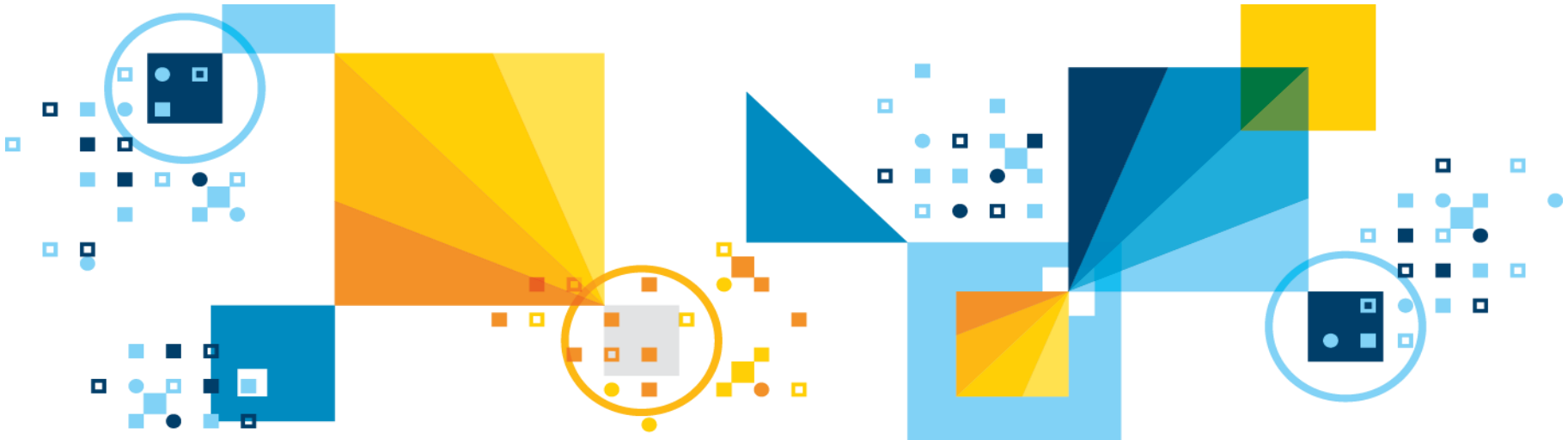


DB2 Essential Maintenance and Autonomics

Module ID | 10104

Length | 1 hour + 1 hour Hands on Lab



For questions about this presentation contact askdata@ca.ibm.com

January 30, 2015

Disclaimer

© Copyright IBM Corporation 2015. All rights reserved.

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

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

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

Module Information

- You should have completed or acquired the necessary knowledge for the following modules in order to complete this module:
 - DB2 Fundamentals

- After completing this module, you should be able to:
 - Describe DB2 Autonomic Features
 - Be able to perform the following tasks:
 - Statistics Collection
 - Data Reorganization
 - Table Space Maintenance
 - Data Movement

Module Content

- **Basic Maintenance & Autonomic Features**

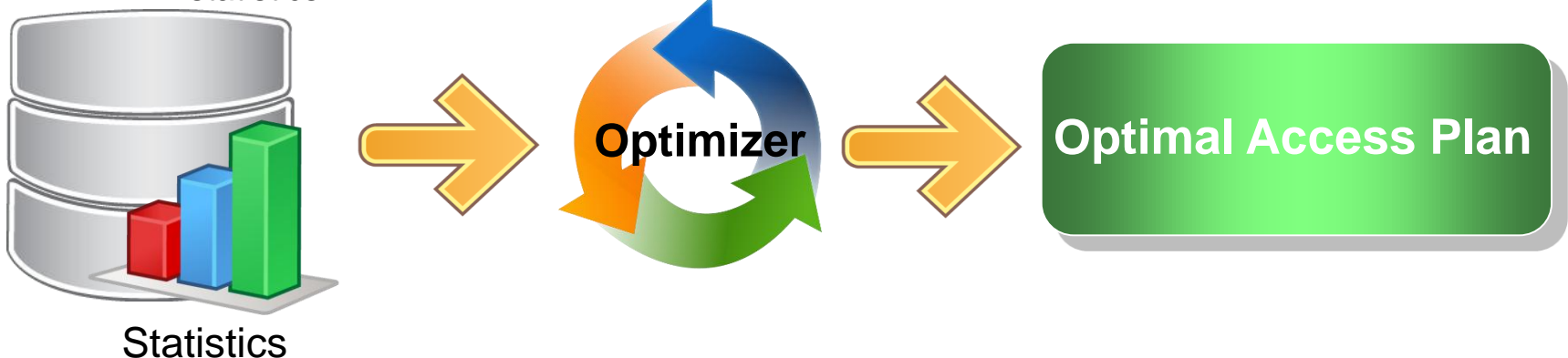
- Statistics Collection
- Data Reorganization
- Table Space Maintenance
- Automatic Maintenance
- Self-Tuning Memory Manager (STMM)
- Advisors

- **Data Movement Methods**

- DDL movement and Data movement
- DB2MOVE Utility
- INGEST Utility

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 or view, along with its associated indexes.
 - Ex. Number of records, number of pages, average record length, etc.
 - 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)
```

- Sampling to collect index statistics

```
db2 RUNSTATS ON TABLE DB2INST1.TABLE1 FOR INDEX INDEX1 INDEXSAMPLE SYSTEM(5)
```

- 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



Statistical Views Enhance Statistics and Statistics Gathering

**NEW in
DB2 10**

- Statistics can be collected from predicates with complex expressions
 - Expression columns – a column with one or more functions
- Reduced the number of statistical views by using referential integrity constraints on the data
 - One view containing many columns from a star join query

- Column group statistics gathered on statistical views
 - Improves access plan

```
runstats on table db2.SV2 on all columns and  
columns ( (C2,D3) ) with distribution;
```

- Automatic statistics collection on statistical views
 - Now, automatic collection function works on views
 - New database configuration parameter, `auto_stats_view`

```
db2 update db cfg for sample using auto_stats_view ON
```

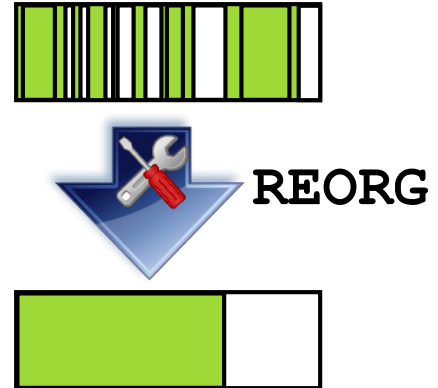
Guidelines for Collecting Statistics

- 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 indicates an online reorg
 - Re-org a **Table**

