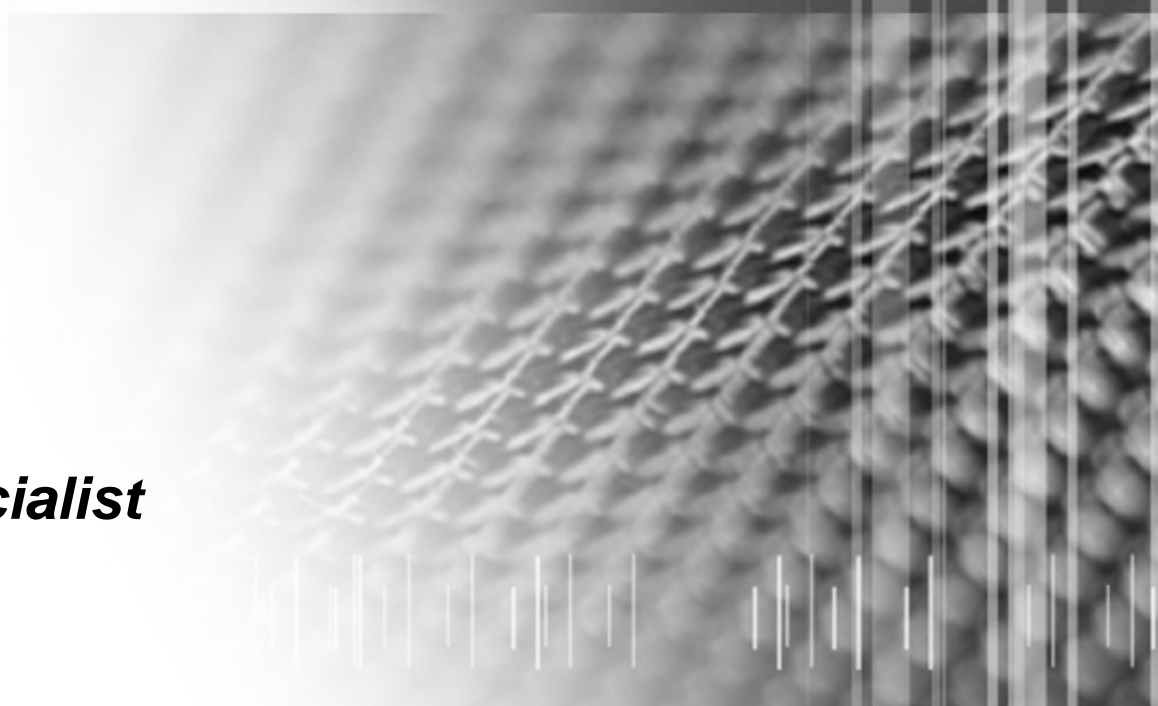




IBM Software Group

# DB2 Viper Technology Preview

***Lonnie McCloud – SW IT Specialist***  
***lonniem@us.ibm.com***



**ON DEMAND BUSINESS**

# Disclaimer/Trademarks

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements, or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

**All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.**

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious, and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

## Trademarks

The following terms are trademarks or registered trademarks of other companies and have been used in at least one of the pages of the presentation:

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both: AIX, AS/400, DataJoiner, DataPropagator, DB2, DB2 Connect, DB2 Extenders, DB2 OLAP Server, DB2 Universal Database, Distributed Relational Database Architecture, DRDA, eServer, IBM, IMS, iSeries, MVS, Net.Data, OS/390, OS/400, PowerPC, pSeries, RS/6000, SQL/400, SQL/DS, Tivoli, VisualAge, VM/ESA, VSE/ESA, WebSphere, z/OS, zSeries

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

# Agenda

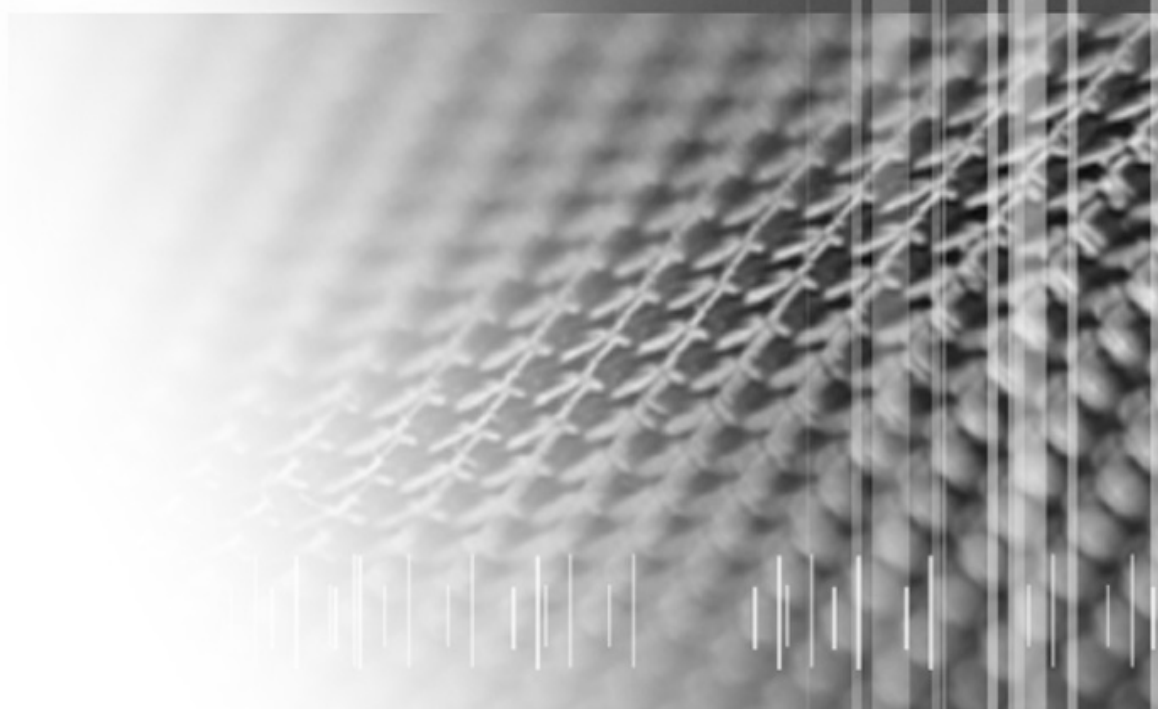
- XML Support
- Autonomic Enhancements and Administration Improvements
- Table Partitioning
- Larger Table spaces
- Label-based Access Control
- Row Compression
- SQL Enhancements



