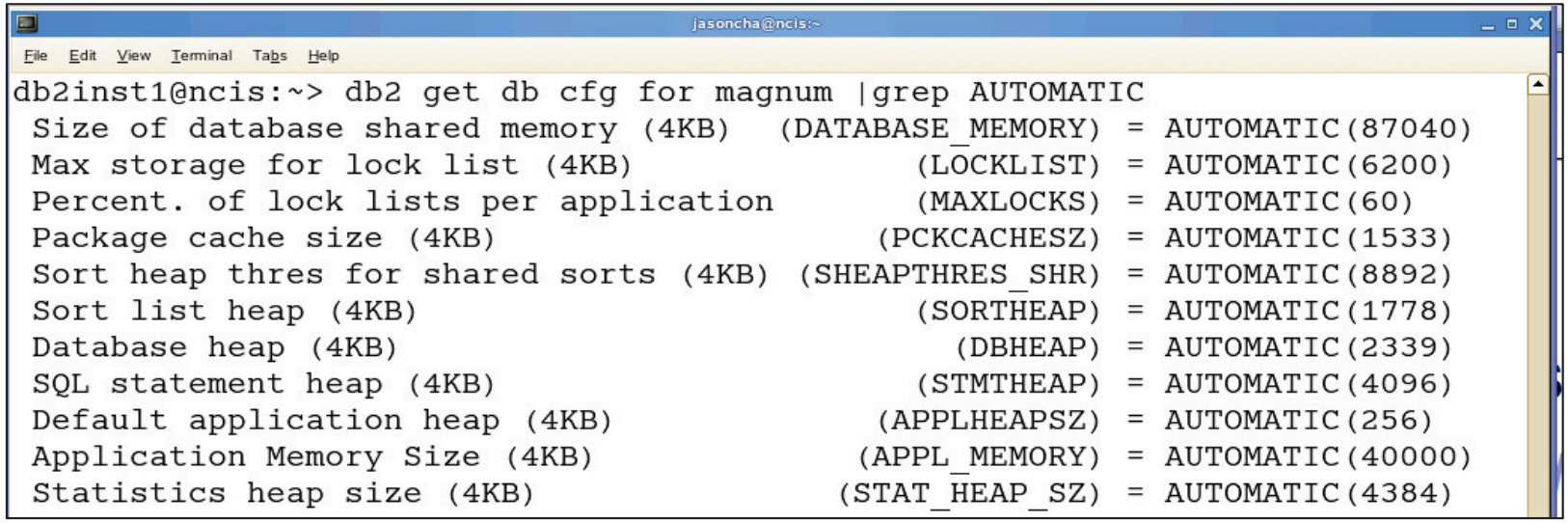
- **Automatic Statistics**
  - **Real-time statistics**
    - Enables/disables collection of real-time stats
  - **Statistical views**
    - Automatically maintain statistical views
  - **Automatic sampling**
    - Control the use of sampling when collecting stats on a large table, sampling rate is automatically determined
- **Automatic Reorg**
  - Automate the reorganization of tables and indexes
  - A reorganization policy may be used to specify the behavior
- **Automatic Backup**
  - A backup policy may be used to specify the automated behavior

Automatic Statistics Collection

- **Enabled by default at database creation**
  - Automatic background statistics collection **auto_runstats**
  - Automatic real-time statistics collection **auto_stmt_stats**
- The **query optimizer** determines how statistics should be collected
  - Based on the query and amount of table update activity
- **Asynchronous statistics collection**
  - Collect statistics that are available to run in the background using runstats utility
- **Real-time statistics collection (synchronous statistics collection)**
  - Provide timely and more accurate statistics at statement compilation
  - Statistics can be *fabricated* using certain meta-data.
  - Statistics can be maintained by the index and data manager and stored directly in the catalog.

Self-Tuning Memory Manager (STMM)

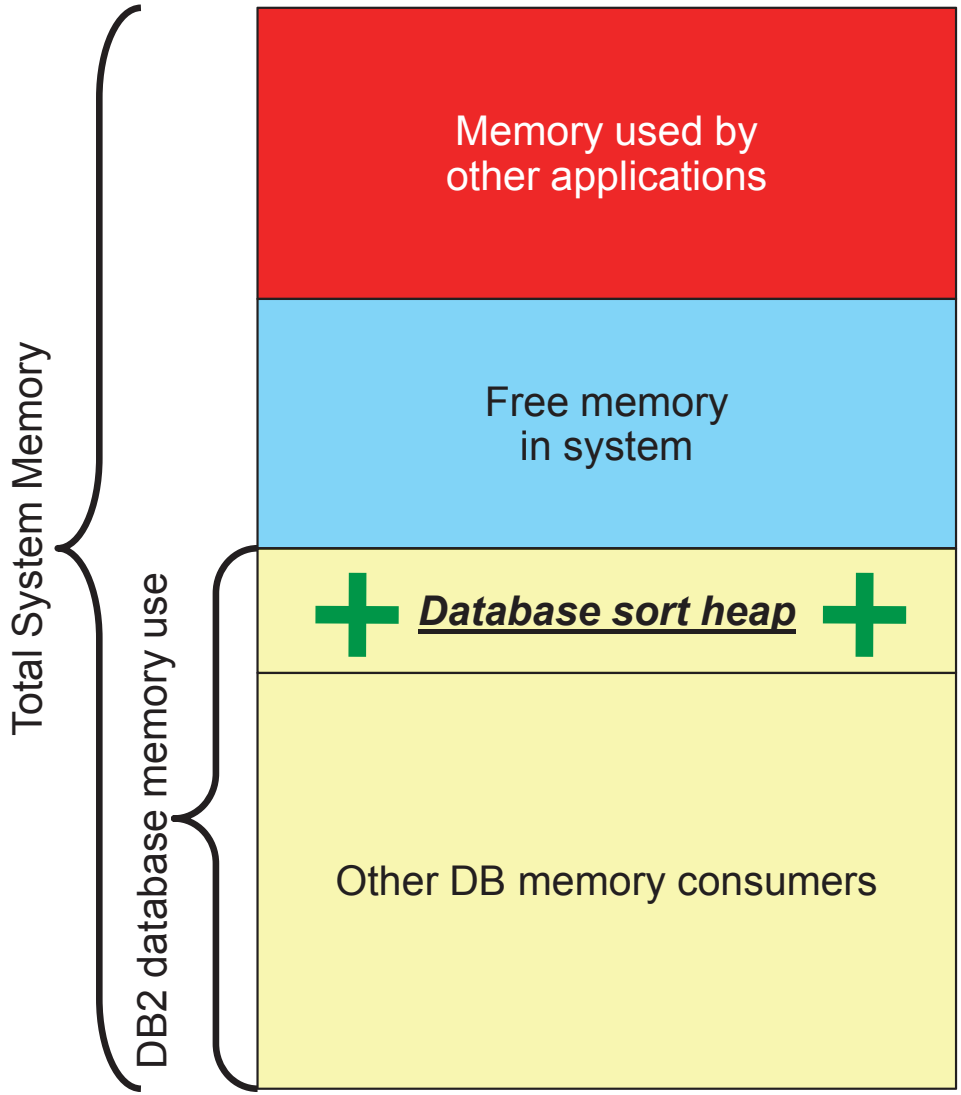
- Optimizes the performance of your database by automatically adjusting the values of:
  - **Total instance memory**
  - **Sort heap, lock list, package cache, SQL statement heap, application heap, application memory size, and total DB memory**
  - **Size of buffer pools**
- STMM is ON by default for all new databases since DB2 9
  - Works with the Database Partitioning Feature (DPF)
  - All affected memory parameters are fully dynamic (does not require instance restart)
- The database configuration parameter **SELF_TUNING_MEM** is the master switch for STMM
  - Configuration parameters and buffer pool sizes should be set to **AUTOMATIC** to enable



STMM Operating Modes

- DATABASE_MEMORY = AUTOMATIC

1)A change in workload occurs that requires more memory for sorts



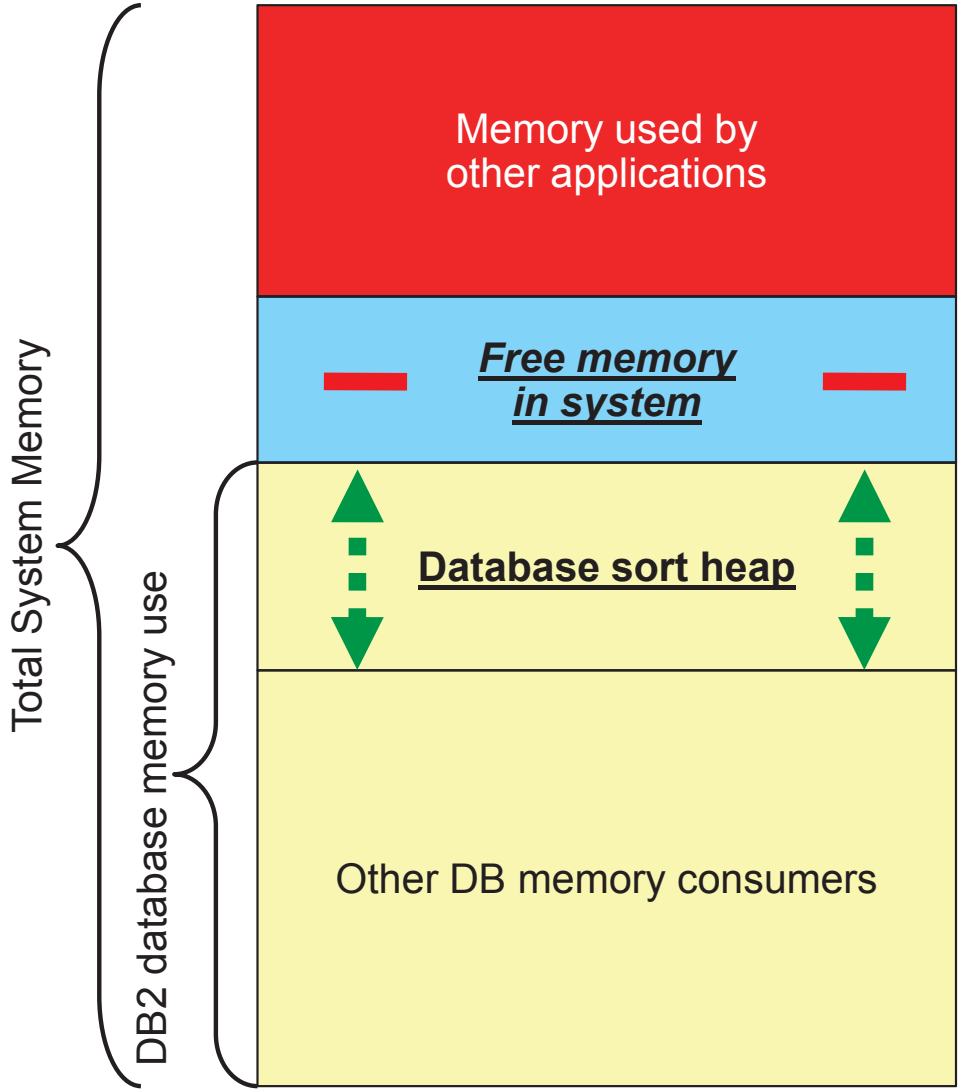
STMM Operating Modes

- DATABASE_MEMORY = AUTOMATIC

1)A change in workload that requires more memory for sorts

2)DB2 requests and gets more memory from the OS

- **Free memory in the system shrinks**
- **DB2 uses newly acquired memory in sort heap**

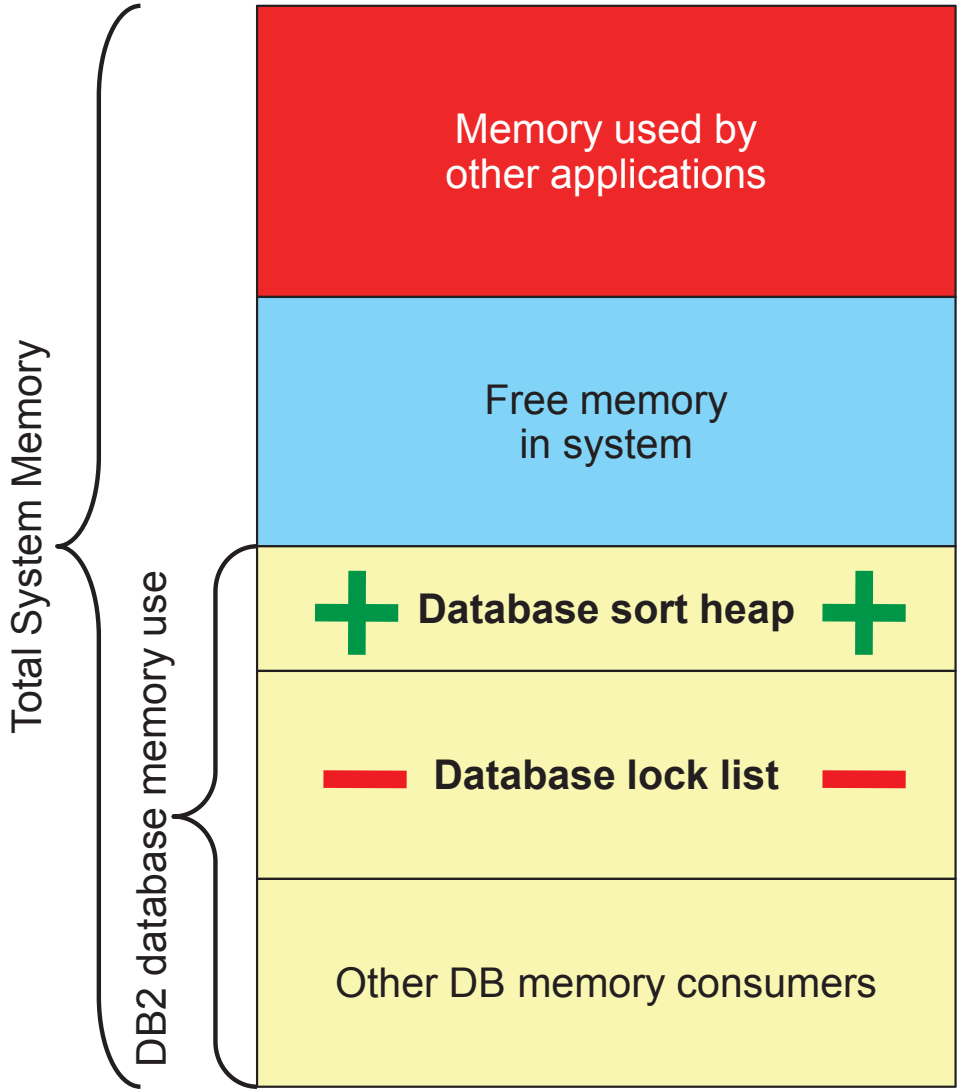


STMM Operating Modes

- DATABASE_MEMORY = COMPUTED or <number>

1) Scenario involves a change in workload that now requires more memory for sorts

2) DB2 database is set at fixed memory usage, and thus cannot take memory from OS. Therefore identifies another memory consumer in the database that does not need its memory anymore (e.g. lock list)