```
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

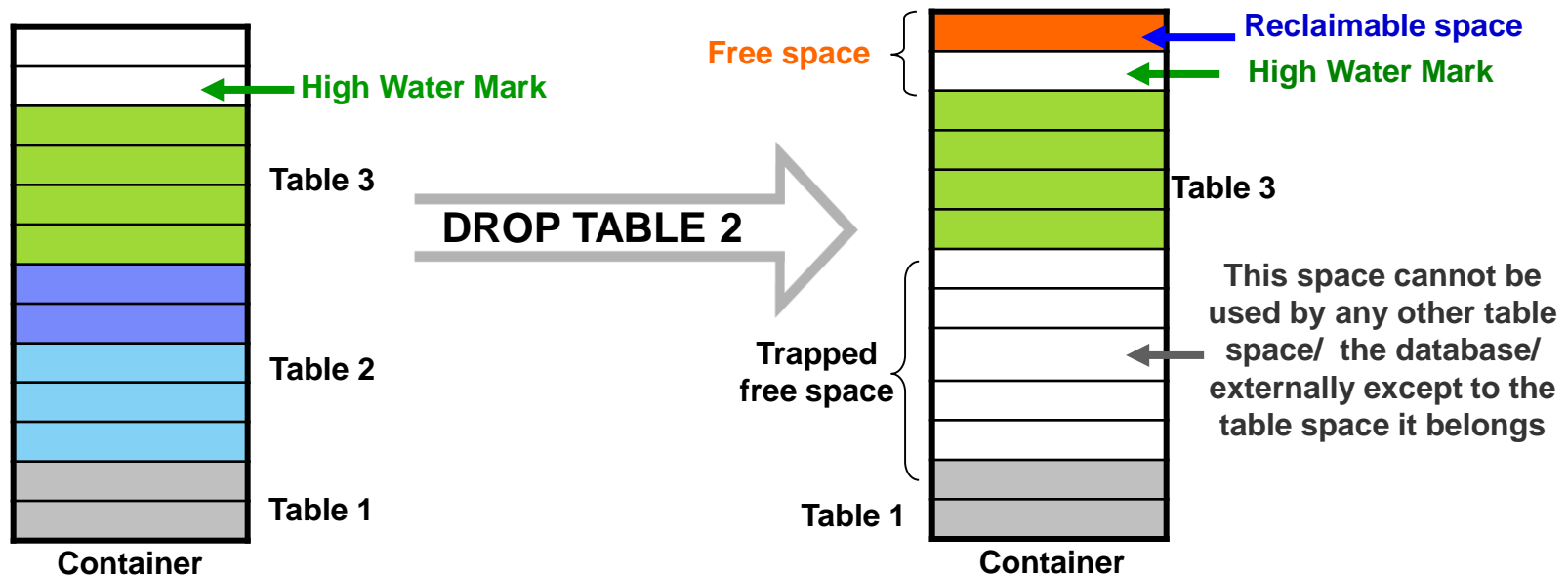
Table space and Storage Management

	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.	<ul style="list-style-type: none"> ▪ Requires that containers be defined as files or devices. ▪ Must specify the initial size for each container. 	<ul style="list-style-type: none"> ▪ 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.	<ul style="list-style-type: none"> ▪ 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.	<ul style="list-style-type: none"> ▪ Containers can be extended or added, reduced or dropped. 	<ul style="list-style-type: none"> ▪ 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 its maximum capacity.	Containers can be extended beyond the initially-allocated size up to constraints imposed by file system.	<ul style="list-style-type: none"> ▪ 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



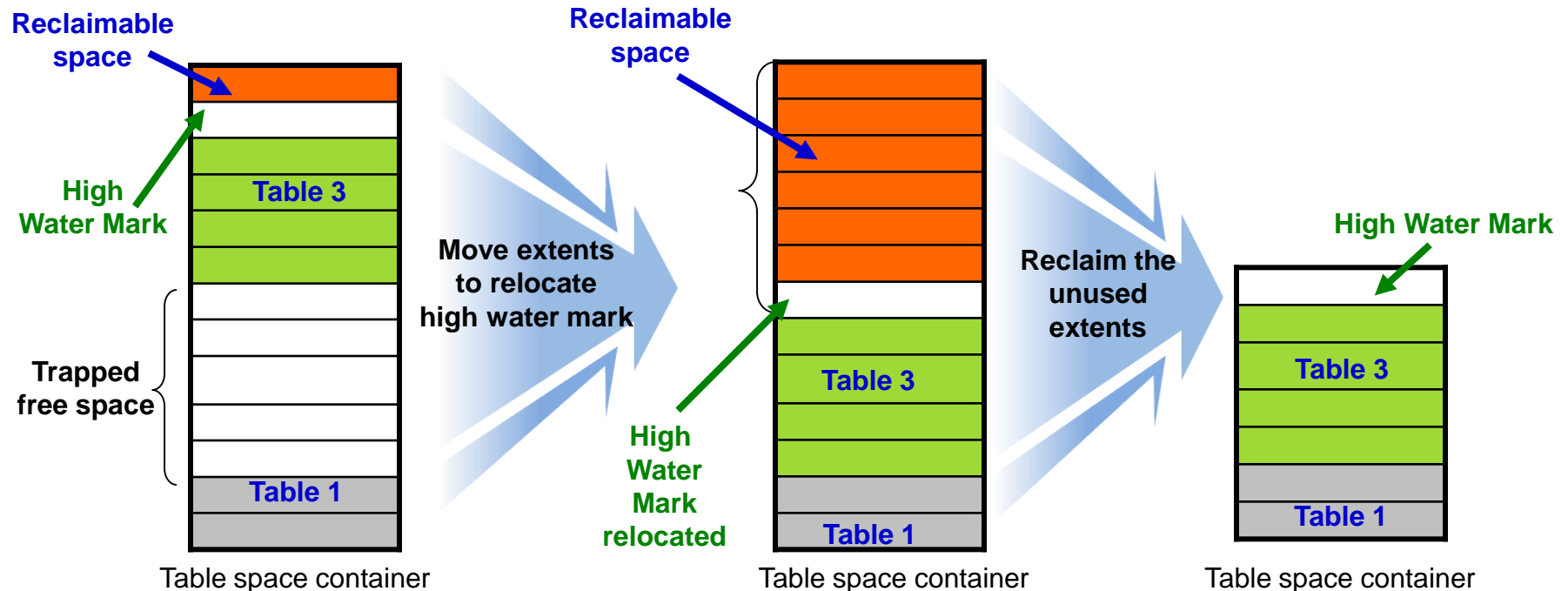
- When an object is dropped, space is freed within the table space.
 - The new free space cannot be shared among different table spaces
 - Only unused extents **above the high water mark** are reclaimable
- **High water mark** refers to the page number of the first page in the extent following the last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