IBM Software Group

# XML Technology

*Enabling Pure XML in DB2*



**ON** DEMAND BUSINESS™

# What is XML?

## ■ XML Technology

- ▶ XML = Extensible Markup Language
- ▶ Self-describing data structures
- ▶ XML Tags describe each element and their attributes

## ■ Benefits

- ▶ Extensible
  - No fixed format or syntax
  - Structures can be easily changed
- ▶ Platform Independent
  - Not tied to any platform, operating system, language or software vendor
  - XML can be easily exchanged
- ▶ Fully Unicode compliant

```
<? xml version="1.0" ?>
<purchaseOrder id='12345' secretKey='4x%$^'>
  <customer id="A6789">
    <name>John Smith Co</name>
    <address>
      <street>1234 W. Main St</street>
      <city>Toledo</city>
      <state>OH</state>
      <zip>95141</zip>
    </address>
  </customer>
  <itemList>
    <item>
      <partNo>A54</partNo>
      <quantity>12</quantity>
    </item>
    <item>
      <partNo>985</partno>
      <quantity>1</quantity>
    </item>
  </itemList>
</purchaseOrder>
```

# XML Example: Financial Data (FIXML)

- Buying 1000 Shares of IBM Stock..

8=FIX.4.2^9=251^35=D^49=AFUNDMGR^56=ABROKER^34=2  
 ^52=20030615-01:14:49^11=12345^1=111111^63=0^64=2003  
 0621^21=3^110=1000^111=50000^55=IBM^48=459200101^22=  
 1^54=1^60=2003061501:14:4938=5000^40=1^44=15.75^15=USD  
 ^59=0^10=127

Old FIX  
Protocol

New FIXML  
Protocol

- extensible
- lower appl development & maintenance cost

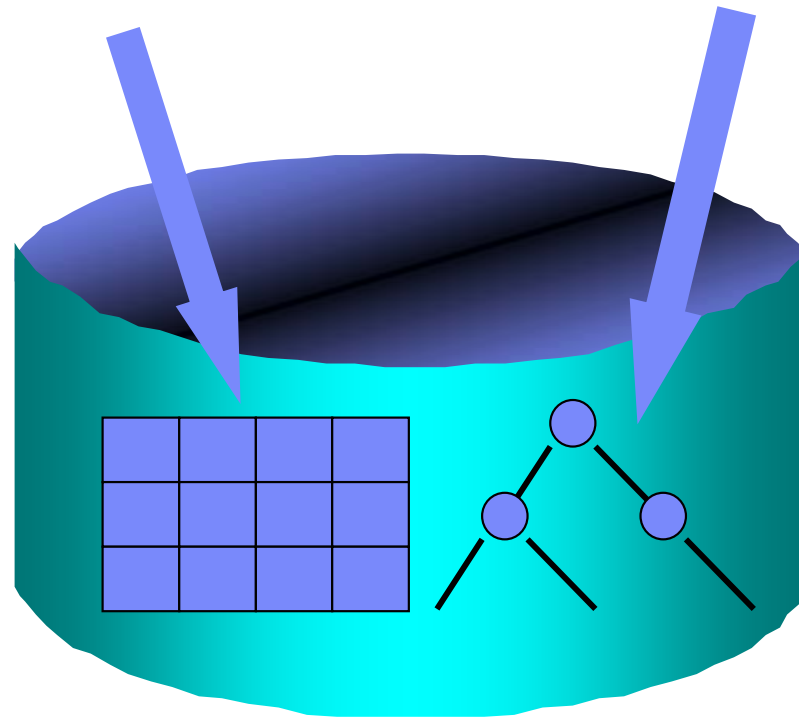
```
<FIXML >
  <NewOrdSingle  C1OrdID ="123456"
    Side ="2"
    TransactIm ="2003 -06 -15T01:14:49 -05:00"
    OrderType ="2"
    Price ="93.25"
    Acct ="26522154">
    <Header  Sent ="2001 -06 -21T01:31:28 -05:00"
      PosDup ="N"
      PosRsnd ="N"
      SeqNum ="521">
      <Sender  ID ="AFUNDMGR"/>
      <Target  ID ="ABROKER"/>
    </Header >
    <Instrument  Symbol ="IBM"
      ID ="459200101"
      IDSrc ="1"/>
    <OrderQuantity  Qty ="1000"  Cur ="USD"/>
  </NewOrdSingle >
</FIXML >
```

# Native XML Storage

- Must store XML in parsed hierarchical format (similar to the DOM representation of the XML infoset)

```
create table dept (deptID char(8), ..., deptdoc xml);
```

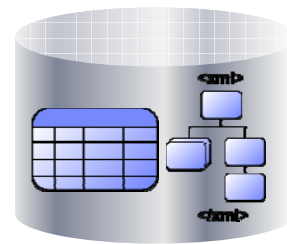
- Relational columns are stored in relational format (tables)
- XML columns are stored natively
- XML stored in UTF8



# XML in DB2 Viper



**SQL Person...** "I see a world class RDBMS that also supports XML"



**DB2 with XML Support**



**XML Person...** "I see a world class XML repository that also supports SQL"

## XML integrated in all facets of DB2!

New XML applications benefit from:

- Ability to seamlessly leverage relational investment
- Proven Infrastructure that provides enterprise-class capabilities

# XQuery: The FLWOR Expression

- **FOR**: iterates through a sequence, bind variable to items
- **LET**: binds a variable to a sequence
- **WHERE**: eliminates items of the iteration
- **ORDER**: reorders items of the iteration
- **RETURN**: constructs query results



```
create table dept(deptID char(8),deptdoc xml);
```