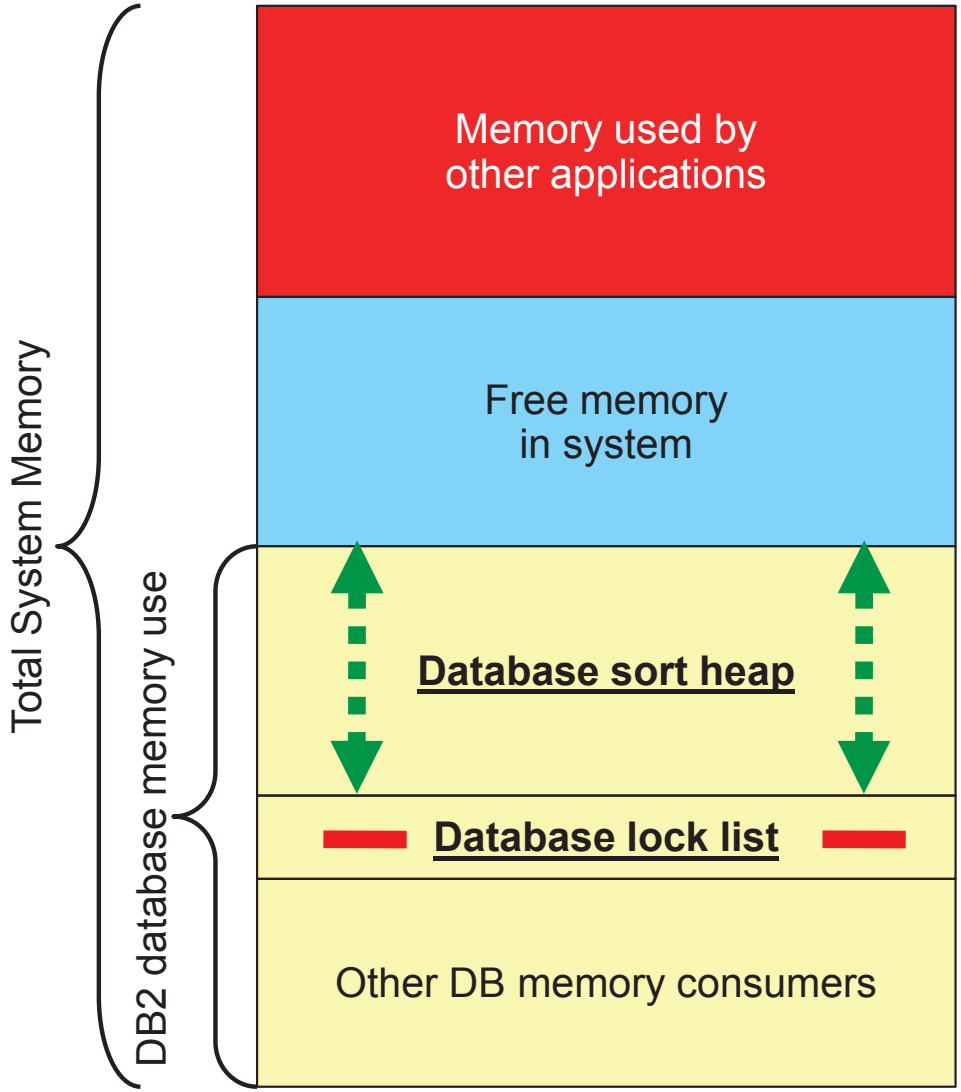
STMM Operating Modes

- DATABASE_MEMORY = COMPUTED or <number>

1) Scenario involves a change in workload that now requires more memory for sorts

2) DB2 database is set at fixed memory usage, and thus cannot take memory from OS. Therefore identifies another memory consumer in the database that does not need its memory anymore (e.g. lock list)

3) The memory is transferred between the memory consumers. The overall memory usage for this DB2 database stays the same.



Automatic Storage, Automatic Dictionary Creation and Utility Throttling

- **Automatic storage**
  - Enabled by default for new databases
    - AUTOMATIC STORAGE clause is deprecated

Automatic table space management	Automatic container management	Automatic resize of DMS table spaces

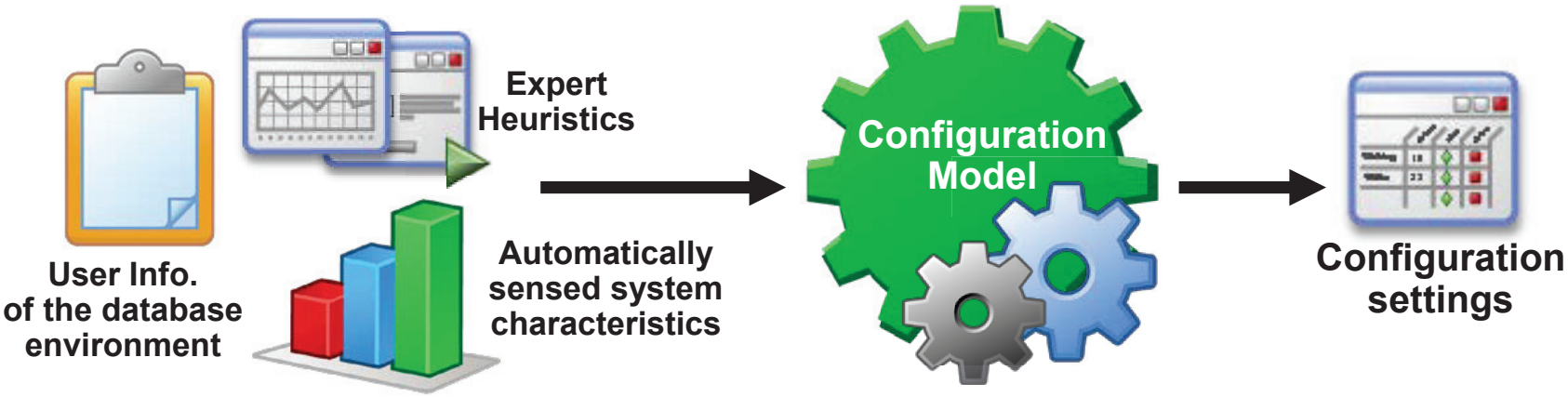
```
CREATE DATABASE <dbname> ON /data/storagePath1, /data/storagePath2

CREATE TABLESPACE TS2 INITIALSIZE 500 K  INCREASESIZE 100 K   MAXSIZE 100 M
```

- **Automatic Dictionary Creation (ADC) for Data Compression**
  - Dictionary is automatically created once amount of data reaches pre-defined threshold
  - Works when using the LOAD utility as well
- **Utility Throttling**
  - Regulates the performance impact of maintenance utilities
  - Ensures that the throttled utilities are run as frequently as possible
    - Statistics collection, backup operations, rebalancing operations, and asynchronous index cleanups
  - Use the db2utilitycontrol API or set UTIL_IMPACT_PRIORITY

Configuration and Design Advisor

Configuration Advisor

- Uses expert heuristics to tune performance and to balance memory requirements

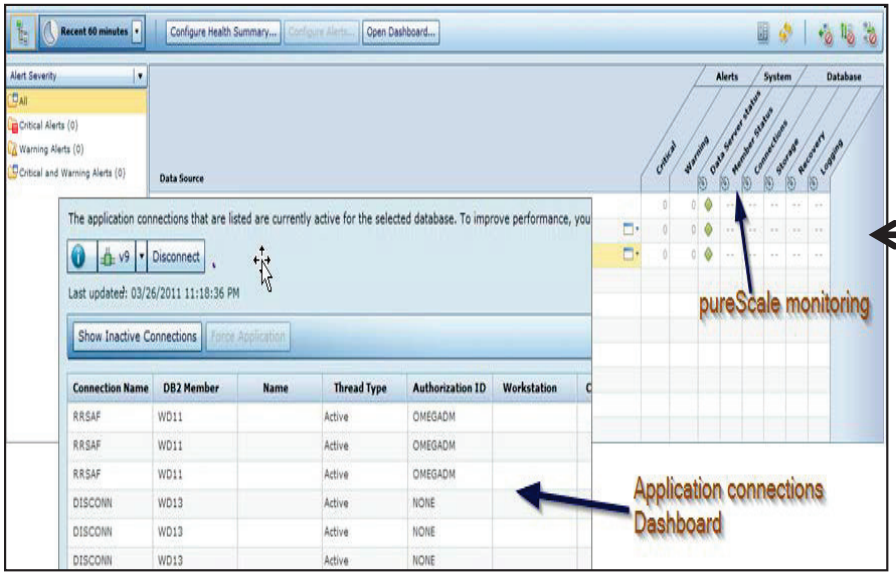


- To use, specify the AUTOCONFIGURE command for an existing database
  - Run by default when you issue the CREATE DATABASE command.

Design Advisor

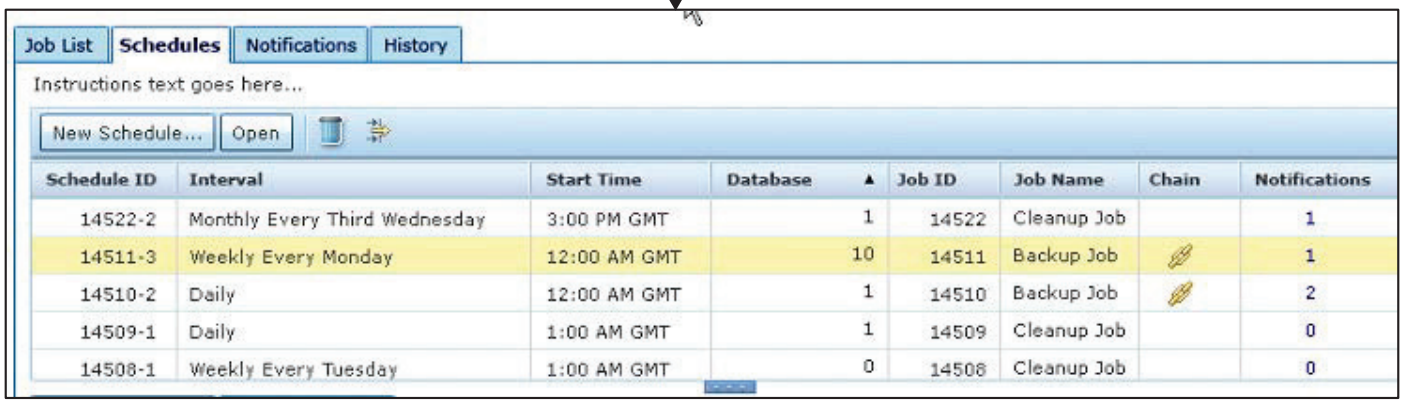
- Tool that suggests modifications to the database's physical design to improve performance
- Allows to display, edit, or save the new database object creation recommendation set
  - Suggestions such as creation of indexes, MQTs, MDCs, or redistribution of tables
  - Invoked with the **db2advis** command or within SQL using the **DESIGN_ADVISOR** procedure

Data Studio Web Console



Data Studio Web Console

- Provides **health and availability monitoring** features and **job creation and management** functions
- Monitoring
  - Health summary, Alert list, Current application connections, Data Sharing Members, Current utilities, System log, Current Table spaces
- Job Manager
  - View, add, edit and delete schedules



Value of BLU Acceleration

Value of DB2 with BLU Acceleration

BLU Acceleration

Next Generation Database for Analytics

- **Extreme performance out-of-the-box**
- **Massive storage savings**
  - No indexes required
- **Lower cost of operational analytics**

Seamlessly Integrated

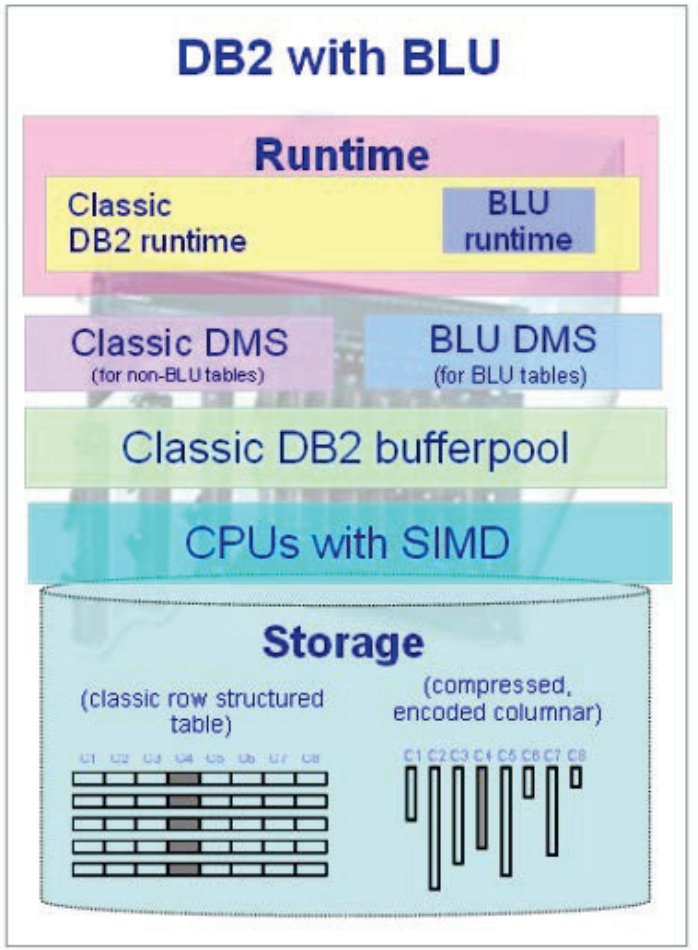
- **Built seamlessly into DB2**
- **Consistent SQL, interfaces, administration**
- **Dramatic simplification**
  - Less to design
  - Less to tune
  - **Just Load and Go**

Hardware Optimized

- **In memory optimized**
  - Compressed in memory
- **Modern CPU Exploitation**
- **I/O optimized**
  - Only read columns of interest

What is DB2 with BLU Acceleration?

- **New innovative technology for analytic queries**
  - Columnar storage
  - New run-time engine with vector (aka SIMD) processing, deep multi-core optimizations and cache-aware memory management
  - "Active compression" - unique encoding for further storage reduction beyond DB2 10 levels, and run-time processing without decompression
- **Value : Order-of-magnitude benefits in …**
  - Performance
  - Storage savings
  - Simplicity !
- **"Revolution by Evolution"**
  - Built directly into the DB2 kernel
  - BLU tables can coexists with traditional row tables, in same schema, tablespaces, buffer pools
  - Query any combination of BLU or row data
  - Memory-optimized (not "in-memory")



Super fast, Super Easy – Create, Load and Go

DATABASE

MICROSOFT
SYBASE
TERADATA
ORACLE

Database Design and Tuning

- Decide on partition strategies
- Select Compression Strategy
- Create Table
- Load data
- Create Auxiliary Performance Structures
  - Materialized views
  - Create indexes
    - B+ indexes
    - Bitmap indexes

Repeat

- Tune memory
- Tune I/O
- Add Optimizer hints
- Statistics collection

Seamless Integration into DB2

- **Built seamlessly into DB2 – integration and coexistence**
  - Column-organized tables can coexist with existing, traditional, tables
    - Same schema, same storage, same memory
  - Integrated tooling support
    - Optim Query Workload Tuner recommends BLU Acceleration deployments