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

Alternatively, specify a
container name



- 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"
```

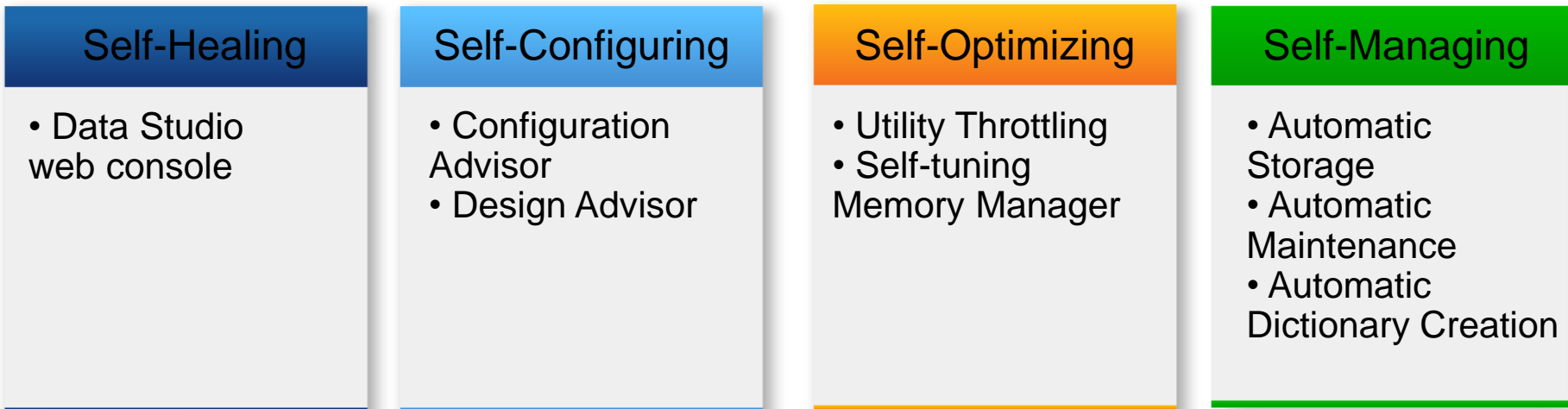
TBSP_NAME	TBSP_FREE_PAGES	TBSP_TOTAL_PAGES	TBSP_PAGE_TOP
-----	-----	-----	-----
EXTENTREMAP	6526	3538280	3536394

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

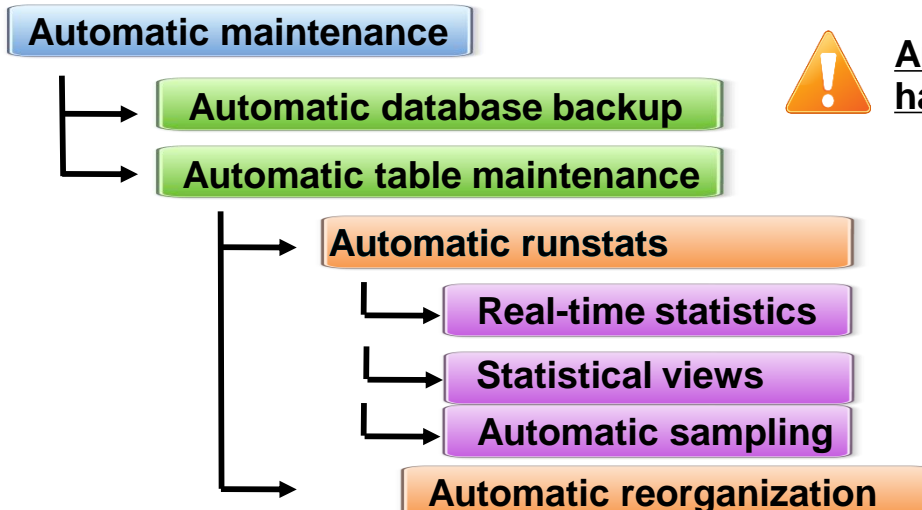


- **Time and cost savings**
 - It shifts the burden of managing a computing environment from DBAs to technology
 - Over 70% of IT budgets are consumed by labor costs

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/sql/lib/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**

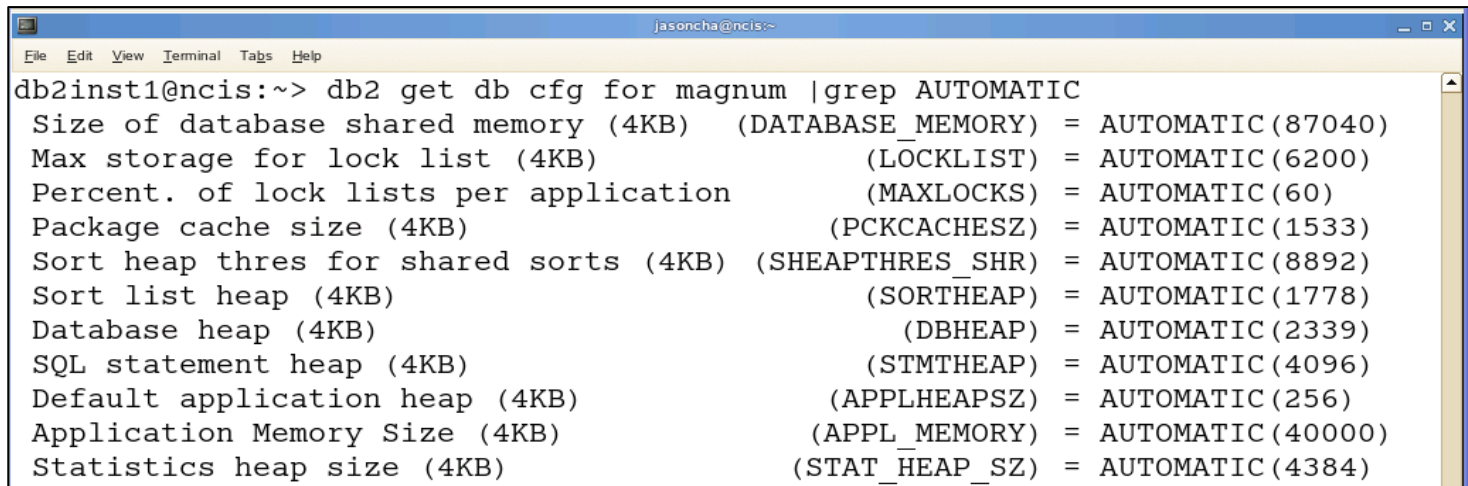
- A backup policy may be used to specify the automated behavior

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

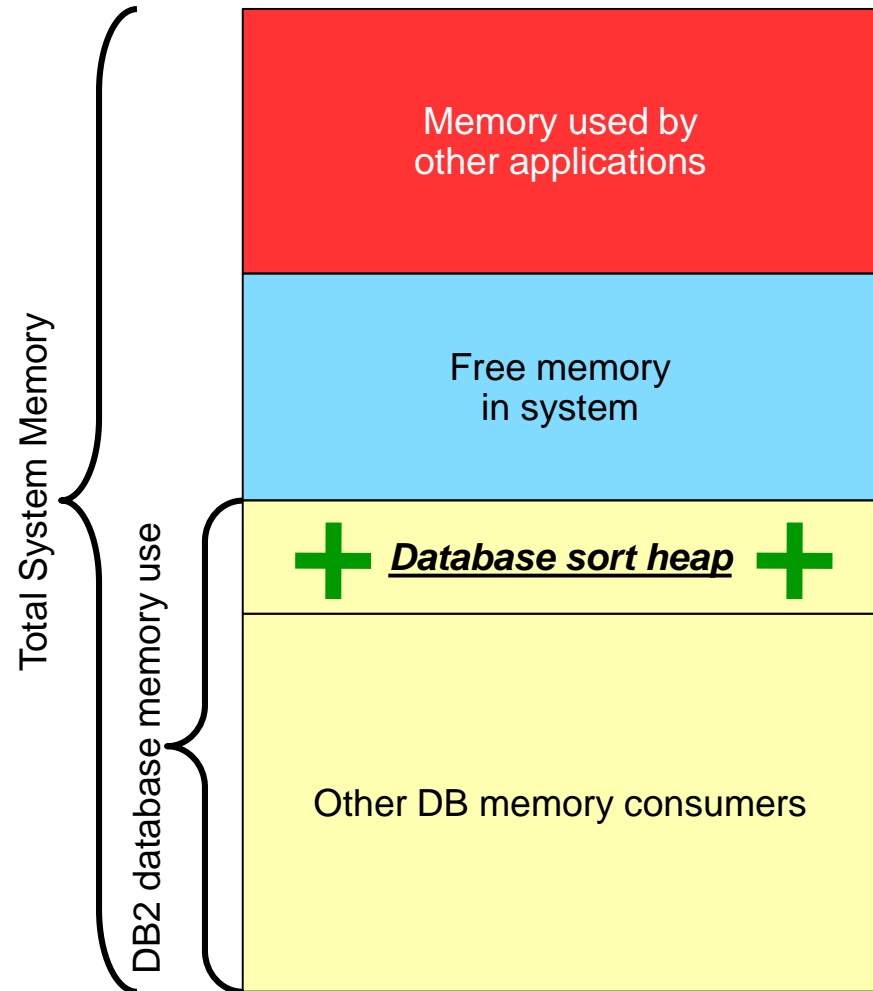