## xquery

```
for $d in db2-fn:xmlcolumn('dept.deptdoc')/dept
let $emp := $d//employee/name
where $d/@bldg > 95
order by $d/@bldg
return
  <EmpList>
    {$d/@bldg, $emp}
  </EmpList>
```

## Input

```
<dept bldg=101>
  <employee id=901>
    <name>John Doe</name>
    <phone>408 555 1212</phone>
    <office>344</office>
  </employee>
  <employee id=902>
    <name>Peter Pan</name>
    <phone>408 555 9918</phone>
    <office>216</office>
  </employee>
</dept>
```



IBM Software Group

# Improved Database Maintenance

*Making it Easier to Maintain the database*



**ON DEMAND BUSINESS**

# Installation Improvements

- Reduce installation complexity
  - ▶ Multiple DB2 versions and fixpacks on the same Windows system
- Multiple instances for maintenance
  - ▶ On Windows, Linux, and UNIX
- Uninstall
  - ▶ Allow full uninstall on Windows

# Backup and Restore

- Table function to list files in a database
  - ▶ Used to automate support of split mirror backup/recovery
- Restartable Recovery
  - ▶ Re-issuing RECOVER command will pick up where it left off
  - ▶ Ability to change Point In Time in either direction
- Rebuild partial database
  - ▶ Eliminate the need for FULL db backup
  - ▶ Ability to rebuild entire DB, including DPF, from set of table space backup images
  - ▶ Scans the history file to build the list of images to restore
- Redirected Restore Script builder
  - ▶ Build a redirected restore script from a backup image

## DB2 Simplified Storage Administration

- User specifies a group of storage devices for DB2, DB2 allocates and grows table consumption of storage on demand.
  - ▶ New to the “Saturn” release of DB2
  - ▶ Intended as a “single point of storage management” for table spaces
  - ▶ Create a database and associate a set of storage paths with it
- AUTOMATIC STORAGE table spaces
  - ▶ No explicit container definitions are provided
  - ▶ Containers automatically created across the storage paths
  - ▶ Growth of existing containers and addition of new ones completely managed by DB2
- Built around DMS storage model
- Add storage paths to the database afterwards
- Redefine those storage paths during a database RESTORE

# Availability Enhancements

- Error Tolerance
  - ▶ Retry read operation
- Error Isolation
  - ▶ Log the specific problems to record the failure and provide appropriate diagnostics.
- Copy, Drop or Rename Schema
  - ▶ Within a database
  - ▶ between databases

## DB2 Thin Client

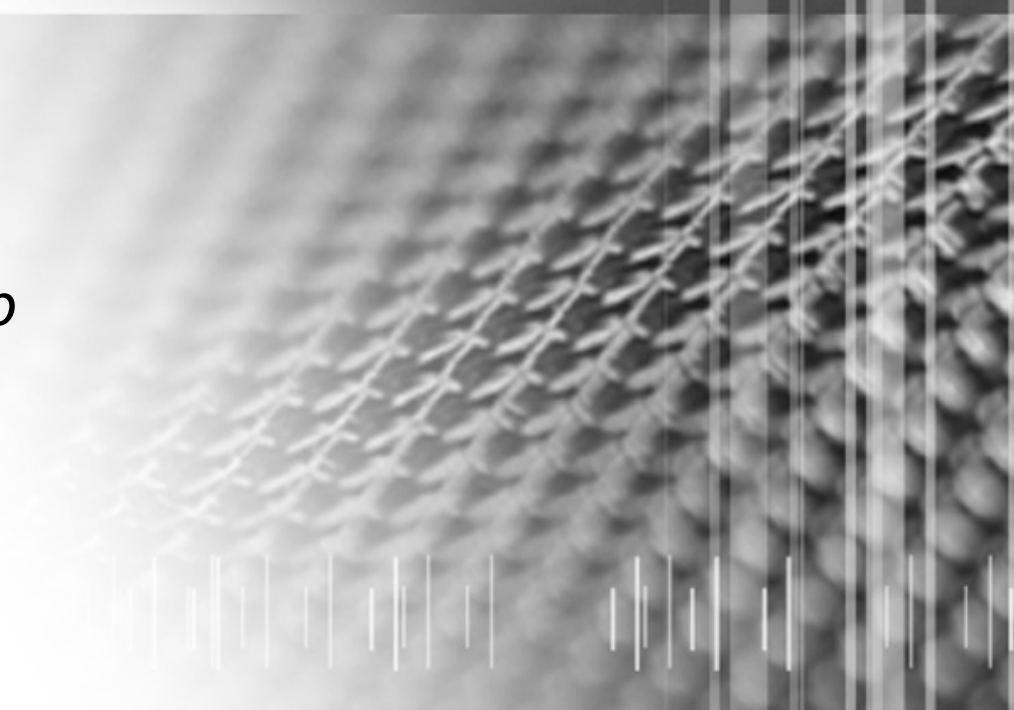
- Provide runtime support for CLI/ODBC
- Eliminate database directories
  - ▶ Connectivity string attributes or db2cli.ini will provide connectivity information
- Allows multiple levels of itself to coexist on a single machine
- Minimal disk and memory footprint
- Installation procedure will simply consist of copying the zip file containing libraries to target directory, unzip, and configure via environment variables and db2cli.ini file
- No dependencies on the Windows registry
- No dependencies on creating a DB2 instance



IBM Software Group

# Autonomic Enhancements

*Reducing the Total Cost of Ownership*



**ON DEMAND BUSINESS**

# Automation Automatically!

- Enable many of the DB2 autonomic computing features by default.
- Examples:
  - ▶ Configuration Advisor (2 second tuning)
  - ▶ Adaptive Self Tuning Memory
  - ▶ Automatic data statistics collection.
- Better defaults for I/O Cleaners and I/O servers
  - ▶ Default for NUM\_IOSERVERS (3) and NUM\_IOCLEANERS (1) set to AUTOMATIC
  - ▶ Values calculated at database startup time  
IOCLEANERS calculated based on number of CPUS and partitions
  - ▶ IOSERVERS calculated based on parallelism settings of all the tablespaces