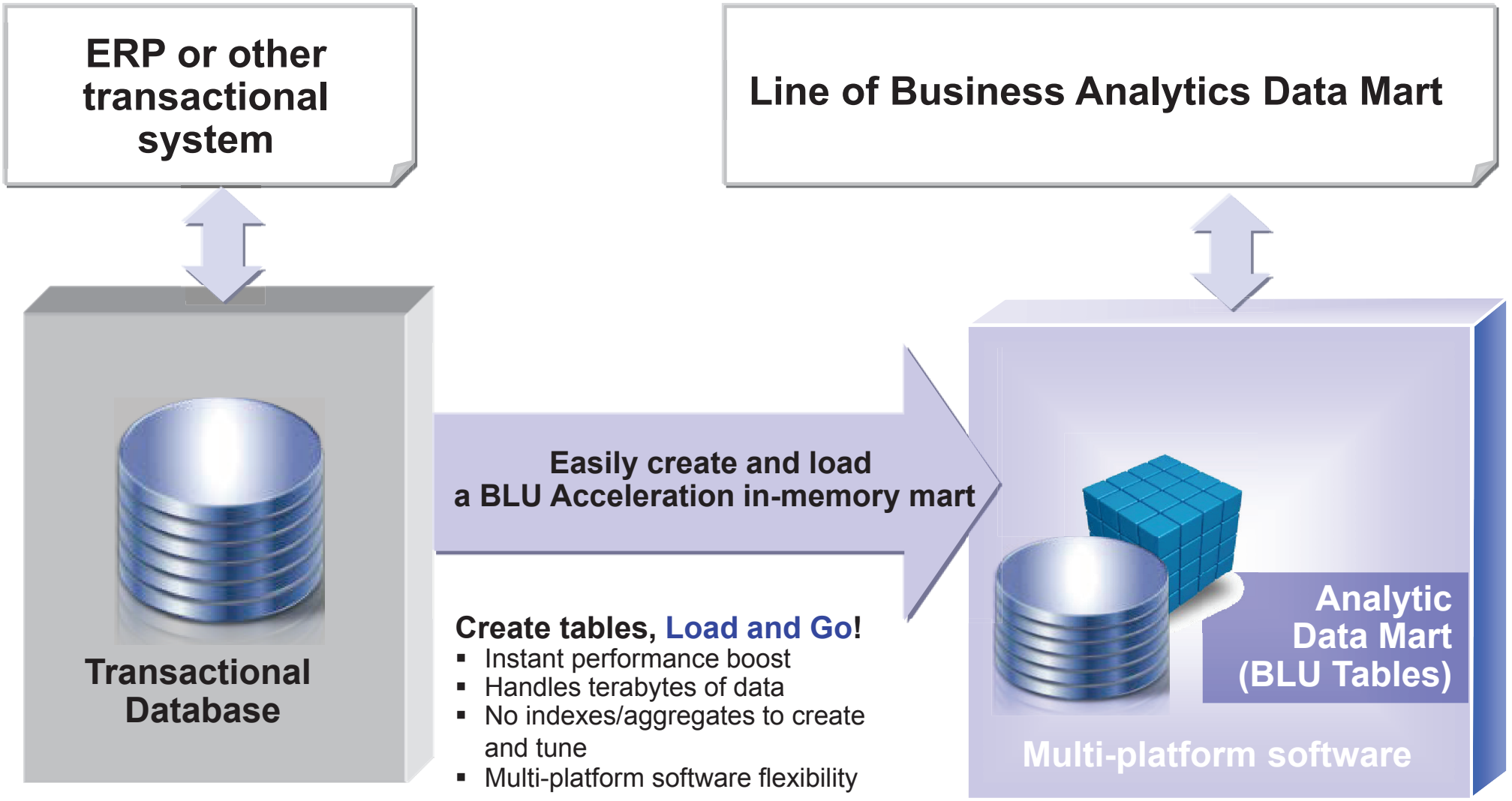
- **Same SQL, language interfaces, administration**
  - Column-organized tables or combinations of column-organized and row-organized tables can be accessed within the same SQL statement

- **Dramatic simplification – Just "Load and Go"**
  - Faster deployment
    - Fewer database objects required to achieve same outcome
  - Requires less ongoing management due to it's optimized query processing and fewer database objects required
  - Simple migration
    - Conversion from traditional row table to BLU Acceleration is easy
    - DB2 Workload Manager identifies workloads to tune
    - Optim Query Workload Tuner recommends BLU Acceleration table transformations
    - Users only notice speed up; DBA's only notice less work!
  - Management of single server solutions less expensive than clustered solutions

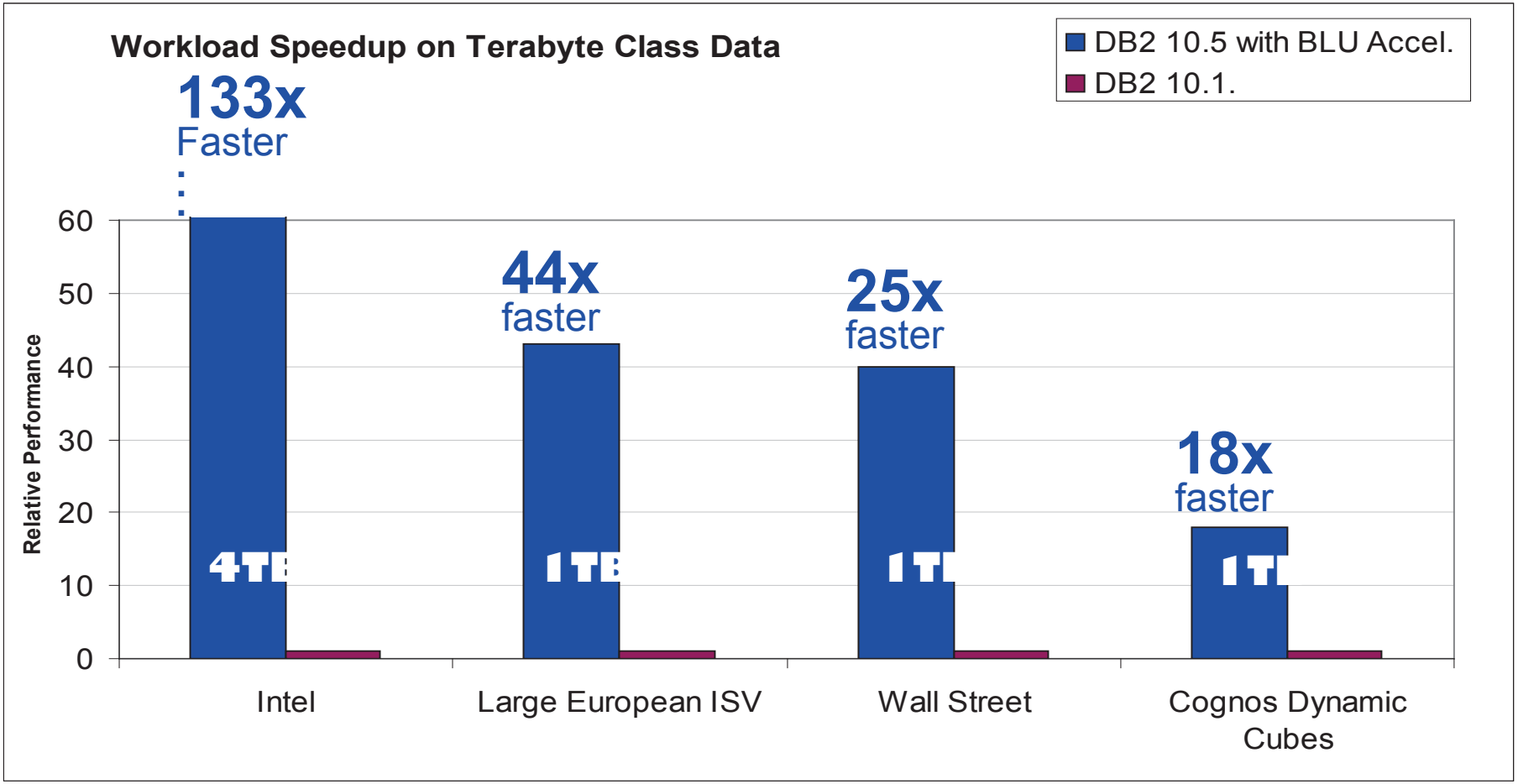
Simple to Deploy and Operate

- **Operations**
  - Simply **Load and Go**
  - Installation to business value in ~2 days
  - Ease of evaluation and performs as advertised
- **BI developers and DBAs – faster delivery**
  - No configuration or physical modeling
  - No indexes or tuning – out of the box performance
  - Data Architects/DBA focus on business value, not physical design
- **ETL developers**
  - No aggregate tables needed – simpler ETL logic
  - Faster load and transformation times
- **Business analysts**
  - Train of thought analysis – 5x to 100x faster
  - True ad-hoc queries – no tuning, no indexes
  - Ask complex queries against large datasets

Analytics Data Mart
From Transactional Database



Terabyte Class Results, March-April 2013



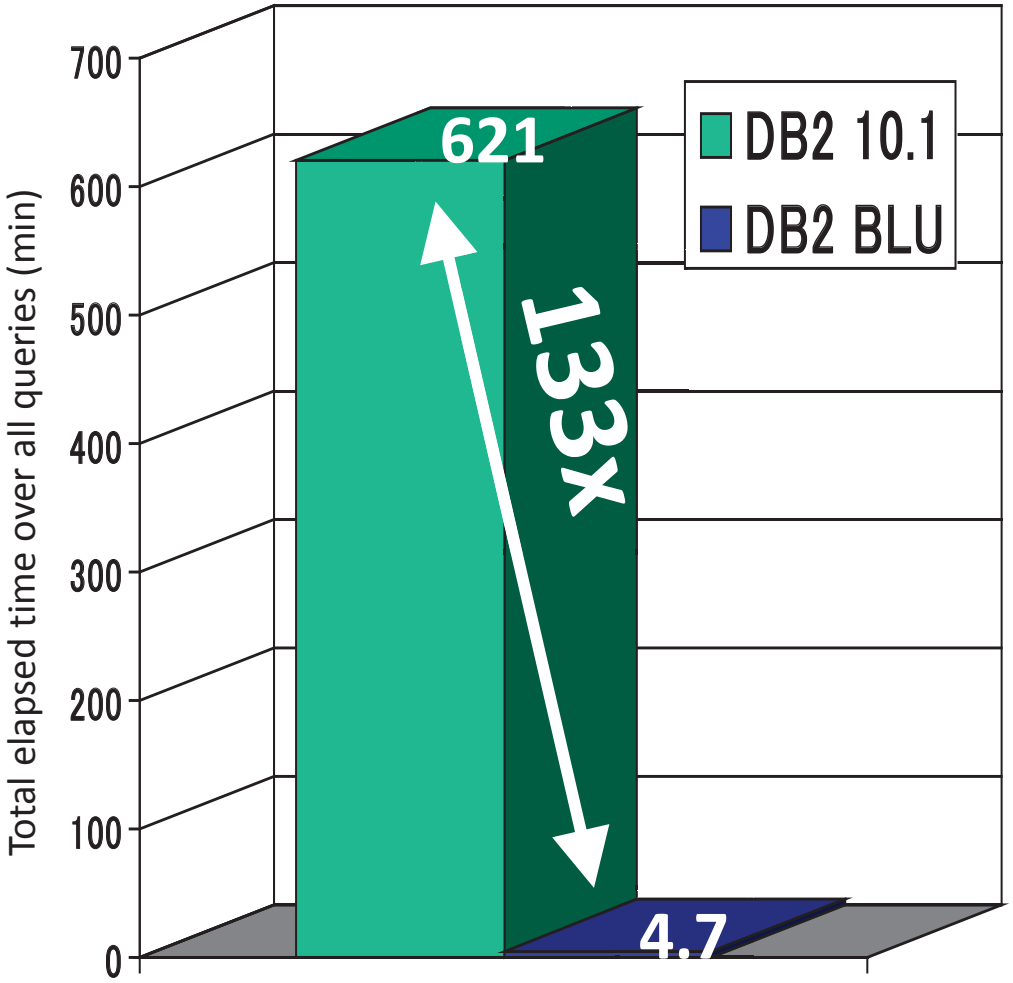
BNSF RAILWAY

"It was amazing to see the faster query times compared to the performance results with our row-organized tables. ***The performance of four of our queries improved by over 100-fold! The best outcome was a query that finished 137x faster by using BLU Acceleration."***

- Kent Collins, Database Solutions Architect, BNSF Railway

Recent Internal Test

- **POPS (Proof of Performance and Scalability)**
  - Derived from Redbrick performance test
  - Classic sales analytics
  - 5.5years of data (2000 days) for 63 stores
    - ~4TB of raw data
    - 2 fact tables
    - 5 dimension tables
  - Broad range of queries with varying selectivity / aggregation
- **Substantial Storage Savings with BLU Acceleration**
  - 2.5x less space than DB2 10.1
- **Massive Performance Gains**
  - **133x** speedup over DB2 10.1
  - Maximum query speed up over **900x**

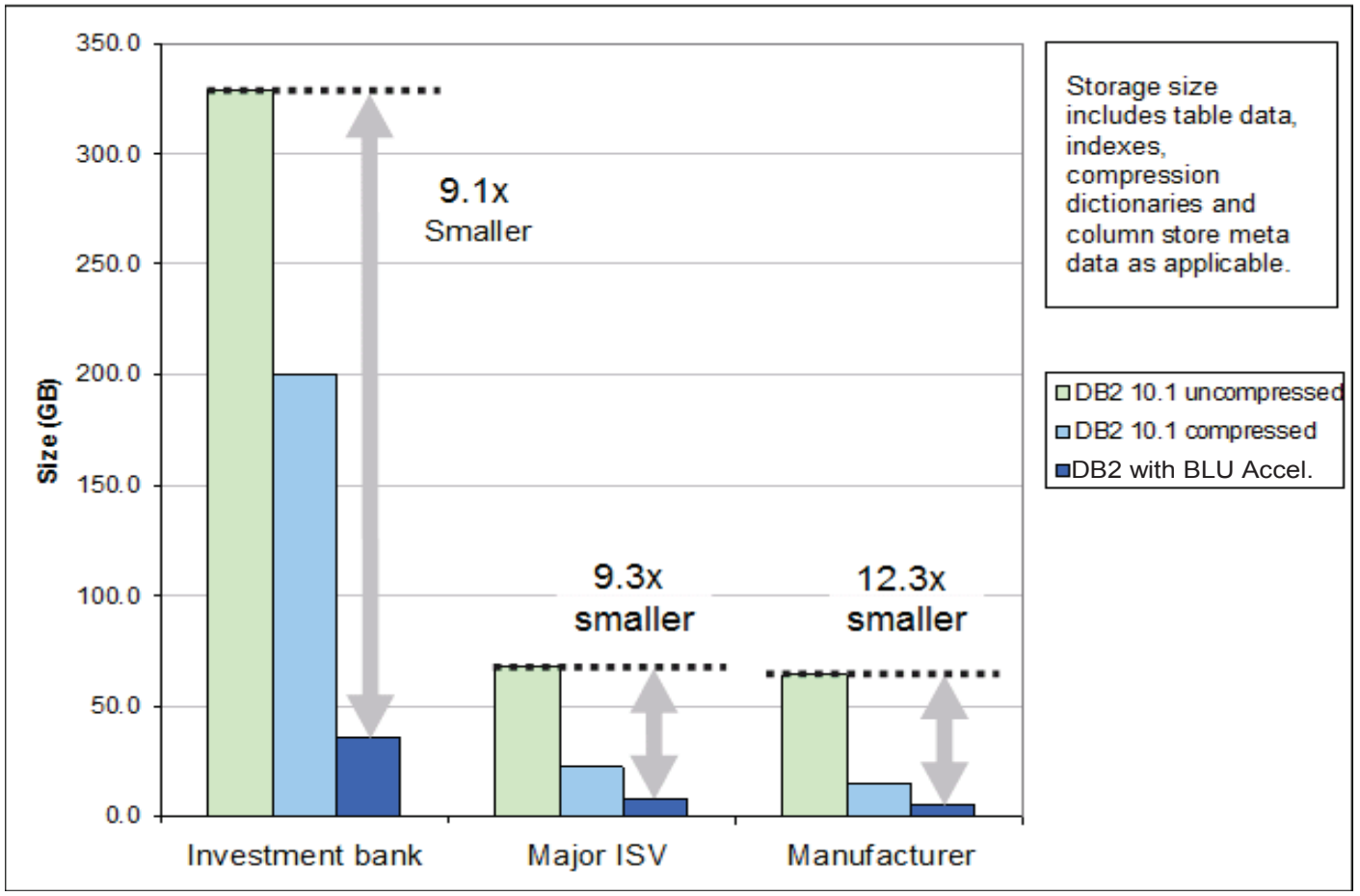


Intel® Xeon® Processor E5-4650
32 cores total (4 CPUs)
384 GB
DS5300 (2x16 disks)