```
db2inst1@ncis:~> db2 get db cfg for magnum |grep AUTOMATIC
Size of database shared memory (4KB) (DATABASE_MEMORY) = AUTOMATIC (87040)
Max storage for lock list (4KB) (LOCKLIST) = AUTOMATIC (6200)
Percent. of lock lists per application (MAXLOCKS) = AUTOMATIC (60)
Package cache size (4KB) (PCKCACHESZ) = AUTOMATIC (1533)
Sort heap thres for shared sorts (4KB) (SHEAPTHRES_SHR) = AUTOMATIC (8892)
Sort list heap (4KB) (SORTHEAP) = AUTOMATIC (1778)
Database heap (4KB) (DBHEAP) = AUTOMATIC (2339)
SQL statement heap (4KB) (STMTHEAP) = AUTOMATIC (4096)
Default application heap (4KB) (APPLHEAPSZ) = AUTOMATIC (256)
Application Memory Size (4KB) (APPL_MEMORY) = AUTOMATIC (40000)
Statistics heap size (4KB) (STAT_HEAP_SZ) = AUTOMATIC (4384)
```

STMM Operating Modes

- DATABASE_MEMORY = **AUTOMATIC**

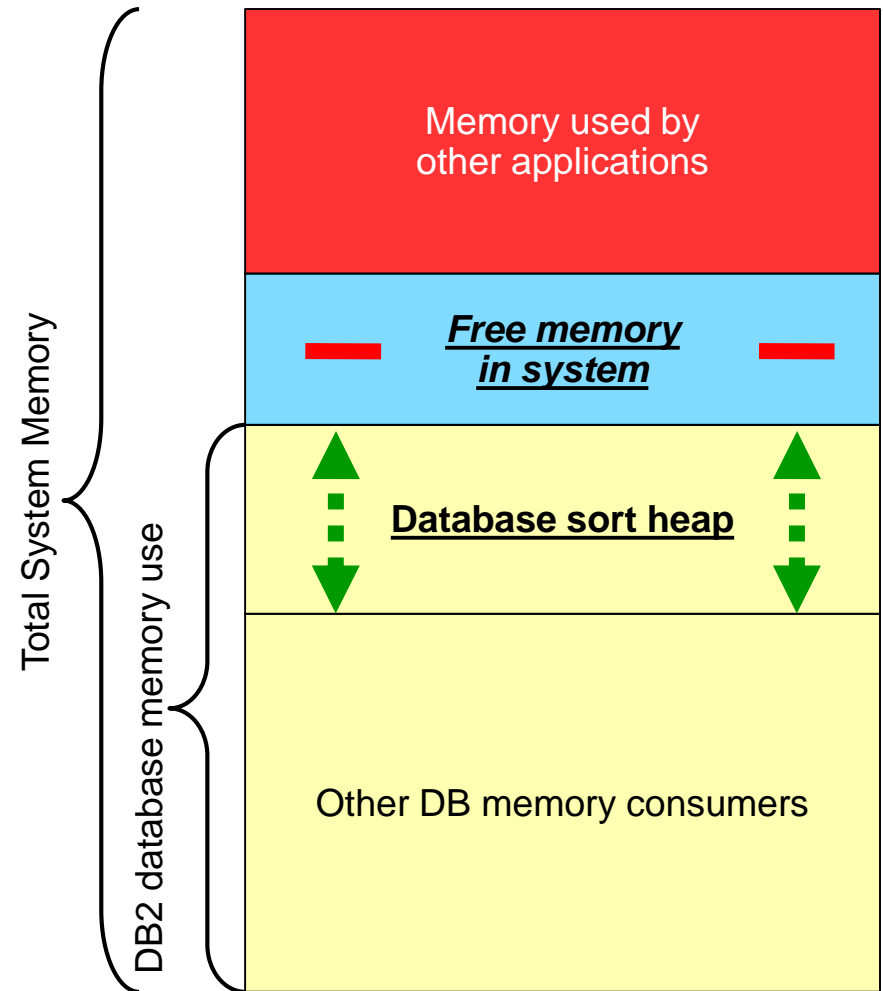
1) A change in workload occurs that requires more memory for sorts



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

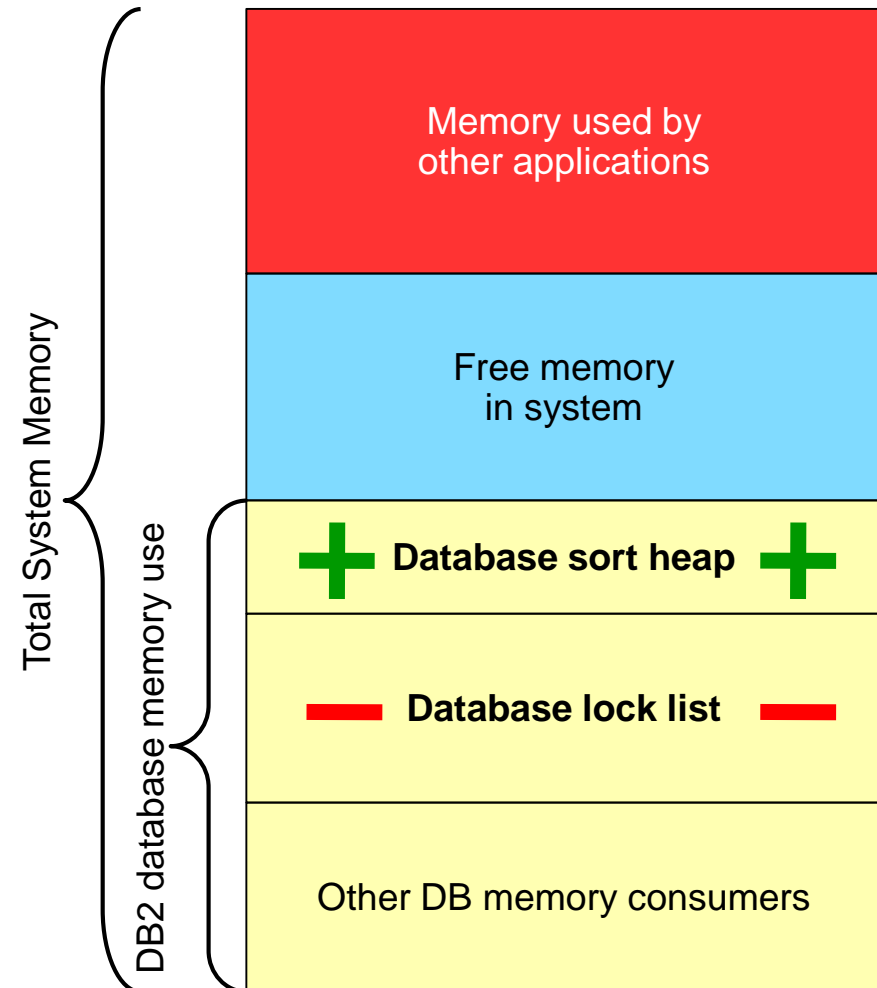


STMM Operating Modes

- DATABASE_MEMORY = **COMPUTED** or <number>

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

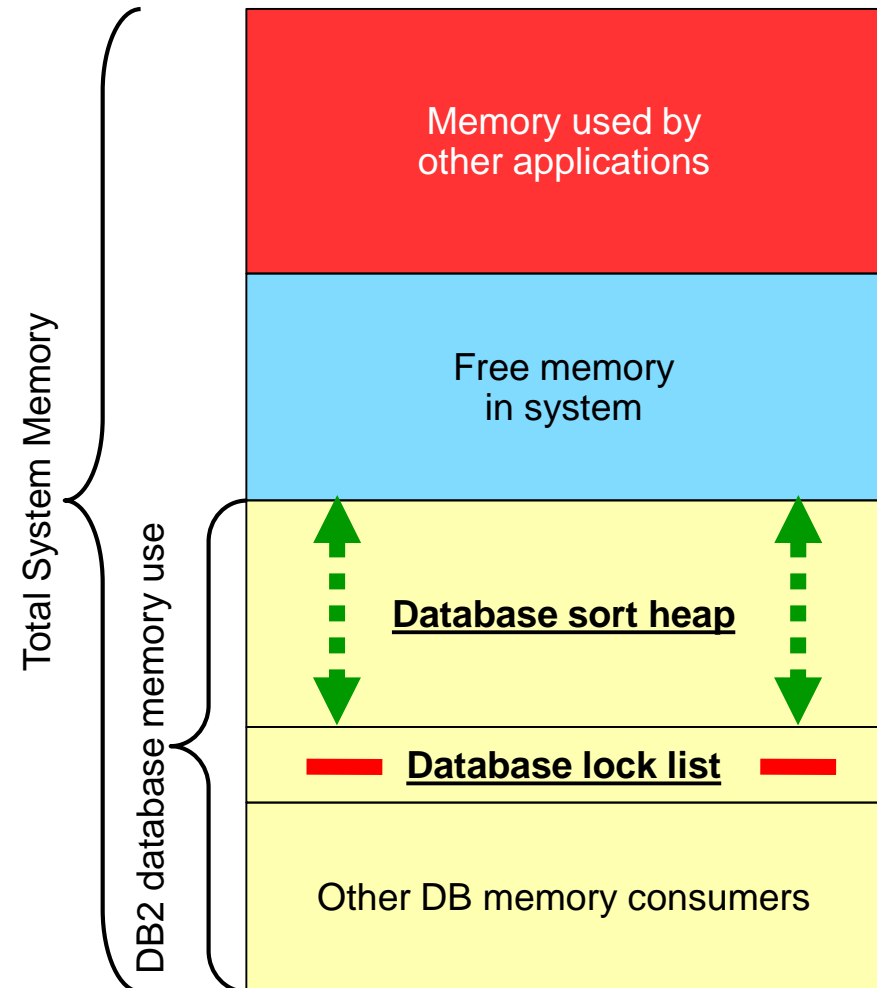
2) DB2 database is set at fixed memory usage, and thus cannot take memory from OS. Therefore identifies another memory consumer in the database that does not need its memory anymore (e.g. lock list)



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

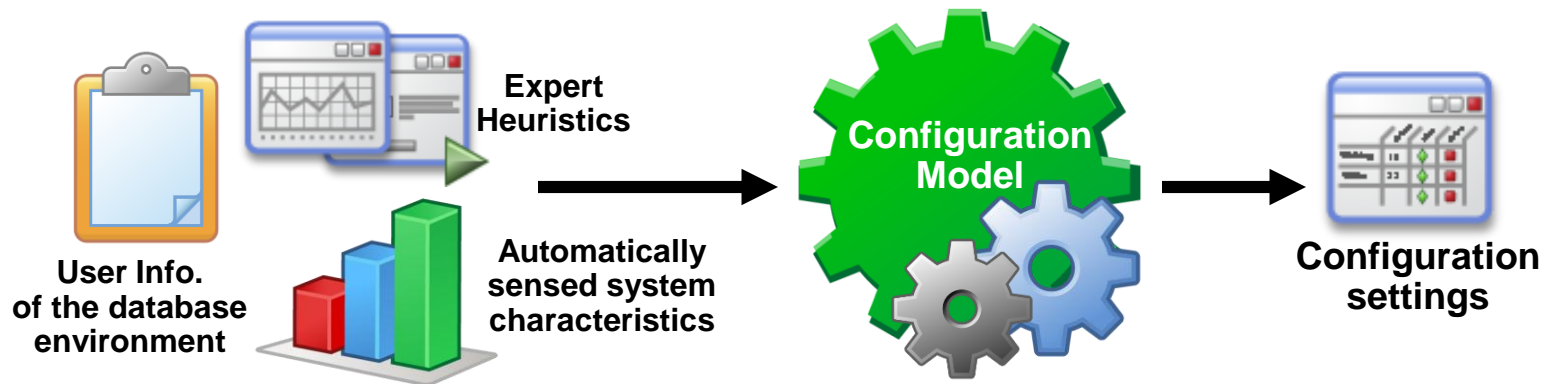
▪ Utility Throttling

- 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.

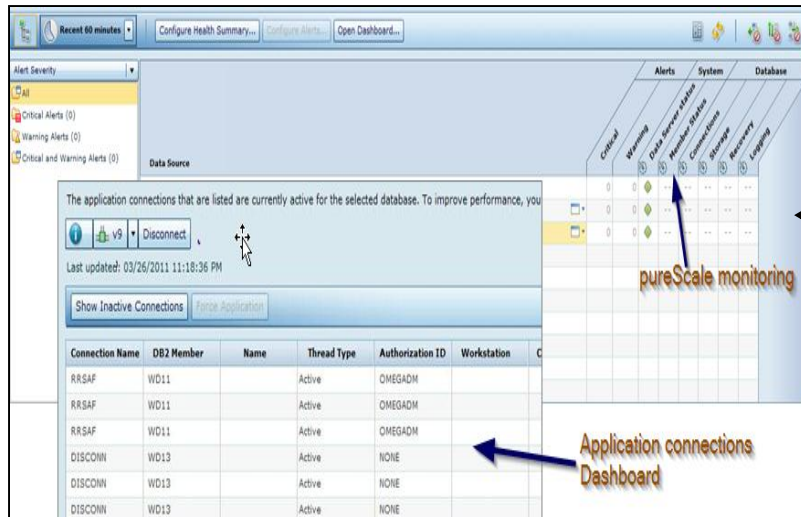
Design Advisor

- 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 **db2adv** 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

Job List Schedules Notifications History								
Instructions text goes here...								
New Schedule... Open  								