# Adaptive Self-Tuning Memory

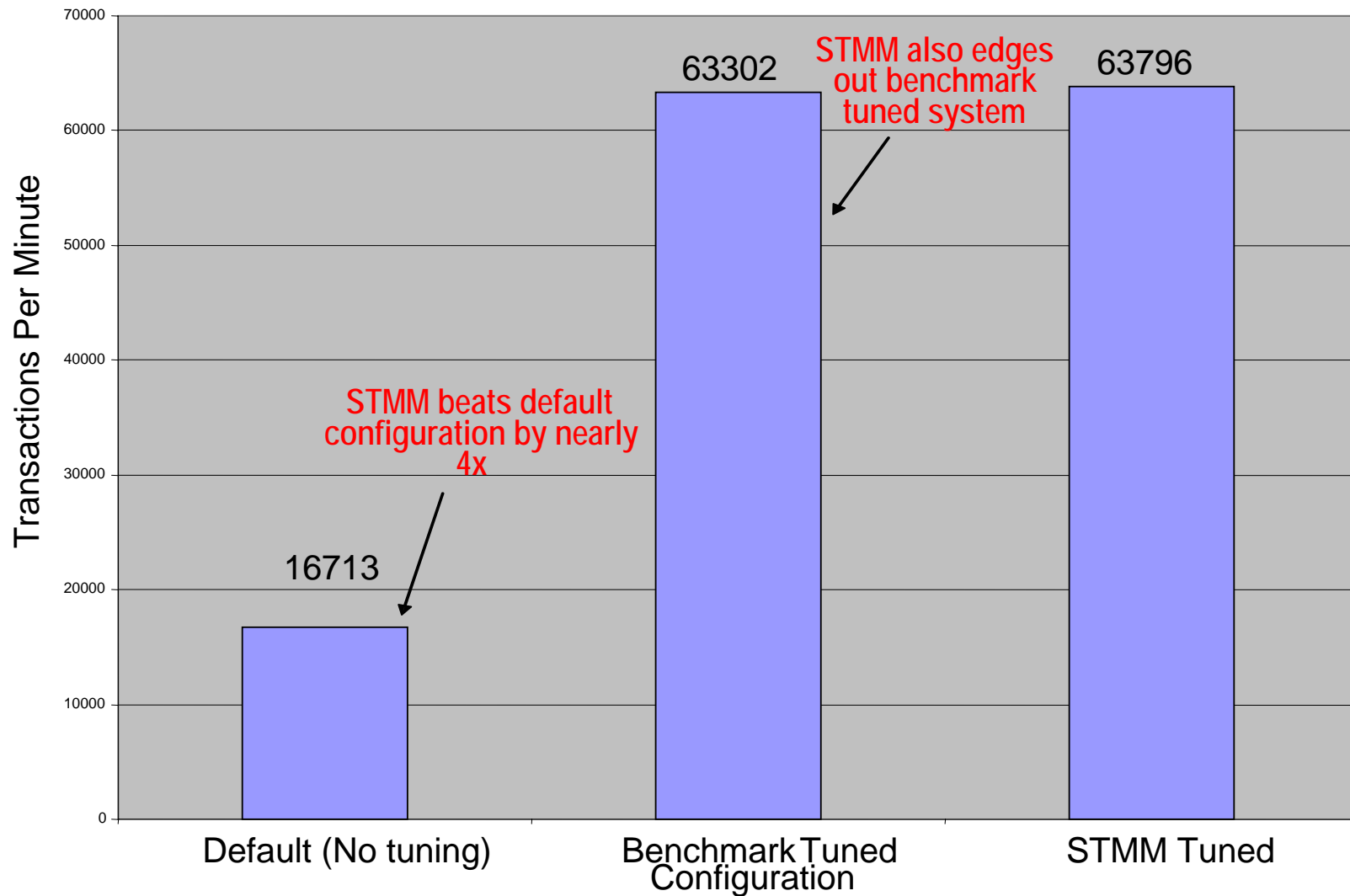
- Viper will introduce a revolutionary memory tuning system called the Self Tuning Memory Manager (STMM)
  - ▶ Works on main database memory parameters
    - Sort, locklist, package cache, buffer pools, and total database memory
  - ▶ Hands-off online memory tuning
    - Requires no DBA intervention
  - ▶ Senses the underlying workload and tunes the memory based on need
  - ▶ Can adapt quickly to workload shifts that require memory redistribution
  - ▶ Adapts tuning frequency based on workload