Lab tests - YMMV

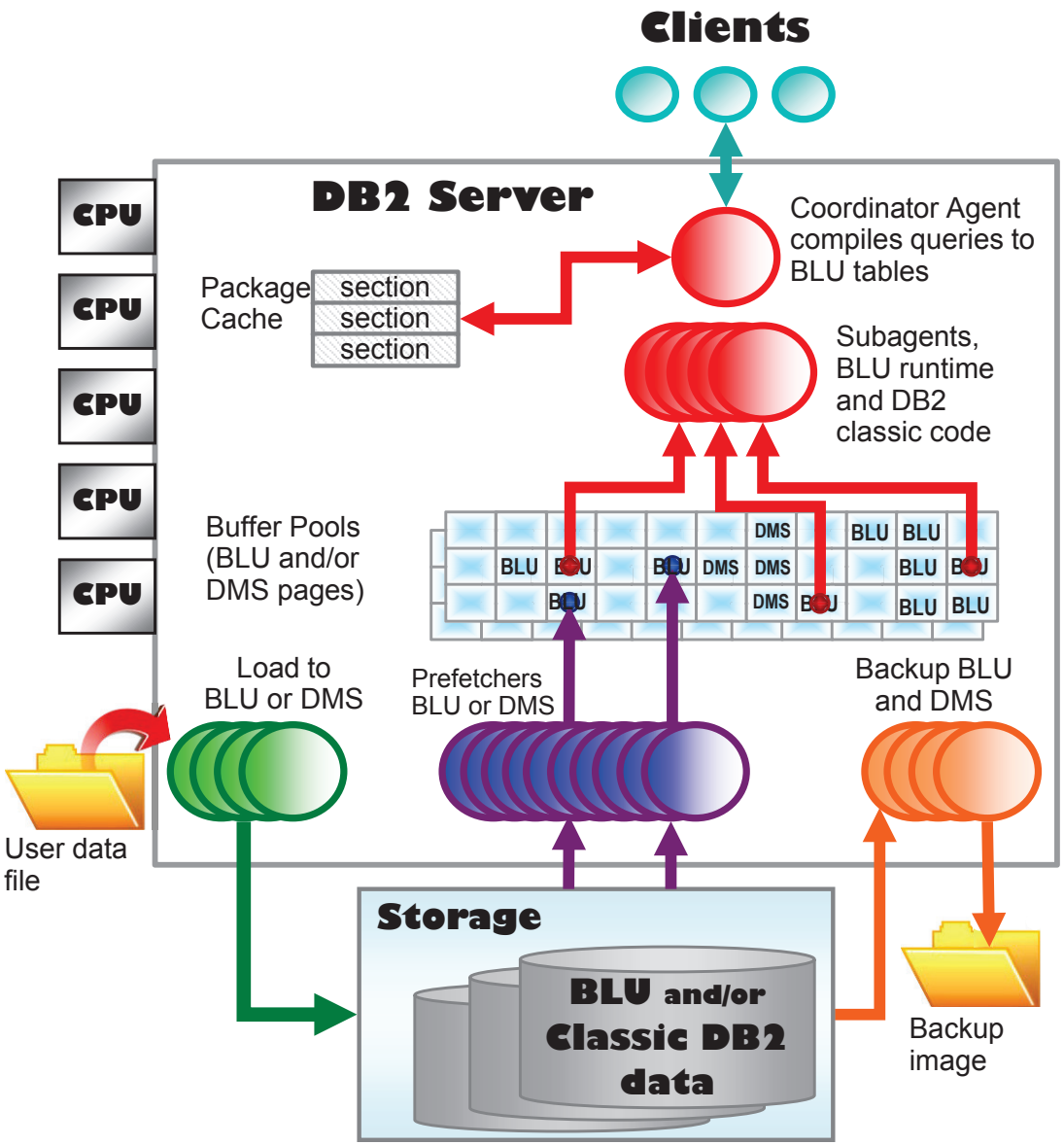
Significant Storage Savings

- **~2x-3x storage reduction vs DB2 10.1 adaptive compression (comparing all objects - tables, indexes, etc)**
  - New advanced compression techniques
  - Fewer storage objects required



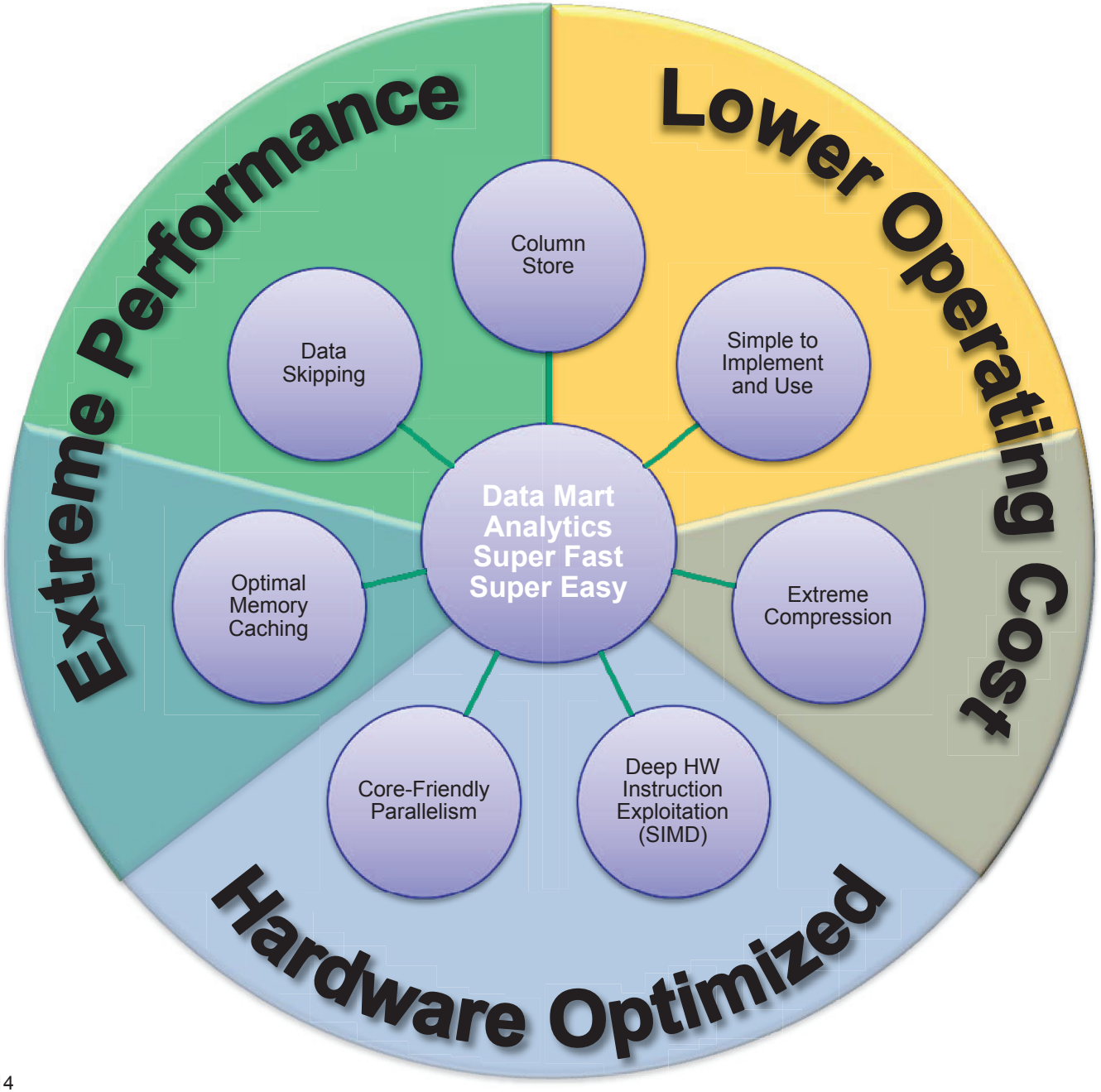
Lab tests - YMMV

BLU Acceleration is Deeply Integrated With the DB2 Kernel



- **Client/Server**
  - BLU Acceleration uses DB2 client server infrastructure. Complete transparency to the application
- **Compiler**
  - BLU Acceleration uses the DB2 compiler to accept SQL, parse, perform semantic checking, and package creation
- **Process model – BLU Acceleration uses**
  - DB2 subagents
  - Prefetchers
  - TCB and Packed Descriptor for metadata
- **Memory**
  - BLU Acceleration uses DB2 bufferpool for storage allocation and caching
  - BLU Acceleration uses DB2 sort heap and package cache
  - OSS memory allocation for private work areas
- **Storage**
  - BLU Acceleration uses normal DB2 table spaces for storage allocations
  - Page sizes: 4K-32K
- **Utilities**
  - `LOAD, BACKUP, RESTORE, EXPORT, SNAPSHOT, db2top, db2pd, etc.`

The Seven Big Ideas of DB2 with BLU Acceleration



7 Big Ideas: 1 Simple to Implement and Use

- **LOAD and then… run queries**
  - No indexes
  - No `REORG` (it's automated)
  - No `RUNSTATS` (it's automated)
  - No MDC or MQTs or Materialized Views
  - No partitioning
  - No statistical views
  - No optimizer hints

- **It is just DB2!**
  - Same SQL, language interfaces, administration
  - Reuse DB2 process model, storage, utilities

Coca-Cola Bottling Co. CONSOLIDATED

"The BLU Acceleration technology has some obvious benefits: It makes our analytical **queries run 4-15x faster** and **decreases the size of our tables by a factor of 10x**. But it's when I think about **all the things I don't have to do with BLU**, it made me appreciate the technology even more: **no tuning, no partitioning, no indexes, no aggregates**."
-Andrew Juarez, Lead SAP Basis and DBA

7 Big Ideas: 1 Simple to Implement and Use

- **One setting optimized the system for BLU Acceleration**
  - Set `DB2_WORKLOAD=ANALYTICS`
  - Informs DB2 that the database will be used for analytic workloads

- **Automatically configures DB2 for optimal analytics performance**
  - Makes column-organized tables the default table type
  - Enables automatic workload management
  - Enables automatic space reclaim
  - Sets up default page (32KB) and extent size (4) appropriate for analytics
  - Memory for caching, sorting and hashing, utilities are automatically initialized based on the server size and available RAM

- **Simple Table Creation**
  - If `DB2_WORKLOAD=ANALYTICS`, tables will be created column organized automatically
  - For mixed table types can define tables as `ORGANIZE BY COLUMN` or `ROW`
  - Compression is always on – No options

- **Easily convert tables from row-organized to column-organized**
  - `db2convert` utility

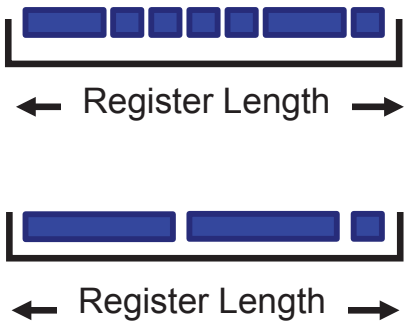
7 Big Ideas: 2 Compute Friendly Encoding and Compression

- **Massive compression with approximate Huffman encoding**
  - More frequent the value, the fewer bits it takes

- **Register-friendly encoding dramatically improves efficiency**
  - Encoded values packed into bits matching the register width of the CPU
  - Fewer I/Os, better memory utilization, fewer CPU cycles to process

LAST_NAME	Encoding
Brown	
Johnson	
Johnson	
Johnson	
Johnson	
Brown	
Johnson	
Gilligan	
Wong	
Johnson	

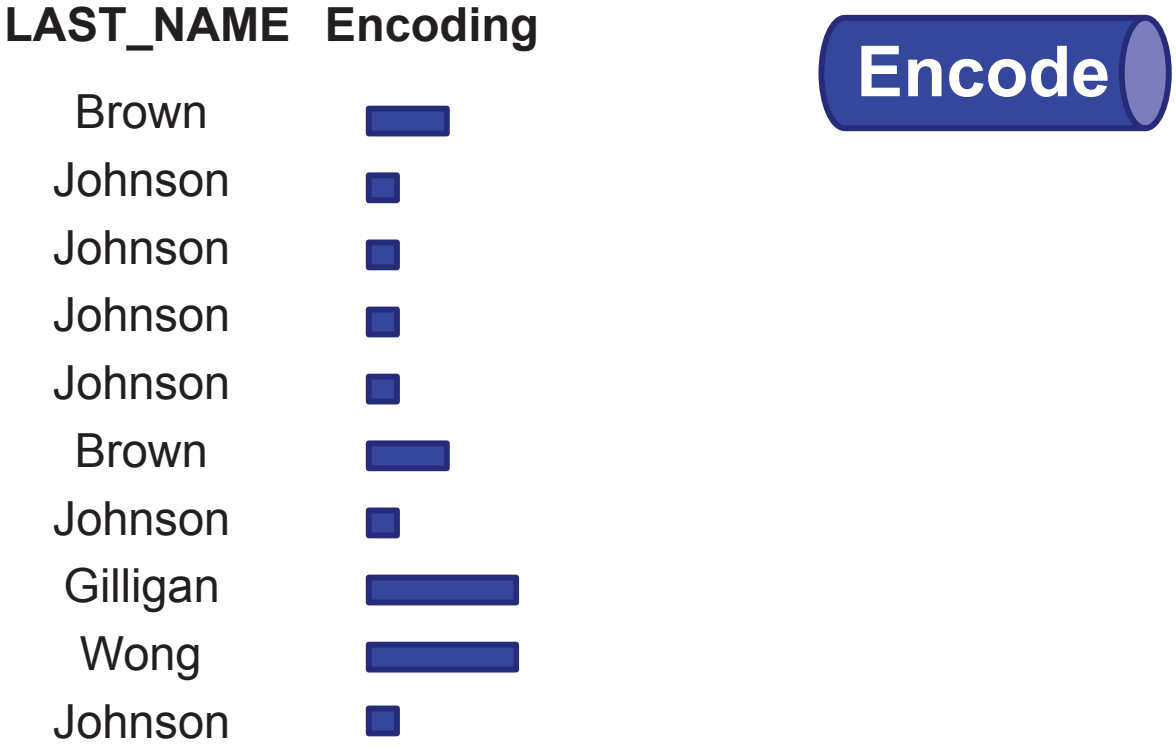
Packed into register length



7 Big Ideas: 2 Data Remains Compressed During Evaluation

- **Encoded values do not need to be decompressed during evaluation**
  - Predicates (=, <, >, >=, <=, Between, etc), joins, aggregations and more work directly on encoded values

```
SELECT COUNT(*) FROM T1 WHERE LAST_NAME = 'Johnson'
```



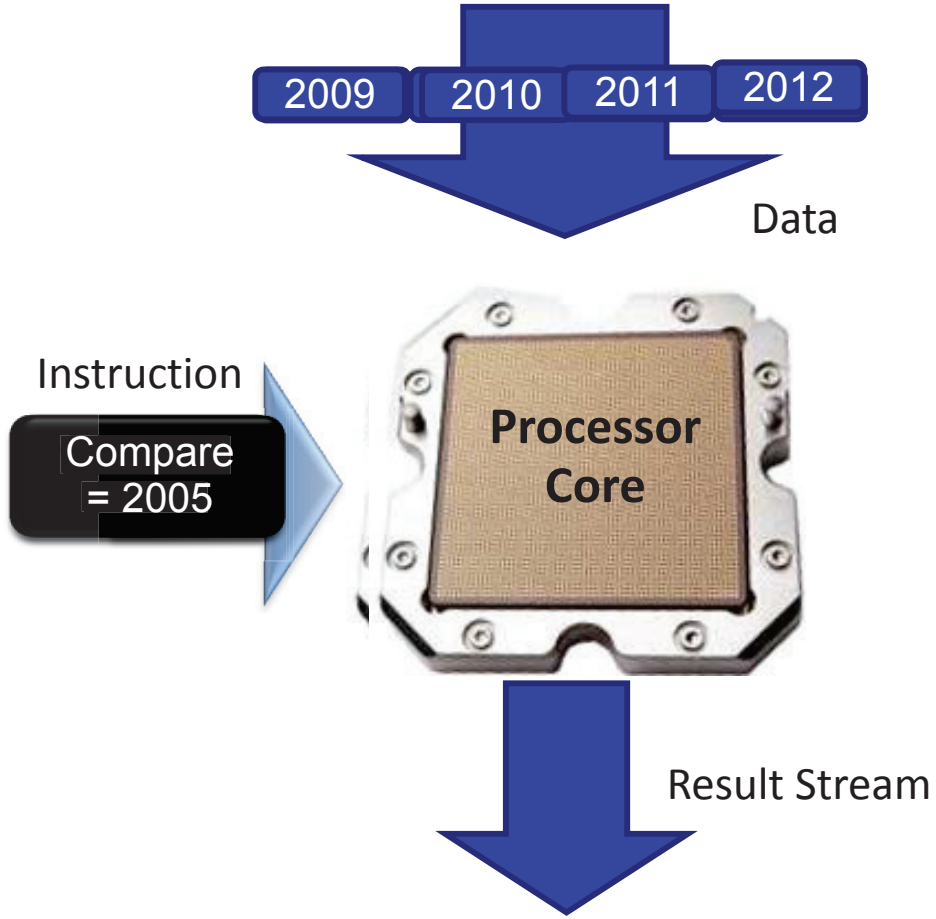
7 Big Ideas: 3 Multiply the Power of the CPU