Schedule ID	Interval	Start Time	Database	▲	Job ID	Job Name	Chain	Notifications
14522-2	Monthly Every Third Wednesday	3:00 PM GMT		1	14522	Cleanup Job		1
14511-3	Weekly Every Monday	12:00 AM GMT		10	14511	Backup Job		1
14510-2	Daily	12:00 AM GMT		1	14510	Backup Job		2
14509-1	Daily	1:00 AM GMT		1	14509	Cleanup Job		0
14508-1	Weekly Every Tuesday	1:00 AM GMT		0	14508	Cleanup Job		0

Module Content

- **Basic Maintenance & Autonomic Features**

- Statistics Collection
- Data Reorganization
- Table Space Maintenance
- Automatic Maintenance
- Self-Tuning Memory Manager (STMM)
- Advisors

- **Data Movement Methods**

- DDL movement and Data movement
- DB2MOVE Utility
- INGEST Utility

Data Movement Methods

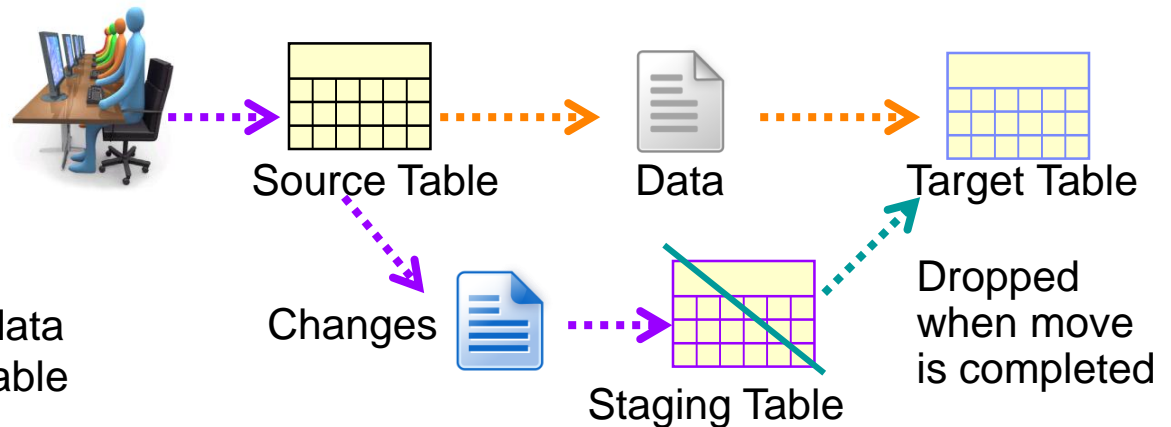
▪ Commands and utilities

- **DB2MOVE** : EXPORT, IMPORT, LOAD and COPY option
New: parallel processing for COPY is supported!
- **DB2LOOK** : Extract DDL statements
- **EXPORT** : moves data out of database into flat files
- **IMPORT** : inserts data into database from flat files
- **LOAD** : fast method to load large amounts of data
- **LOAD QUERY** : check state/phase of LOAD operation or state of a table
- **INGEST** : high-speed client-side DB2 utility that streams data from files or pipes into DB2 target tables, without affecting normal workload

▪ Stored Procedures

- **ADMIN_MOVE_TABLE**: move a table, online or offline
- **ADMIN_COPY_SCHEMA**: copy a specific schema and all objects contained in it and copy tables with or without the data of the original tables

Table Maintenance Procedure – ADMIN_MOVE_TABLES



- The **ADMIN_MOVE_TABLE** stored procedure moves the data in an active table into a new table
- Allows data to remain online and available for access.
- Two ways to use the procedure:
 - Move table while modifying certain parts of the table definition for the target table