# STMM in Action – Dropping an Important Index

TPCH Query 21 - After drop index - Average times for the 10 streams



# STMM in Action – Comparing Different Configurations

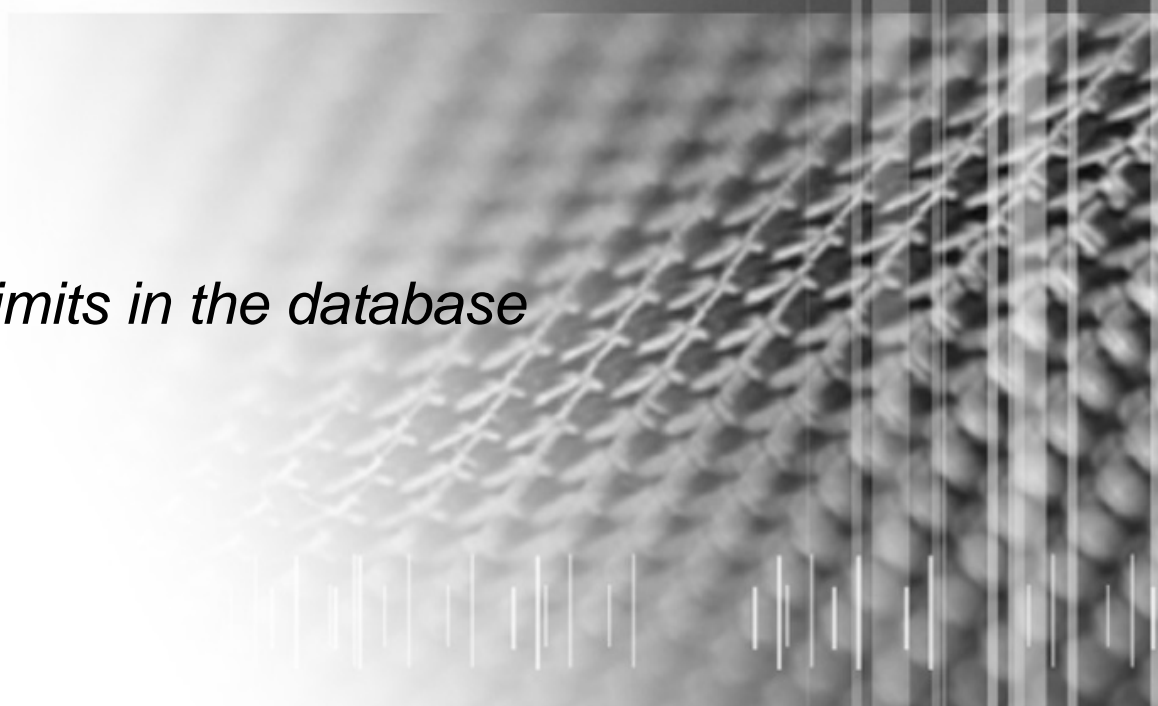




IBM Software Group

# Table Partitioning

*More room for growth and less limits in the database*



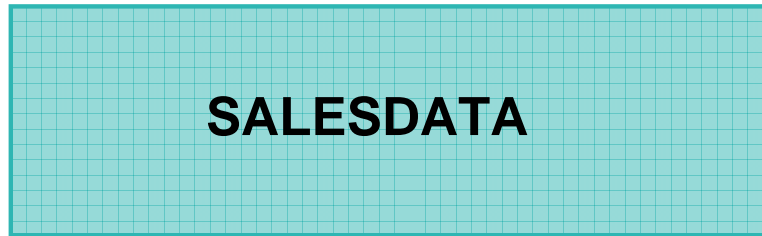
**ON** DEMAND BUSINESS™

# Table Partitioning

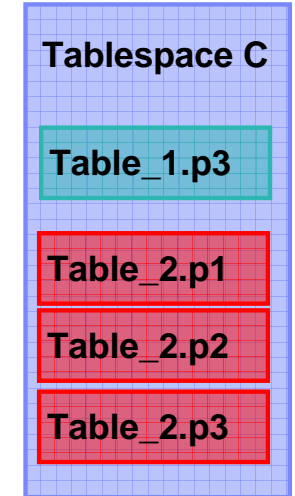
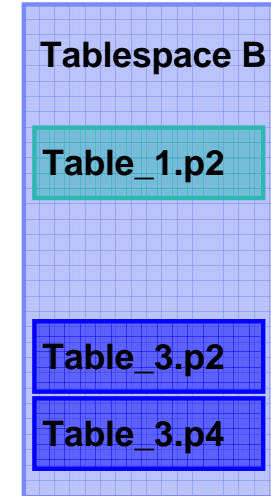
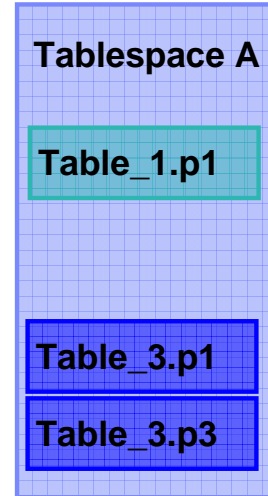
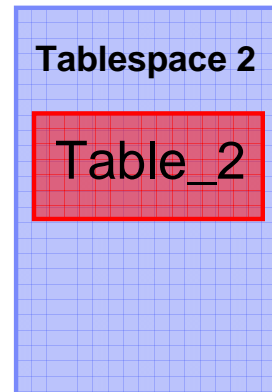
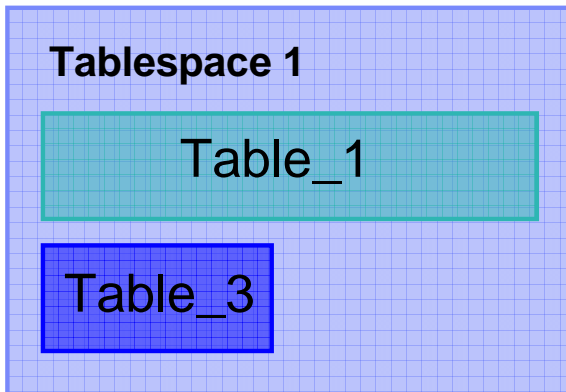
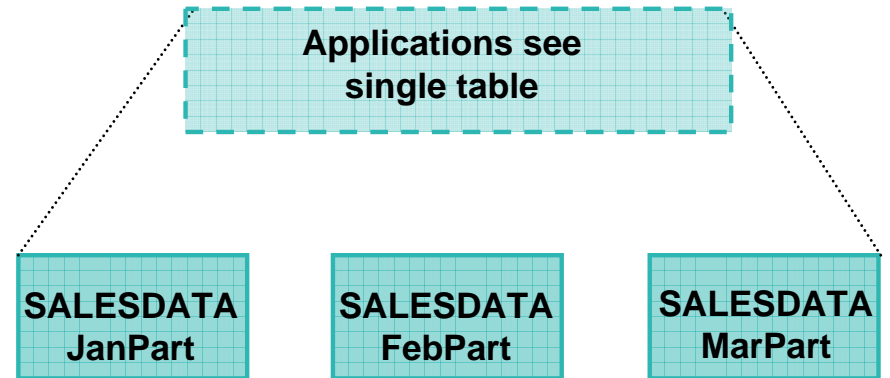
- What is Table (Range) Partitioning ?
  - ▶ Storing a table in more than one physical object, across one or more table spaces
  - ▶ Each table space contains a range of the data that can be found very efficiently
  
- Why?
  - ▶ Increase table capacity limit
  - ▶ Increase large table manageability
  - ▶ Improve SQL performance through partition elimination
  - ▶ Provide fast & online data roll-in and roll-out
  - ▶ Converge towards Informix functionality
  - ▶ Family compatibility with DB2 on zOS and IDS