- **Performance increase with Single Instruction Multiple Data (SIMD)**
- **Using hardware instructions, DB2 with BLU Acceleration can apply a single instruction to many data elements simultaneously**
  - Predicate evaluation, joins, grouping, arithmetic

intel

*"Intel is excited to see a **25x improvement in query processing performance** using DB2 10.5 with BLU acceleration over DB2 10.1. To achieve these amazing gains, IBM has taken advantage of the Advanced Vector Extensions (AVX) instruction set on Intel® Xeon® processor E5-based systems."*

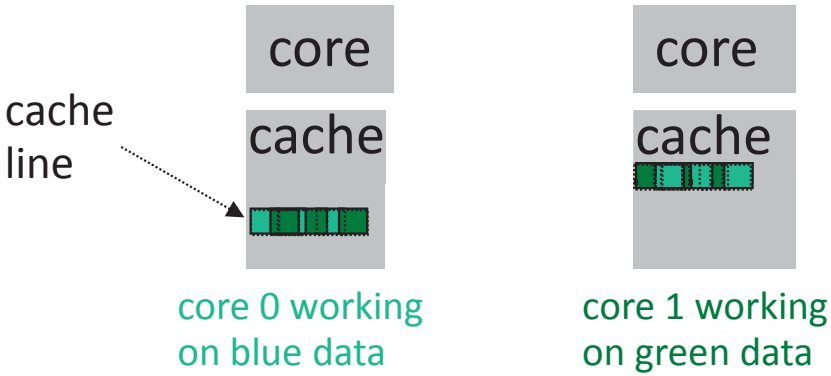
- Pauline Nist, GM, Enterprise Software Alliances, Datacenter & Connected Systems Group

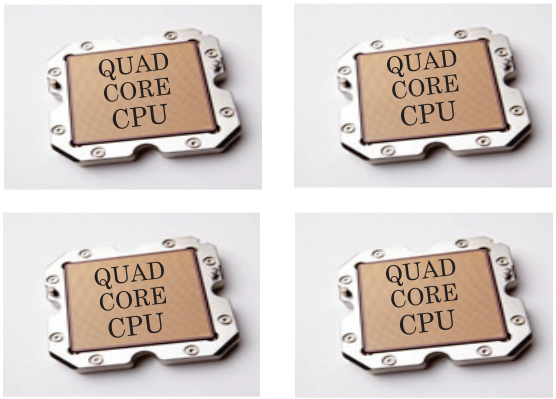


7 Big Ideas: 4 *Core-Friendly Parallelism*

- **BLU queries automatically parallelized across cores, and, achieve excellent multi-core scalability via …**
  - careful data placement and alignment
  - careful attention to physical attributes of the server
  - and other factors, designed to …

… maximize CPU cache hit rate & cacheline efficiency

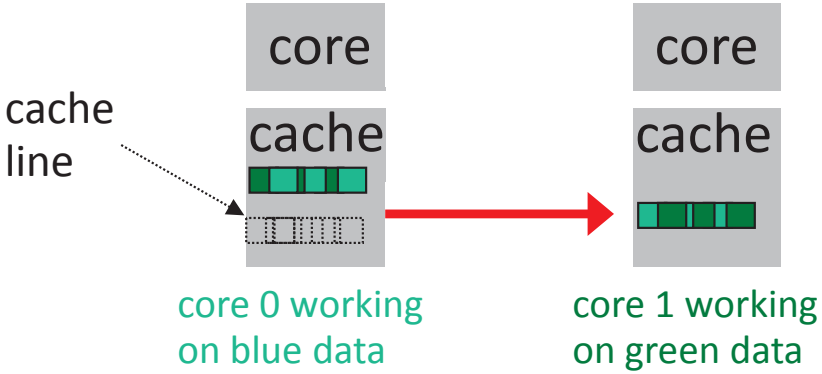


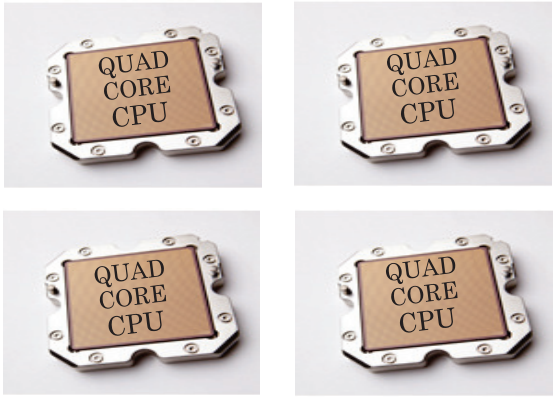


7 Big Ideas: 4 *Core-Friendly Parallelism*

- **BLU queries automatically parallelized across cores, and, achieve excellent multi-core scalability via …**
  - careful data placement and alignment
  - careful attention to physical attributes of the server
  - and other factors, designed to …

… maximize CPU cache hit rate & cacheline efficiency

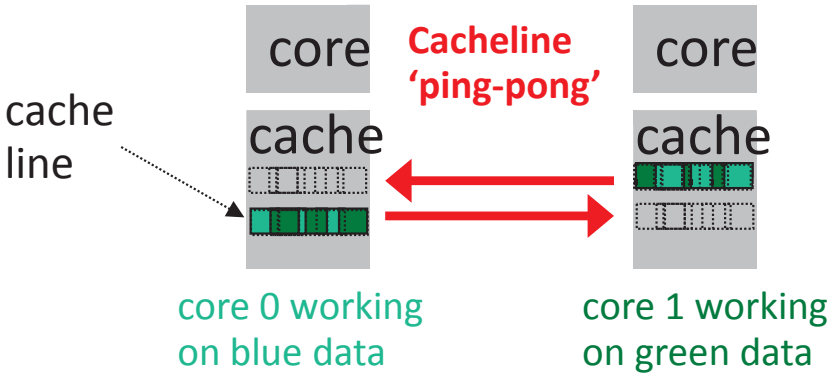


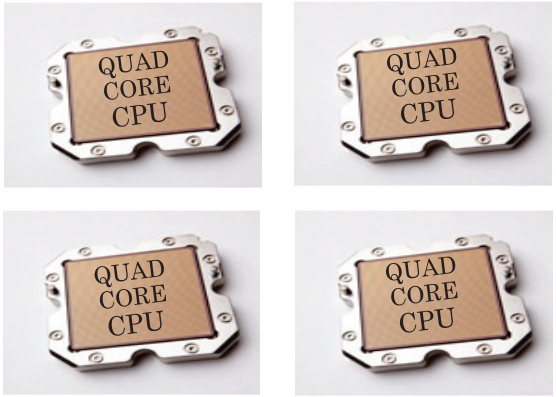


7 Big Ideas: 4 *Core-Friendly Parallelism*

- **BLU queries automatically parallelized across cores, and, achieve excellent multi-core scalability via …**
  - careful data placement and alignment
  - careful attention to physical attributes of the server
  - and other factors, designed to …

… maximize CPU cache hit rate & cacheline efficiency

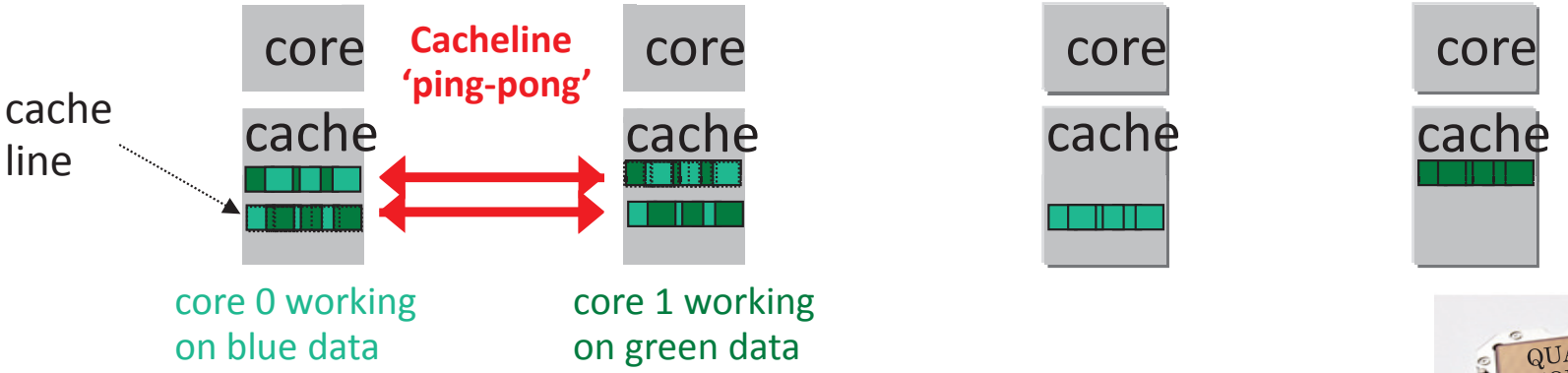


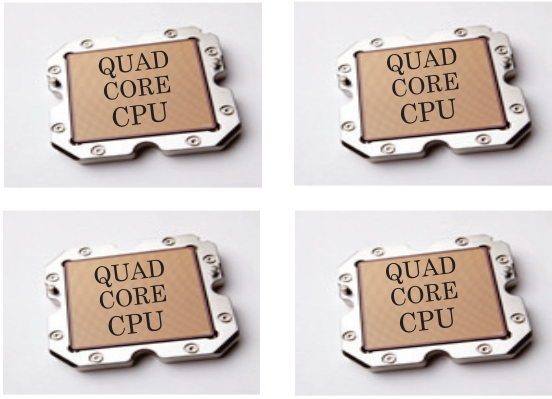


7 Big Ideas: 4 *Core-Friendly Parallelism*

- **BLU queries automatically parallelized across cores, and, achieve excellent multi-core scalability via …**
  - careful data placement and alignment
  - careful attention to physical attributes of the server
  - and other factors, designed to …

… maximize CPU cache hit rate & cacheline efficiency

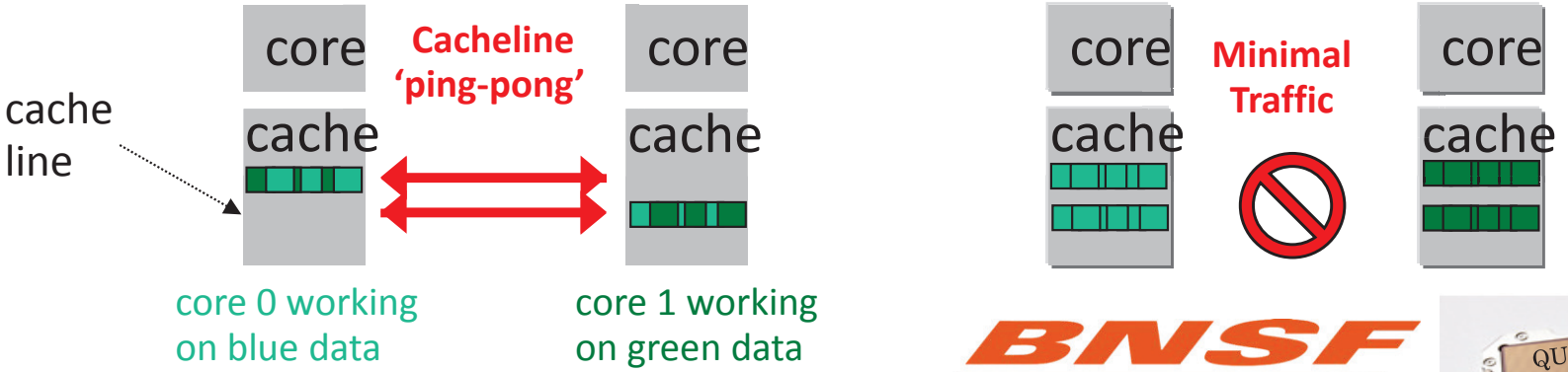




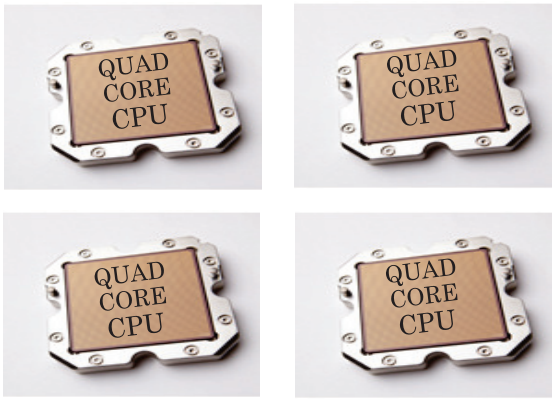
7 Big Ideas: 4 *Core-Friendly Parallelism*

- **BLU queries automatically parallelized across cores, and, achieve excellent multi-core scalability via …**
  - careful data placement and alignment
  - careful attention to physical attributes of the server
  - and other factors, designed to …

… maximize CPU cache hit rate & cacheline efficiency



BNSF RAILWAY

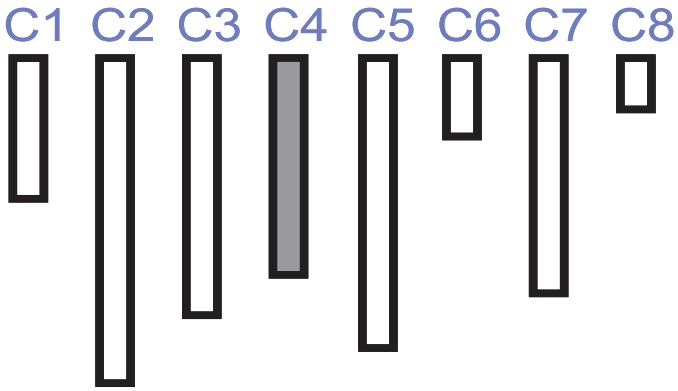


"During our testing, we couldn't help but notice that DB2 10.5 with BLU Acceleration is excellent at utilizing our hardware resources. The core-friendly parallelism that IBM talks about was clearly evident and I didn't even have to partition the data across multiple servers."
- Kent Collins, Database Solutions Architect, BNSF Railway

7 Big Ideas: 5 Column Oriented Storage

- **Minimal I/O**
  - Only perform I/O on the columns and values that match query
  - As queries progresses through a pipeline the working set of pages is reduced
- **Work performed directly on columns**
  - Predicates, joins, scans, etc. all work on individual columns
  - Rows are not materialized until absolutely necessary to build result set
- **Improved memory density**
  - Columnar data kept compressed in memory
- **Extreme compression**
  - Packing more data values into very small amount of memory or disk
- **Cache efficiency**
  - Data packed into cache friendly structures
  - No need to consume memory/cache space & bandwidth for unneeded columns