```
CALL SYSPROC.ADMIN_MOVE_TABLE (... , 'MOVE')
```

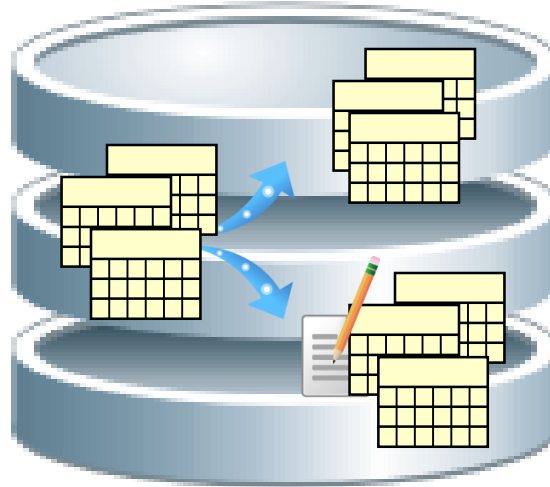
- Move table with greater control on table creation by allowing you to create the target table beforehand.

```
CREATE TABLE (COL1 VARCHAR(5))
```

```
CALL SYSPROC.ADMIN_MOVE_TABLE (... , 'MOVE')
```


Table Maintenance Procedure – ADMIN_COPY_SCHEMA

- The **ADMIN_COPY_SCHEMA** procedure is used to copy a specific schema and its objects.



- Procedure can be run in 3 different copy modes:
 - 'DDL': create empty copies of all supported objects from the source.
 - 'COPY': Create copies and load data from the source tables to target. LOAD is done in NONRECOVERABLE mode.
 - 'COPYNO': LOAD is done in 'COPY NO' mode.