# Table Partitioning : Benefits

## Without Partitioning

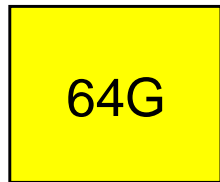


## With Partitioning



# Table Partitioning

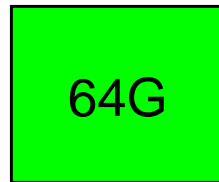
32K Partitions



64G

A-Z

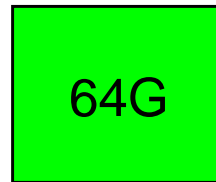
Backup  
Load  
Recover



64G

A-C

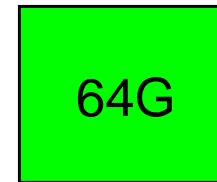
Backup  
Load  
Recover



64G

D-M

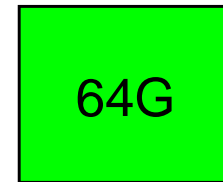
Backup  
Load  
Recover



64G

N-Q

Backup  
Load  
Recover



64G

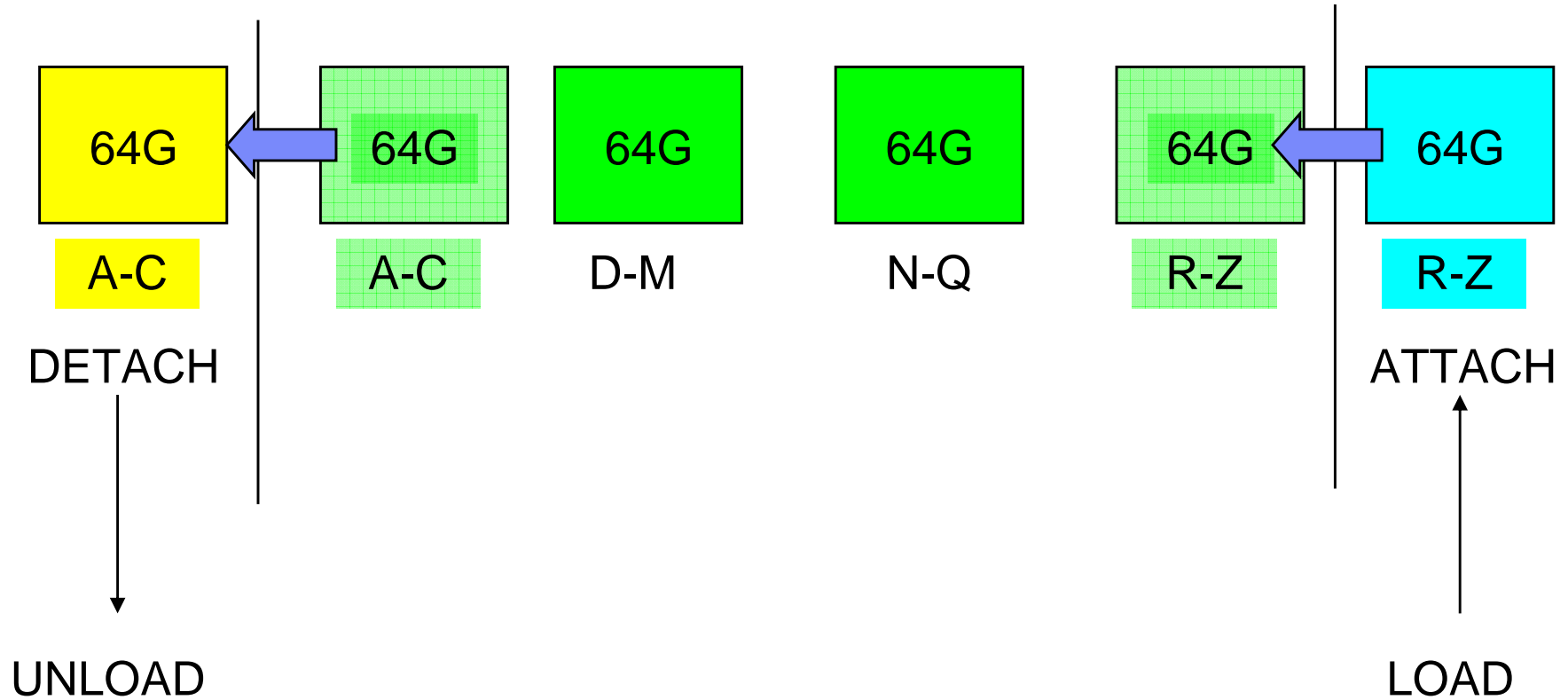
R-Z

Backup  
Load  
Recover



Backup  
Load  
Recover

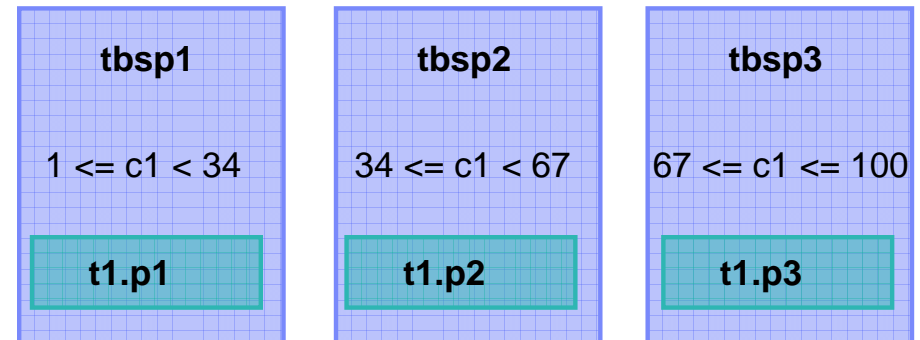
# Table Partitioning / Attach and Detach



# Creating a Range Partitioned Table

- Short and Long Forms
- Partitioning column(s)
  - ▶ Must be base types (eg. No LOBS, LONG VARCHARS)
  - ▶ Can specify multiple columns
  - ▶ Can specify generated columns
- Notes
  - ▶ Special values, MINVALUE, MAXVALUE can be used to specify open ended ranges, eg:
 

```
CREATE TABLE t1 ...
(STARTING(MINVALUE)
ENDING(MAXVALUE) ...
```