Columns stored separately and packed in different buffers in memory

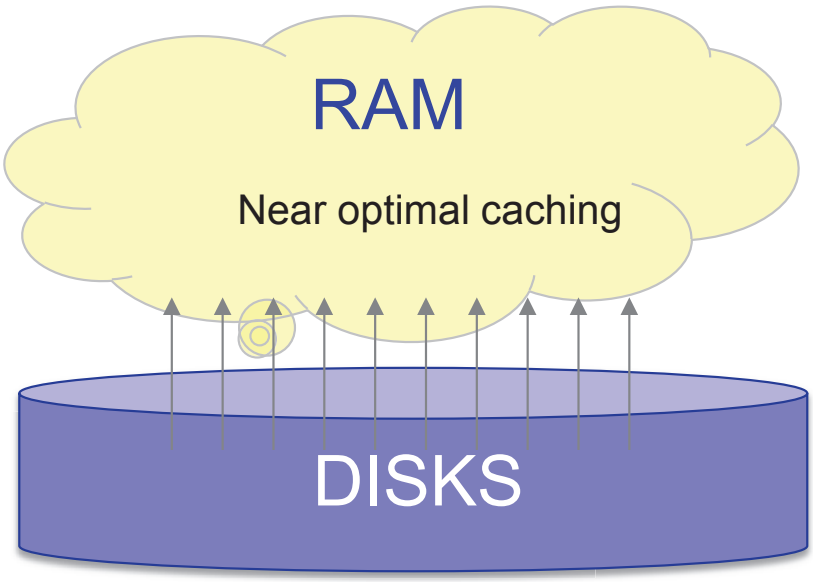


SELECT C4 ... WHERE C4=X

Consumes I/O bandwidth memory buffers and memory bandwidth only for C4

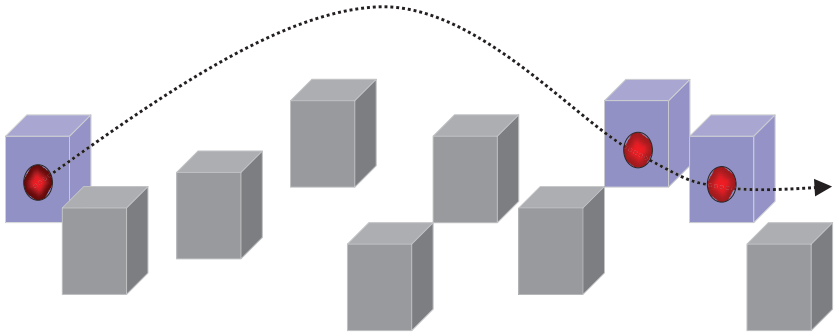
7 Big Ideas: 6 *Scan-Friendly Memory Caching*

- **Memory-optimized (not "In-Memory")**
  - No need to ensure all data fits in memory
- **BLU includes new scan-friendly victim selection to keep a near optimal % of pages buffered in memory**
  - Traditional RDMSes use 'most recently used' victim selection for large scans
    - "There's no hope of caching everything, so just victimize the last page read"
  - A key BLU design point is to run well when all data fits in memory, and when it doesn't !
    - Even with large scans, BLU prefers selected pages in the bufferpool, using an algorithm that adaptively computes a target hit ratio for the current scan, based on the size of the bufferpool, the frequency of pages being re-accessed in the same scan, and other factors
  - Benefit: less I/O !



7 Big Ideas: 7 Data skipping

- Automatic detection of large sections of data that do not qualify for a query and can be ignored
- Order of magnitude savings in all of I/O, RAM, and CPU
- No DBA action to define or use – truly invisible "Synopsis" automatically created and maintained as data is LOADed or INSERTed
  - Persistent storage of min and max values for sections of data values



"One thing evident to me is that there is a lot of technology behind BLU Acceleration. It's ***beyond a simple in-memory column store. It includes leveraging the latest CPU technologies, parallelism techniques, and so much more."***

-Andrew Juarez, Lead SAP Basis and DBA

Optimize the Entire Hardware Stack

In-Memory Optimized

- **Memory latency optimized for**
  - Scans
  - Joins
  - Aggregation
- **More useful data in memory**
  - Data stays compressed
  - Scan friendly caching
- **Less to put in memory**
  - Columnar access
  - Late materialization
  - Data skipping

CPU Optimized

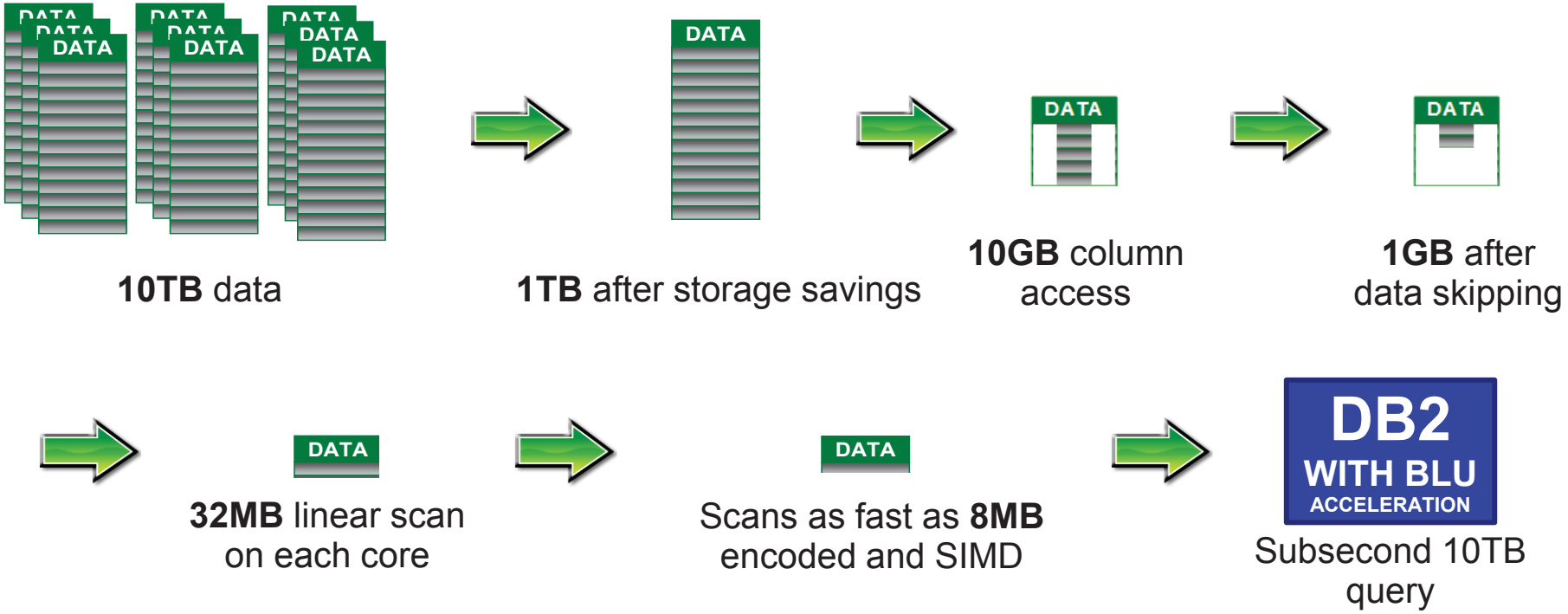
- **CPU acceleration**
  - SIMD processing for
    - Scans
    - Joins
    - Grouping
    - Arithmetic
- **Keeping the CPUs busy**
  - Core friendly parallelism
- **Less CPU processing**
  - Operate on compressed data
  - Late materialization
  - Data skipping

I/O Optimized

- **Less to read**
  - Columnar I/O
  - Data skipping
  - Late materialization
- **Read less often**
  - Scan friendly caching
- **Efficient I/O**
  - Specialized columnar prefetching algorithm

7 Big Ideas: How DB2 with BLU Acceleration Helps
~Sub second 10TB query – An Optimistic Illustration

- **The system – 32 cores, 10TB table with 100 columns, 10 years of data**
- **The query: `SELECT COUNT(*) from MYTABLE where YEAR = '2010'`**
- **The optimistic result: sub second 10TB query! Each CPU core examines the equivalent of just 8MB of data**



Getting Started with BLU Acceleration

Getting Started with BLU: Platforms and Hardware

- Supported platforms
  - Linux 64-bit on Intel/AMD hardware
  - RHEL 6 or higher, SLES 10 SP4, SLES 11 SP2
  - AIX on Power hardware
  - AIX 6.1 TL7 SP6, AIX 7.1 TL1 SP6

- **For best results, use:**
  - Intel Nehalem or better
  - Power 7

Capacity Recommendations

	Small	Medium	Large
raw data (CSV)	~1TB	~5TB	~10TB
Minimum:			
#cores	8	16	32
Memory	64GB	256GB	512GB
High-end perf:			
#cores	16	32	64
Memory	128 – 256GB	384 – 512GB	1 – 2TB

Assumption: all data is active and equally “hot”.

Will your workload benefit from BLU?

Probably:

- Analytical workloads, data marts, etc.
- Grouping, aggregation, range scans, joins
- Queries touch only a subset of the columns in a table
- Star Schema
- SAP Business Warehouse

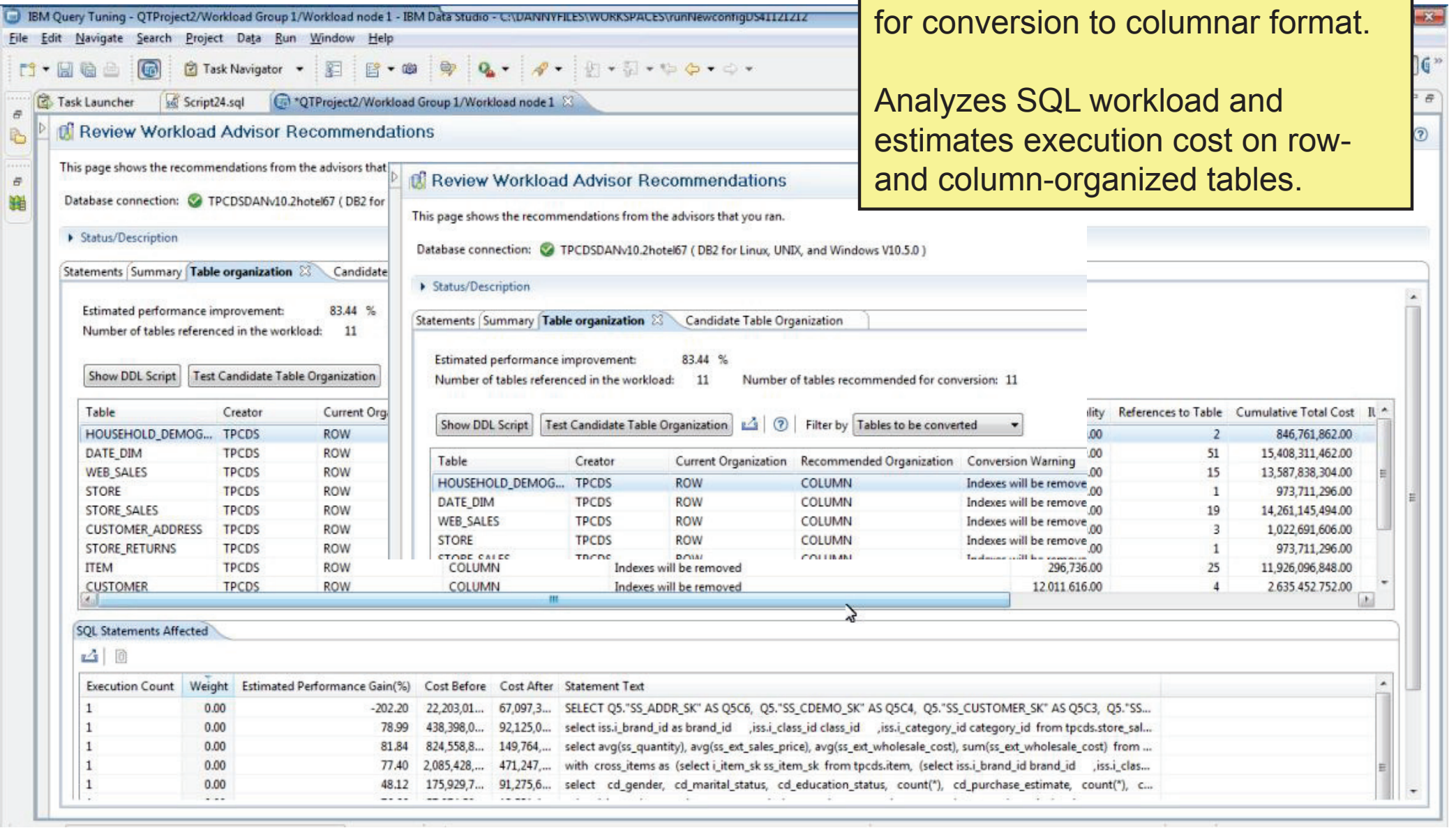
Probably not:

- OLTP
- Point access to 1 or few rows
- Insert, Update, Delete of few rows per transaction
- Queries touch many or all columns in a table
- Use of XML, Temporal, LOBs, etc. which are not supported in BLU yet

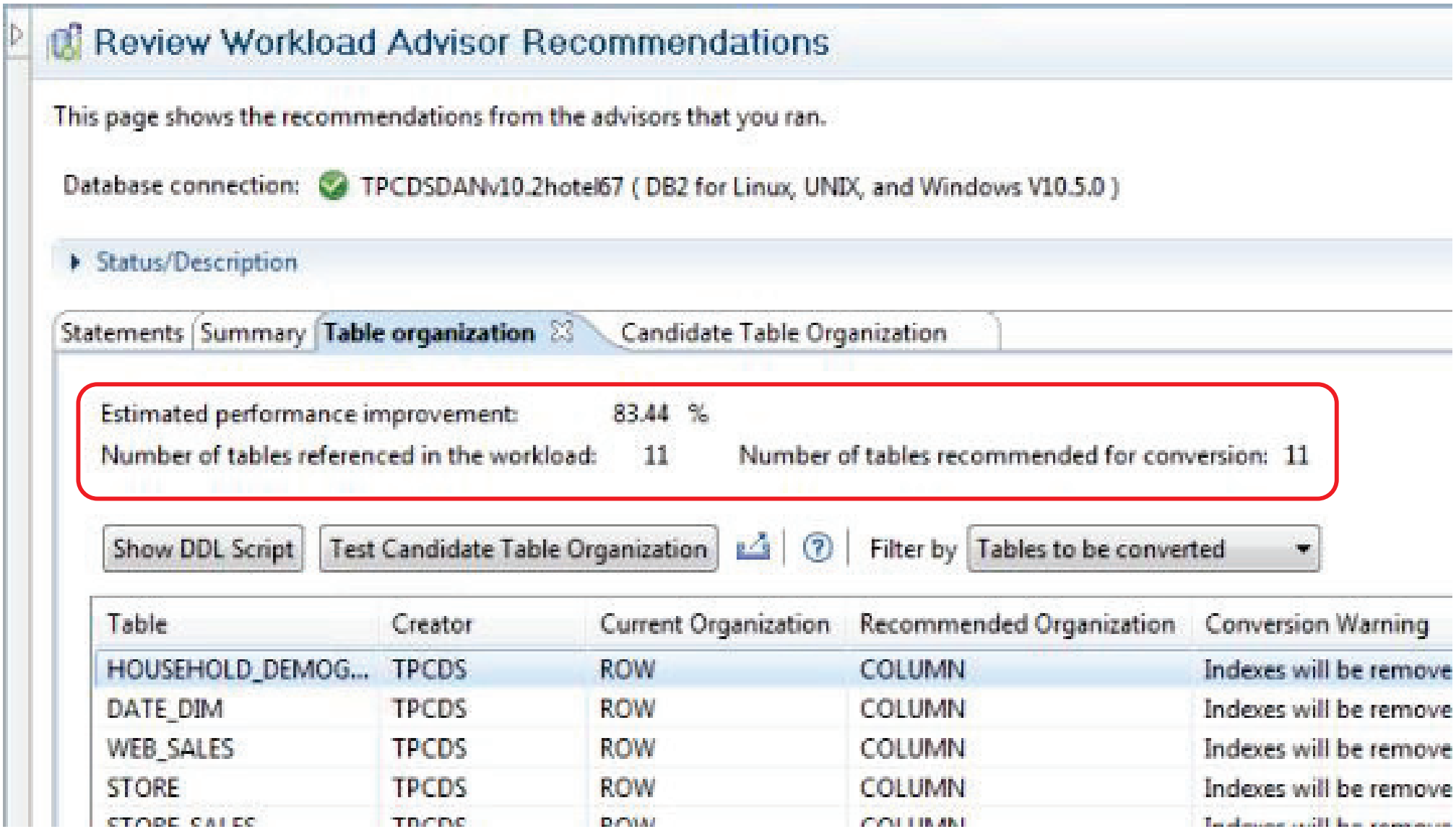
IBM Optim Query Tuner

Advisor identifies candidate tables for conversion to columnar format.