```
CALL SYSPROC.ADMIN_COPY_SCHEMA  
('SOURCE_SCHEMA', 'TARGET_SCHEMA', 'COPY', NULL,  
 'SOURCETBSP1', 'TARGETTBSP1', 'ERRORTABSCHEMA', 'ERRORTABNAME')
```

DB2LOOK, Export, Import and LOAD Utilities

▪ DB2LOOK

- Extract DDL statements Object by Object

```
db2look -d department -wlm -e -l
```

▪ DB2 Export

- Extract data from table or view to files
- Invoke from Data Studio, CLP, or the db2Export API via a user application

▪ DB2 Import

- Import data from a file using SQL inserts
- Invoke from Data Studio, CLP, or the db2Import API via a user application

▪ DB2 LOAD

- Move large quantities of data efficiently into tables
- 4 distinct phases: LOAD, BUILD, DELETE, INDEX COPY
- With CURSOR file type
 - Load results of an SQL query directly into a target table
 - No intermediate storage of data needed
 - LOAD can be used across multiple databases in a CURSOR

Writes formatted
pages directly into
the database

```
DECLARE C1 CURSOR DATABASE SRCDB USER user1 USING password FOR  
SELECT * FROM SOURCE_TABLE;  
LOAD FROM C1 OF CURSOR REPLACE INTO TARGET_TABLE;
```

DB2MOVE command

- Utilizes the EXPORT/IMPORT/LOAD/COPY APIs to move large number of tables or entire schemas.
- Options
 - **EXPORT**
 - Exports all tables that meet the filtering criteria according to the option specified.
 - Internal staging information is stored in the db2move.lst file
 - **IMPORT**
 - Imports all tables listed in the db2move.lst internal staging file
 - **LOAD**
 - Loads all tables listed in the internal staging file db2move.lst
 - **COPY**
 - Duplicates schemas into a target database.

```
db2move SRCDB COPY -sn "SANTA" -co target_db TESTDB -u db2admin -p password
```

INGEST utility

- **High-speed client-side DB2 utility that ingests data from files and pipes into DB2 LUW tables, using SQL-like commands.**



- **Advantages:**
 - Move and process large amounts of **real-time data** **without affecting availability**
 - Allows decisions to be based on the latest set of data
 - Increases data analysis capabilities
 - DPF-aware: routes rows to correct partition

No need to choose between data concurrency and availability!



Supported Data Inputs

- **Input data formats:**

- **Delimited ASCII (DEL) Format**

- Stream of characters separated by row and column delimiters

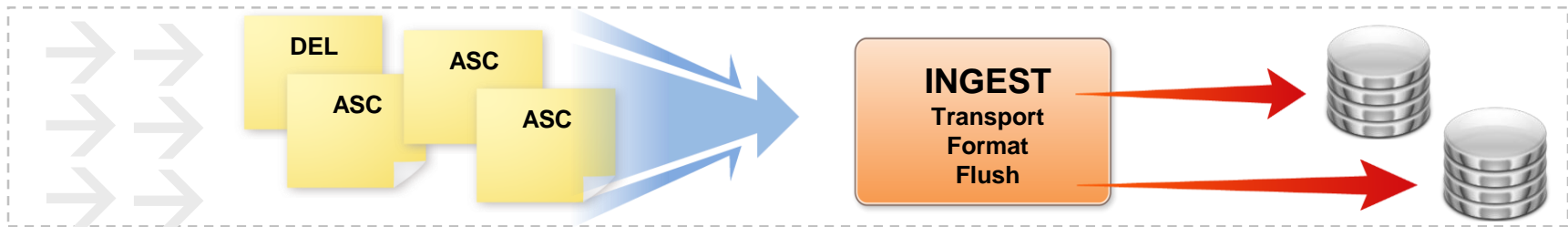
```
1, "Mark Kate", "DB2 Specialist", "IBM Canada"  
2, "John Doe", "DB2 Specialist", "IBM Canada"  
3, "Steven Johns", "Manager", "IBM London"
```

- **Fixed Format ASCII (ASC)**

- Each column length in the file has the same length of the column definition

1	Mark Kate	DB2 Specialist	IBM Canada
2	John Doe	DB2 Specialist	IBM Canada
3	Steven Johns	Manager	IBM London

- Continuously pumps data into DB2 tables using SQL arrays until the source is exhausted



Supported Operations

▪ DML Operations

- INSERT, UPDATE, DELETE, MERGE (REPLACE option available)

```
INGEST FROM FILE my_file.del FORMAT DELIMITED
```

```
( $key_fld1 INTEGER EXTERNAL,  
  $key_fld2 INTEGER EXTERNAL,  
  $data_fld1 CHAR(8),  
  $data_fld2 CHAR(8),  
  $data_fld3 CHAR(8) )
```

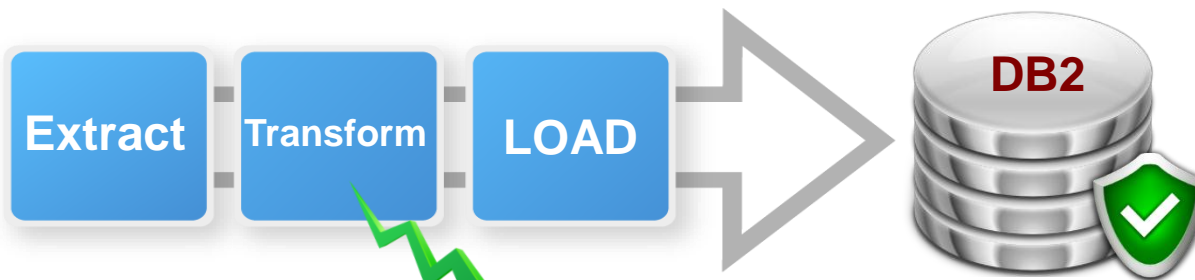
```
UPDATE my_table SET (data1_col, data2_col, data3_col) = ($data_fld1,  
$data_fld2, $data_fld3)
```

```
WHERE (key_col1 = $key_fld1) AND (key_col2 = $key_fld2);
```

**IMPORT and
LOAD DO NOT
support data
transformation**

▪ Lightweight ETL (Extract, Transform, Load)

- SQL expressions including basic predicates and casting ✓



Data can be transformed by the
INGEST utility using SQL expressions.

INGEST Command Examples

- Example 1: basic command

```
INGEST FROM FILE my_file.txt FORMAT DELIMITED INSERT INTO my_table;
```

- Example 2: file using a POSITIONAL format with fields in fixed positions

```
INGEST FROM FILE my_file.txt FORMAT POSITIONAL(  
    $field1 POSITION(1:8) INTEGER EXTERNAL,  
    $field2 POSITION(10:19) DATE 'yyyy-mm-dd',  
    $field3 POSITION(25:34) CHAR(10))  
INSERT INTO my_table VALUES($field1, $field2, $field3);
```

- Example 3: pipe using DEL format and a customized delimiter