## Short Form

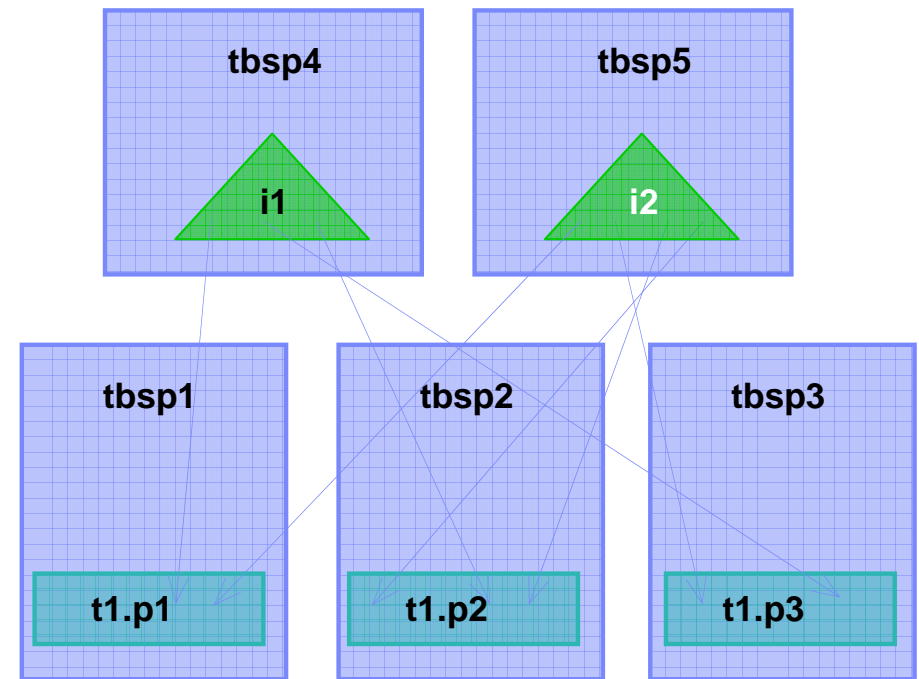
```
CREATE TABLE t1(c1 INT)
  IN tbasp1, tbasp2, tbasp3
  PARTITION BY RANGE(c1)
  (STARTING FROM (1) ENDING( 100) EVERY (33))
```

## Long Form

```
CREATE TABLE t1(c1 INT)
  PARTITION BY RANGE(a)
  (STARTING FROM (1) ENDING(34)IN tbasp1,
  ENDING(67) IN tbasp2,
  ENDING(100) IN tbasp3)
```

# Storage Mapping: Indexes are Global in Viper

- Indexes are **global** (in Viper)
- Each index is in a separate storage object
  - ▶ By default, in the same tablespace as the first data partition
  - ▶ Can be created in different tablespaces, via
    - INDEX IN clause on CREATE TABLE (default is tablespace of first partition)
    - New IN clause on CREATE INDEX
- Recommendation
  - Place indexes in LARGE tablespaces

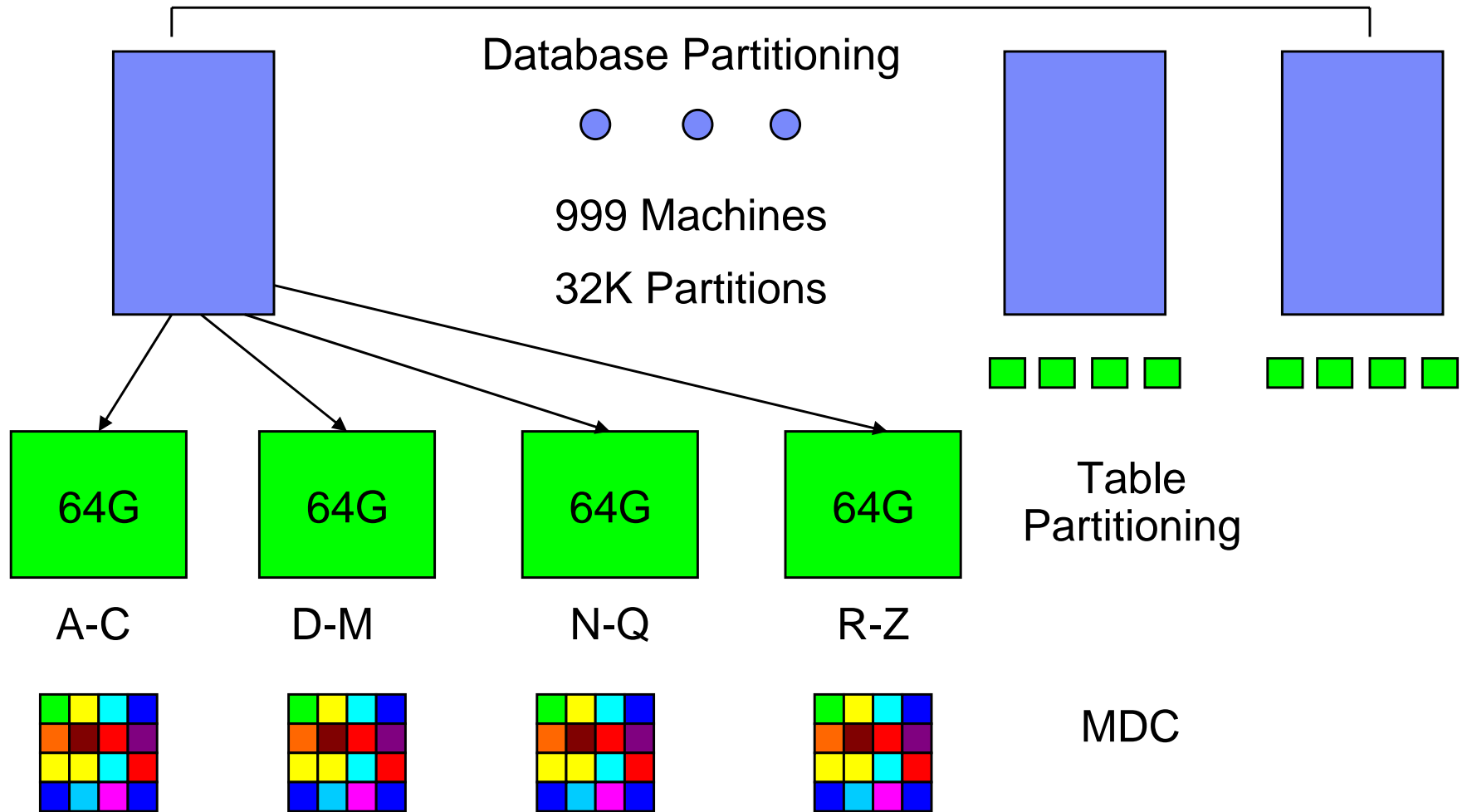


```
CREATE TABLE t1(c1 INT, c2 INT, ...)
  IN tbsp1, tbsp2, tbsp3
  INDEX IN tbsp4
  PARTITION BY RANGE(a)
  (STARTING FROM (1) ENDING (100)
   EVERY (33))
CREATE INDEX i1(c1)
CREATE INDEX i2 (c2) IN tbsp5
```