Analyzes SQL workload and estimates execution cost on row- and column-organized tables.



IBM Optim Query Tuner



db2set DB2_WORKLOAD=ANALYTICS

- Set DB2_WORKLOAD=ANALYTICS **before** creating your database
- Don't disable AUTOCONFIGURE

- For an existing database:
  - set DB2_WORKLOAD=ANALYTICS
  - then run AUTOCONFIGURE
    to get some (but not all) of the recommended settings

- Ideally, you won't need to set anything else!
- Verify that sort heap, utility heap, and BPs are large

DB2_WORKLOAD=ANALYTICS – What does it do?

- **dft_table_org** = COLUMN
- **default page size** for a new database is 32KB
- **dft_extent_sz** = 4
- **dft_degree** = ANY
- **Intra query parallelism** is enabled for any workload (including SYSDEFAULTUSERWORKLOAD) that specifies MAXIMUM DEGREE DEFAULT, even if intra_parallel is disabled.
- **catalogcache_sz** - higher value than default
- **sortheap** and **sheapthres_shr** - higher value than default.
- **util_heap_sz** – higher value than default
- **WLM** controls concurrency on SYSDEFAULTMANAGEDSUBCLASS.
- **Automatic table maintenance** and **auto_reorg** = ON, performs space reclamation for column-organized tables by default.

Creating a column-organized table

- Example:

```
CREATE TABLE sales_col (
  c1 INTEGER NOT NULL,
  c2 INTEGER,
   ...
  PRIMARY KEY (c1) ) ORGANIZE BY COLUMN;
```

Columnar tables are always compressed by default.

- If dft_table_org = COLUMN (e.g. DB2_WORKLOAD= ANALYTICS):
  - ORGANIZE BY COLUMN is the default and can be omitted
  - Use ORGANIZE BY ROW to create row-organized tables

- Do not specify compression, MDC, or partitioning for BLU tables.
- Do not create indexes or MQTs.

Informational Uniqueness

- **DB2 10.5 introduces informational uniqueness constraints**
  - Primary keys and unique constraints can be enforced or not enforced
  - Enforced uniqueness remains the default
  - Informational (i.e., `NOT ENFORCED`) constraints do not enforce uniqueness
  - Valuable when data is coming from a trusted source
  - Be sure data is cleansed and truly unique!
- **Benefits**
  - Less storage required! No index is created to enforce the constraint.
  - No runtime overhead to maintain unique indexes during `LOAD, INSERT, UPDATE, or DELETE`
  - The uniqueness definition informs the query compiler of unique data, enabling opportunities for superior query execution plans

```
CREATE TABLE t1 (c1 INTEGER NOT NULL, c2 INTEGER,
                                 PRIMARY KEY (c1) NOT ENFORCED);

ALTER TABLE t1 ADD CONSTRAINT unique1 UNIQUE (c2) NOT ENFORCED;
```


© 2013 IBM Corporation

Storage orientation

- Input data is in row format (CSV files, etc.):

John Piconne	47	18 Main Street	Springfield	MA	01111
Susan Nakagawa	32	455 N. 1st St.	San Jose	CA	95113
Sam Gerstner	55	911 Elm St.	Toledo	OH	43601
Chou Zhang	22	300 Grand Ave	Los Angeles	CA	90047
Mike Hernandez	43	404 Escuela St.	Los Angeles	CA	90033
Pamela Funk	29	166 Elk Road #47	Beaverton	OR	97075
Rick Washington	78	5661 Bloom St.	Raleigh	NC	27605
Ernesto Fry	35	8883 Longhorn Dr.	Tucson	AZ	85701
Whitney Samuels	80	14 California Blvd.	Pasadena	CA	91117
Carol Whitehead	61	1114 Apple Lane	Cupertino	CA	95014

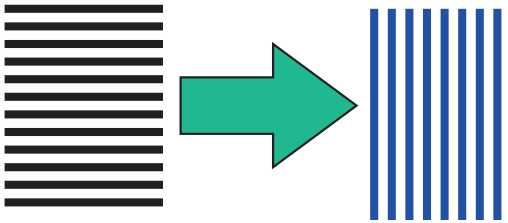
- Gets compressed and converted to columnar format upon LOAD/Insert

Columnar storage in DB2 (conceptual)

- Separate set of extents and pages for each column

- Typically, column-organized tables use less space than row-organized tables
- Column-organized tables with many columns and few rows can be larger than row-organized tables if many mostly empty extents
- TSNs (like logical Row ID) used to stitch column values that belong together in a row during query processing

Converting existing tables: db2convert



- Converts a row-organized table into a column-organized table
- Calls ADMIN_MOVE_TABLE
- Has the same options and restrictions as ADMIN_MOVE_TABLE

```
db2convert
    -d <database-name>   (this is the only mandatory parameter)
    -stopBeforeSwap
    -continue                    (resumes a previously stopped conversion)
    -z <schema-name>
    -t <table-name>
    -ts <tablespace for new table>
    -opt <ADMIN_MOVE_TABLE options>  (e.g. COPY_USE_LOAD)
    …
```

What you see in the DB2 catalog: TABLEORG

- **Which tables are column-organized?**
  - New column in syscat.tables: TABLEORG

```
SELECT tabname, tableorg, compression
FROM   syscat.tables
WHERE  tabname like 'SALES%';

TABNAME                    TABLEORG   COMPRESSION
-------------------------- ---------- -----------
SALES_COL                  C
SALES_ROW                  R          N

  2 record(s) selected.
```

For column-organized tables, COMPRESSION is always blank because you cannot enable/disable compression.

What you see in the DB2 catalog: Synopsis Tables

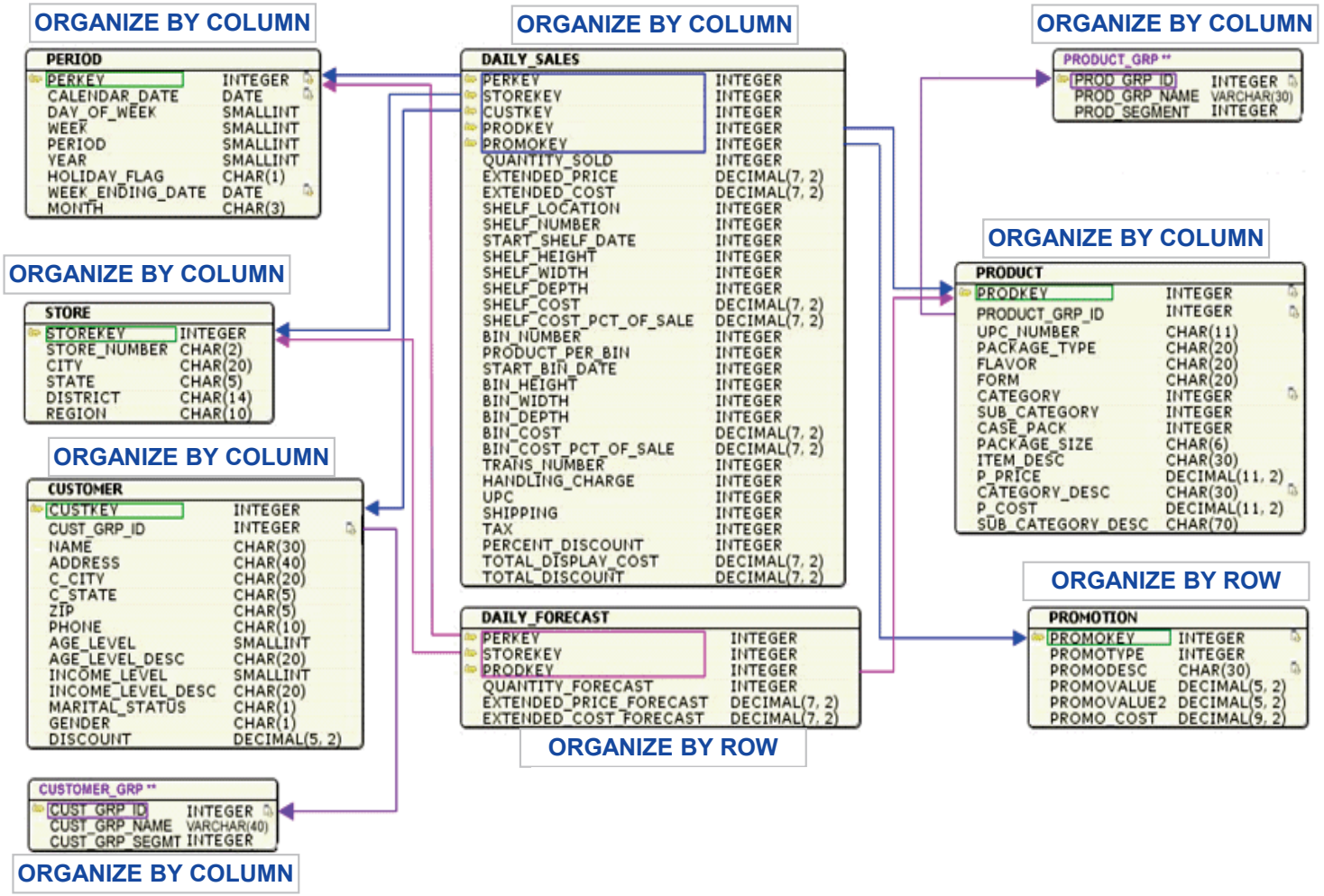
- **For each columnar table there is a corresponding *synopsis table*, automatically created and maintained**
  - Size of the synopsis table: ~0.1% of the user table
  - 1 row for every 1024 rows in the user table

```
SELECT tabschema, tabname, tableorg
FROM syscat.tables
WHERE tableorg = 'C';

TABSCHEMA  TABNAME                                                TABLEORG
---------- ------------------------------------------------------ --------
MNICOLA    SALES_COL                                              C
SYSIBM     SYN13033016521627 5152_SALES_COL                       C

  2 record(s) selected.
```

- **Synopsis Table is a meta-data that describes which ranges of values exist in which parts of the user table**
  - Enables DB2 to skip portions of table when scanning data during query
  - Benefits from data clustering, loading pre-sorted data

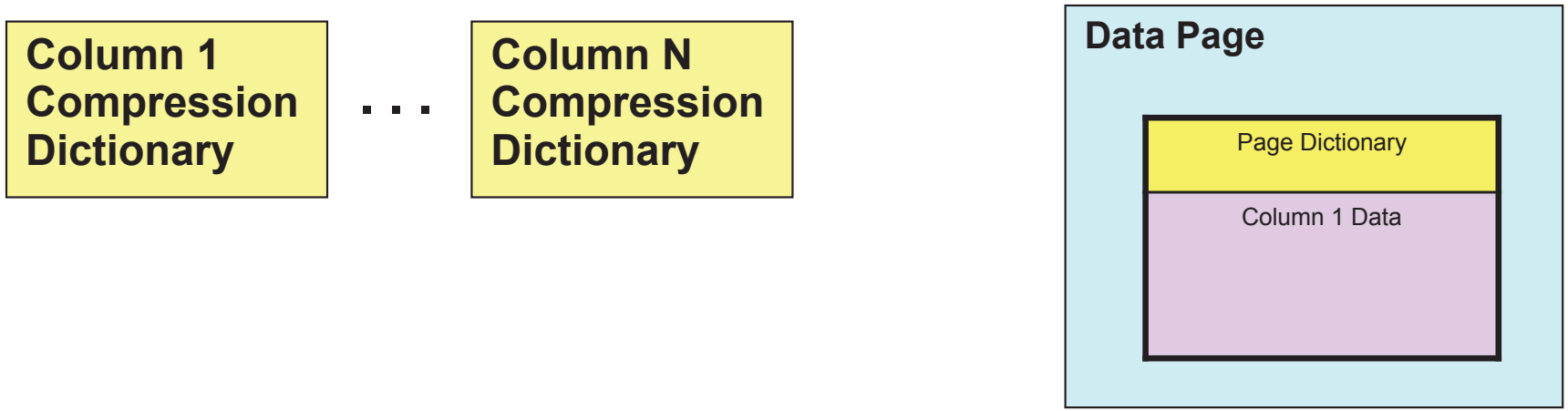
Mixing Row and Columnar Tables

- **DB2 10.5 supports mixing row and columnar tables seamlessly**
  - In the same tablespace and bufferpools
  - In the same query
- **Best query performance for analytic queries usually occurs with all tables columnar**
- **Mixing row and columnar can be necessary**
  - Point queries (highly selective access) favor row-organized tables with index access
  - Small, frequent, write operations favor row-organized tables



LOAD and Compression

Compression Dictionaries for Column-Organized Tables



- **Column-level dictionaries:** **Always one per column**
  - Dictionary populated during load replace, load insert into an empty table, or Automatic Dictionary Creation during Insert
- **Page-level dictionaries:** **May also be created**
  - Exploit local data clustering at page level to further compress data
  - Space savings must outweigh cost of storing page-level dictionaries

Load Example

LOAD FROM /db1/svtdbm1/data.del OF DEL INSERT INTO colTable1;