```
INGEST FROM PIPE mypipe FORMAT DELIMITED BY '/' (  
    $prod_ID CHAR(8),  
    $description CHAR(32),  
    $price DECIMAL(5,2) EXTERNAL,  
    $sales_tax DECIMAL(4,2) EXTERNAL,  
    $shipping DECIMAL(3,2) EXTERNAL )  
INSERT INTO my_table(prod_ID, description, total_price)  
    VALUES($prod_id, $description, $price + $sales_tax + $shipping);
```

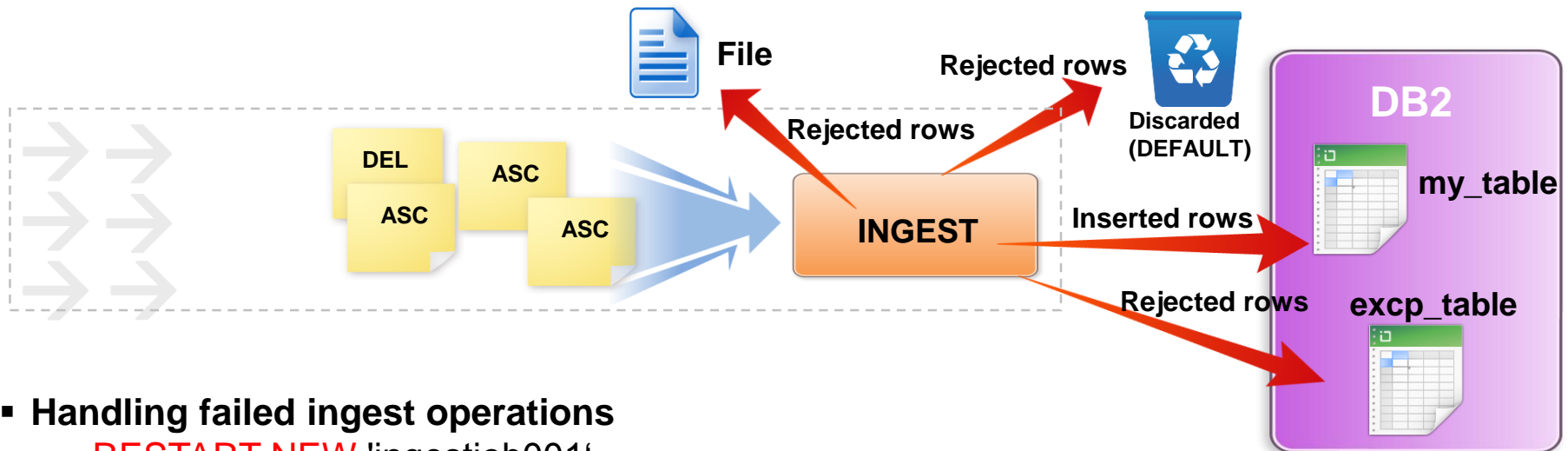
Custom delimiter

Data Transformation during Ingest

Recoverability

- Rejected rows can be discarded or placed into a file or table.

```
INGEST FROM FILE my_file.txt FORMAT DELIMITED (
    $field1 INTEGER EXTERNAL,
    $field2 DATE 'mm/dd/yyyy',
    $field3 CHAR(32) )
EXCEPTION TABLE excp_table MESSAGES messages.txt
INSERT INTO my_table VALUES($field1, $field2, $field3);
```



- Handling failed ingest operations

- **RESTART NEW** 'ingestjob001'
- **RESTART CONTINUE** 'ingestjob001'
- **RESTART TERMINATE** 'ingestjob001'



Comparison Between INGEST, LOAD and IMPORT

▪ Supported Table Types

Object	INGEST	LOAD	IMPORT
Detached table	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Global Temporary Table	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Multidimensional clustering table	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Materialized query table (MQT) that is maintained by user	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Nickname	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Range-clustered table	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Range-partitioned table	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Summary table	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Typed table	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Untyped (regular) table	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Updatable view (except typed view)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

When To Use INGEST

- Use **INGEST** when any of the following is true:
 - You need other applications to update the table while it is being loaded.
 - The input file contains fields you want to skip over.
 - You need to specify an SQL statement other than INSERT.
 - You need to specify an SQL expression (to construct a column value from field values).
 - You need to recover and continue on when the utility gets a recoverable error.
- **Performance vs. LOAD:**
 - **50% faster** than to load into a staging table *followed by multiple sequential* INSERT/SELECTs from the staging table to the target table**

****Tests were run inserting into a table with:**

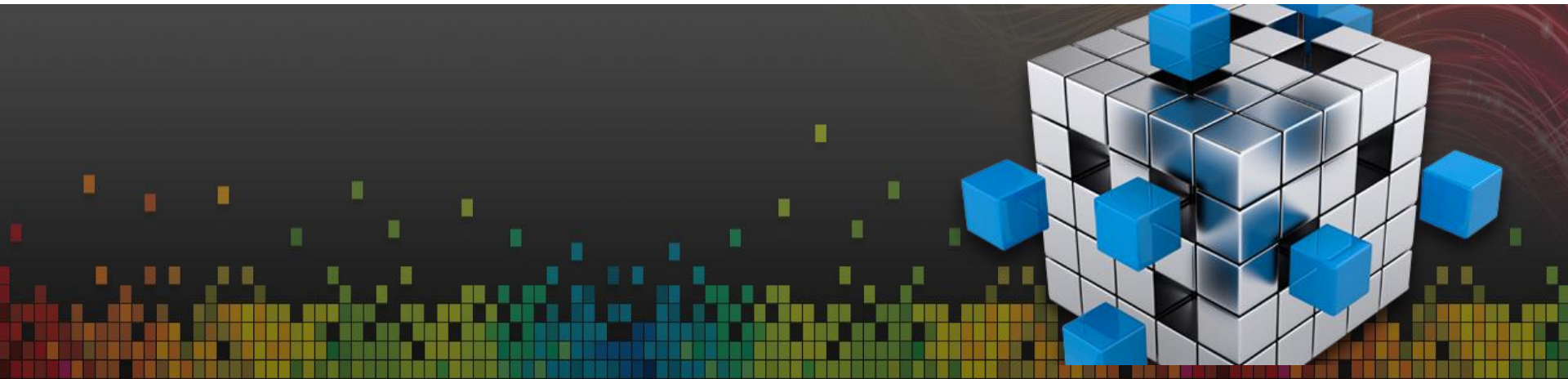
- 100 byte rows
- 4 indexes
- an 8 partition D or E class BCU with a 4 core admin node and 2 data nodes with 4 cores each
- From 30 million to 500 million rows

Summary

- **DB2 autonomic features integrated in many forms to help DBAs**
 - Automatic Storage
 - Automatic Object Maintenance: Auto - Backup, Reorg, Runstats etc.
 - STMM
 - So on...

- **INGEST utility meets modern DW requirements by processing data from a continuous data stream**
 - Minimizes impact to concurrent use with no table locking
 - Performs data transformation which is not possible with IMPORT or LOAD
 - Unwanted rows can be placed into an exception file or table, or just discarded

The next steps...



The Next Steps...

- Complete the Hands on Lab for this module
 - Log onto SKI, go to “My Learning” page, and select the “In Progress” tab.
 - Find the module
 - Download the workbook and the virtual machine image
 - Follow the instructions in the workbook to complete the lab
- Complete the online quiz for this module
 - Log onto SKI, go to “My Learning” page, and select the “In Progress” tab.
 - Find the module and select the quiz
- Provide feedback on the module
 - Log onto SKI, go to “My Learning” page
 - Find the module and select the “Leave Feedback” button to leave your comments



An abstract graphic design featuring a horizontal arrangement of various geometric shapes and patterns. The color palette is primarily blue, yellow, and orange. On the left, there's a cluster of small squares and circles, some grouped within a light blue circle. A large yellow square with diagonal stripes is prominent in the center-left. To its right, a blue triangle points towards the center. Further right, a large blue square contains a smaller yellow square with diagonal stripes. On the far right, a dark blue square with a light blue circle and square inside is circled in light blue. The overall composition is dynamic and modern, with elements overlapping and scattered across the white background.