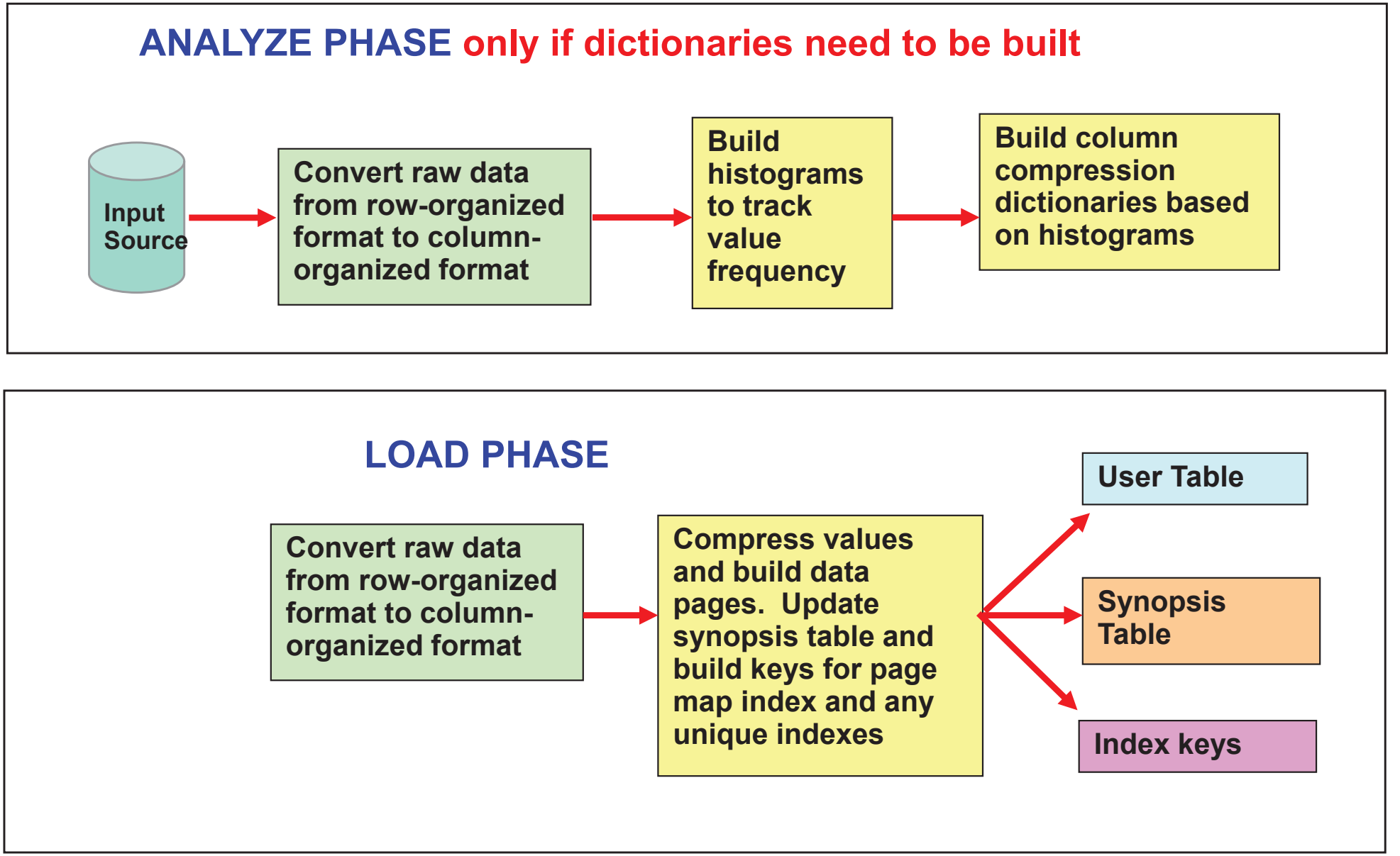
SQL3109N The utility is beginning to load data from file "/db1/svtdbm1/data.del".
SQL3500W The utility is beginning the **"ANALYZE"** phase at time "04/15/2013 14:56:02.272825".
SQL3519W Begin Load Consistency Point. Input record count = "0".
SQL3520W Load Consistency Point was successful.
SQL3515W The utility has finished the **"ANALYZE"** phase at time "04/15/2013 14:56:03.327893".

SQL3500W The utility is beginning the **"LOAD"** phase at time "04/15/2013 14:56:03.332048".
SQL3110N The utility has completed processing. "300000“ rows were read from the input file.
SQL3519W Begin Load Consistency Point. Input record count = "300000".
SQL3520W Load Consistency Point was successful.
SQL3515W The utility has finished the **"LOAD"** phase at time "04/15/2013 14:56:04.639261".

SQL3500W The utility is beginning the **"BUILD"** phase at time "04/15/2013 14:57:06.848727".
SQL3213I The indexing mode is "REBUILD".
SQL3515W The utility has finished the **"BUILD"** phase at time "04/15/2013 14:59:07.487172".

Number of rows read = 300000
Number of rows skipped = 0
Number of rows loaded = 300000
Number of rows rejected = 0
Number of rows deleted = 0
Number of rows committed = 300000

Load for Column-Organized Tables

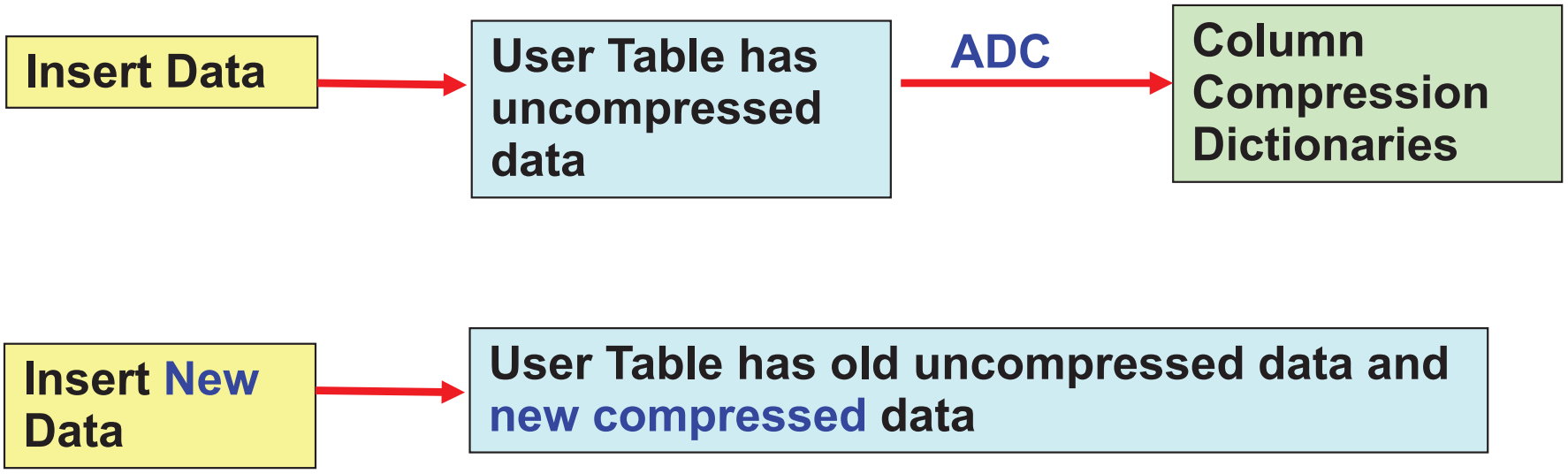


Memory Considerations for Load

- **Faster Load Performance**
- **Better Compressed Tables**
- **Faster Query Performance**

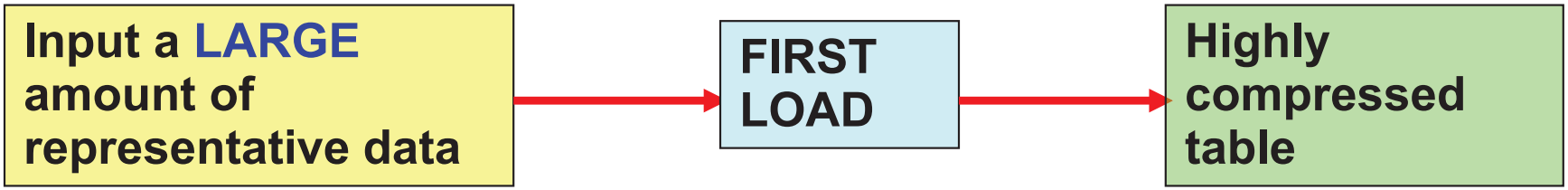
- Load allocates memory from utility heap
- **util_heap_sz** recommendations:
  - At least **1,000,000** pages
  - **4,000,000** pages if database server has >= 128 GB of memory
  - If concurrent utilities need to be run, util_heap_sz should be increased to accommodate higher memory requirements
  - Consider reducing util_heap_sz after load completes to have more SORTHEAP memory for query usage

Inserting into Column-Organized Tables



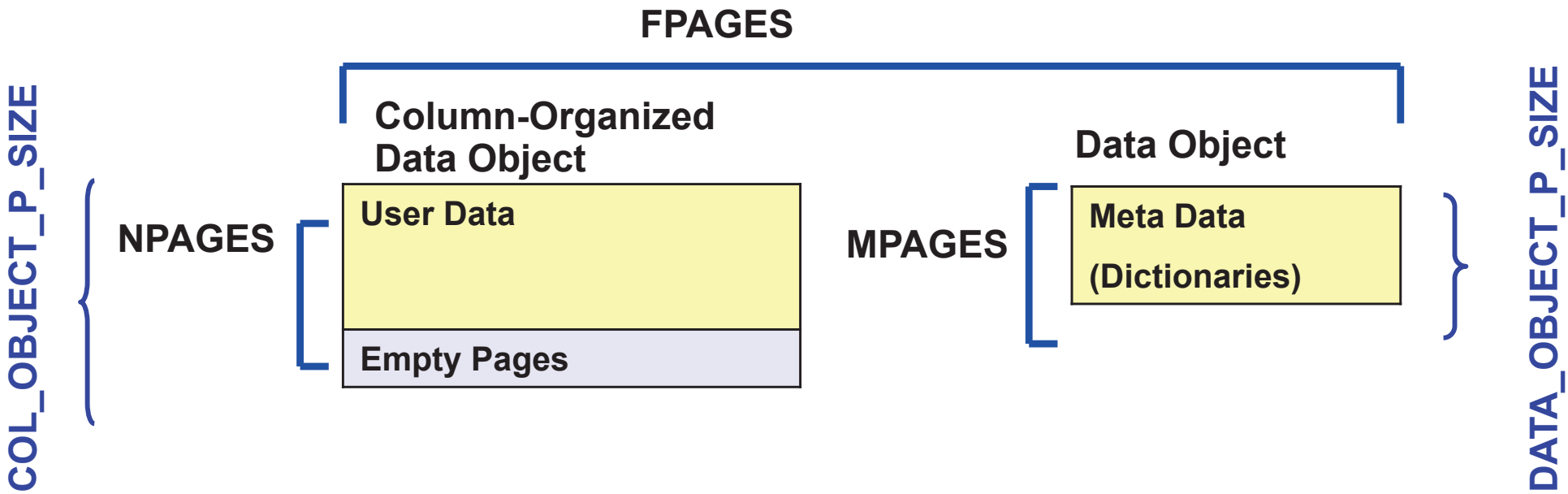
- Initial data inserted before Automatic Dictionary Creation is uncompressed
- When threshold number of values inserted, ADC builds column compression dictionaries
  - Need enough input values to build effective dictionaries
- New values inserted after dictionaries are built are compressed

Recommendations to get Good Compression



- ☑ Load instead of Insert for initial dictionary creation
  - Load utility can analyze more initial data than ADC during Insert and build better column compression dictionaries
  - Values inserted before ADC won't be compressed at the column level
- ☑ Use sufficiently large amount of representative data in 1st Load that builds dictionaries
- ☑ Set util_heap_sz >= 1,000,000 pages
- ☒ Don't load a small initial subset of data for 1st Load

ADMIN_GET_TAB_INFO for Column-Organized Tables



- ADMIN_GET_TAB_INFO table function reports
  - **COL_OBJECT_P_SIZE:** Physical size of column-organized data object containing user data
  - **DATA_OBJECT_P_SIZE:** Physical size of data object containing meta data

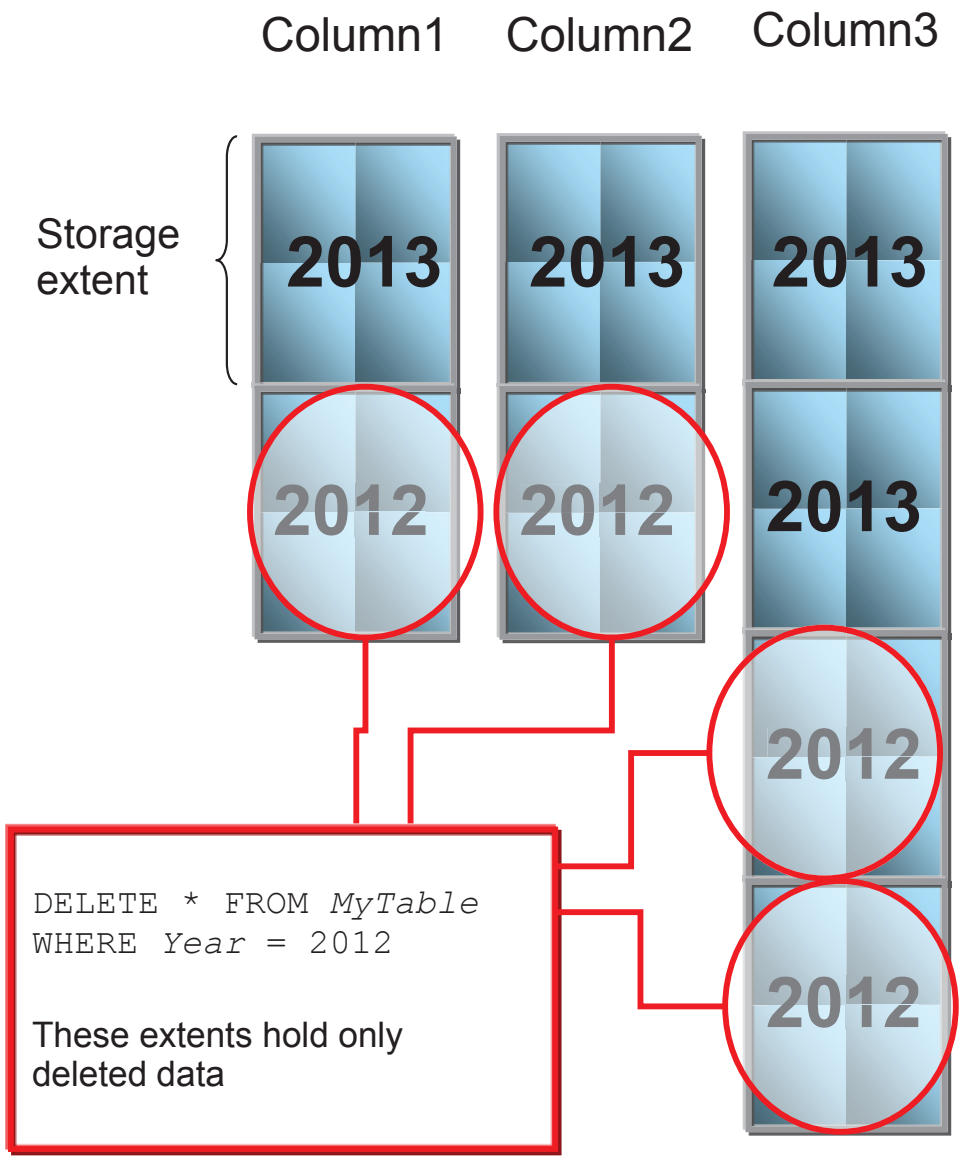
Calculating Column-Organized Storage Sizes

User Table	COL_OBJECT_P_SIZE
User Table +	COL_OBJECT_P_SIZE +
Meta Data +	DATA_OBJECT_P_SIZE +
Page Map/Unique Indexes	INDEX_OBJECT_P_SIZE

- NPAGES is approximate, but doesn't take meta data or empty pages into account
- Use the table function ADMIN_GET_TAB_INFO or admin view ADMINTABINFO to retrieve
  - COL_OBJECT_P_SIZE
  - DATA_OBJECT_P_SIZE
  - INDEX_OBJECT_P_SIZE

Automatic Space Reclaim

- **Automatic space reclamation**
  - Frees extents with no active values
  - The storage can be subsequently reused by any table in the table space
- **No need for costly DBA space management and `REORG` utility**
- **Enabled out-of-the box for column-organized tables when `DB2_WORKLOAD=ANALYTICS`**
- **Space is freed online while work continues**
- **Regular space management can result in increased performance of `RUNSTATS` and some queries**



Reclaiming Space in the Table

- **Objective: Find empty storage extents and return pages to table space for re-use**
- **Option 1: If DB2_WORKLOAD=ANALYTICS, automatic space reclamation is active for all column-organized tables**
- **Option 2: Enable Automatic Table Maintenance (ATM)**

```
update db cfg using auto_maint ON auto_tbl_maint ON auto_reorg ON;
```

- **Option 3: Use REORG TABLE explicitly**

```
                                                  .-ALLOW WRITE ACCESS--.
>>-REORG-TABLE--table-name--RECLAIM EXTENTS--+---------------------+------><
                                             +--ALLOW READ ACCESS--+
                                             '--ALLOW NO ACCESS----'
```