# New Operations for Roll-Out and Roll-In

- ALTER TABLE ... DETACH
  - ▶ An existing range is split off as a stand alone table
  - ▶ Data instantly becomes invisible
  - ▶ Minimal interruption to other queries accessing table
- ALTER TABLE ... ATTACH
  - ▶ Incorporates an existing table as a new range
  - ▶ Follow with SET INTEGRITY to validate data and maintain indexes
  - ▶ Data becomes visible all at once after COMMIT
  - ▶ Minimal interruption to other queries accessing table
- Key points
  - ▶ No data movement
  - ▶ Nearly instantaneous
  - ▶ SET INTEGRITY is now online

# Hybrid Partitioning





IBM Software Group

# Larger Table space Support

*More room for growth*

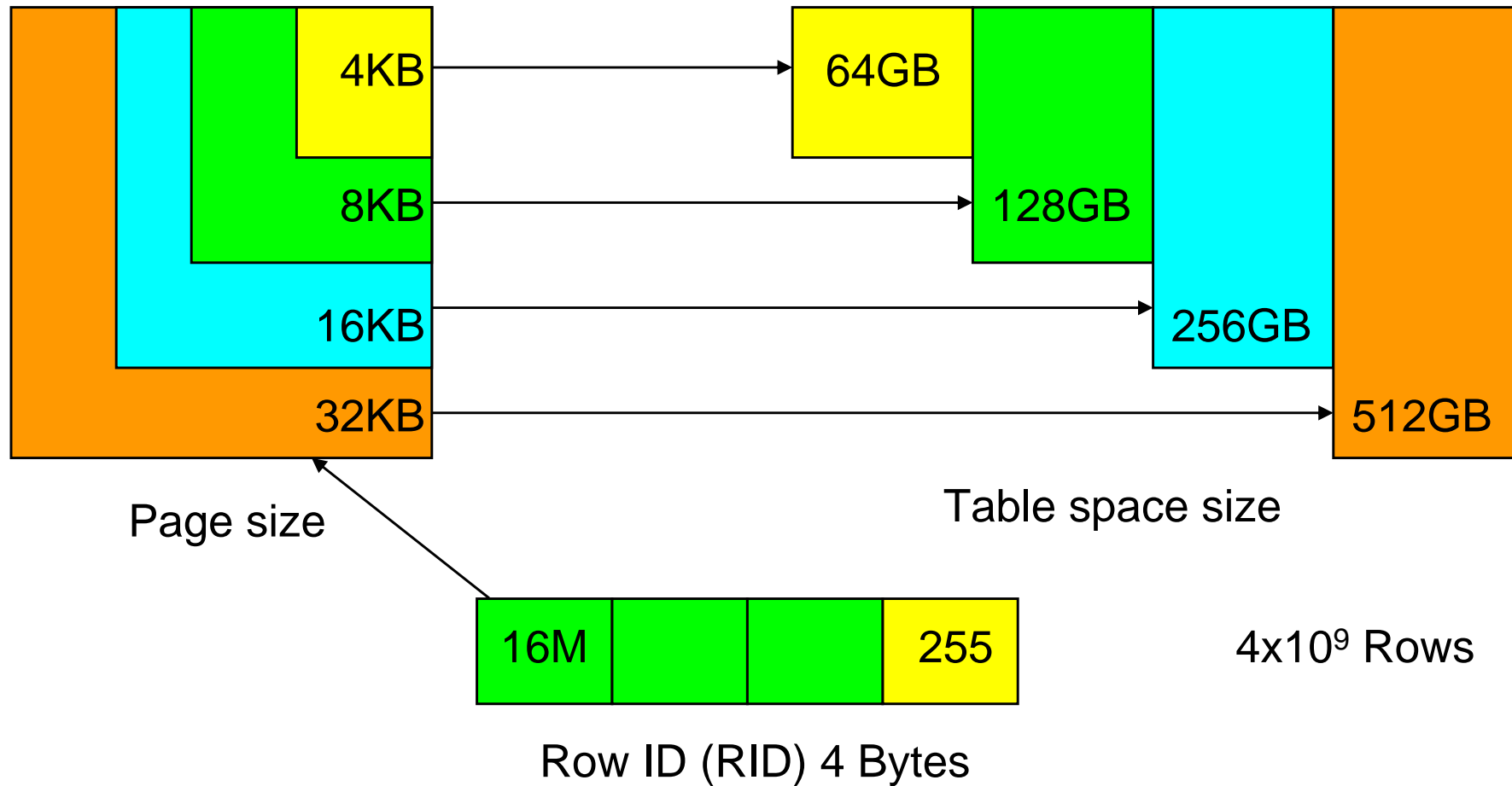


**ON DEMAND BUSINESS**

## Viper Large RIDs – the New Default

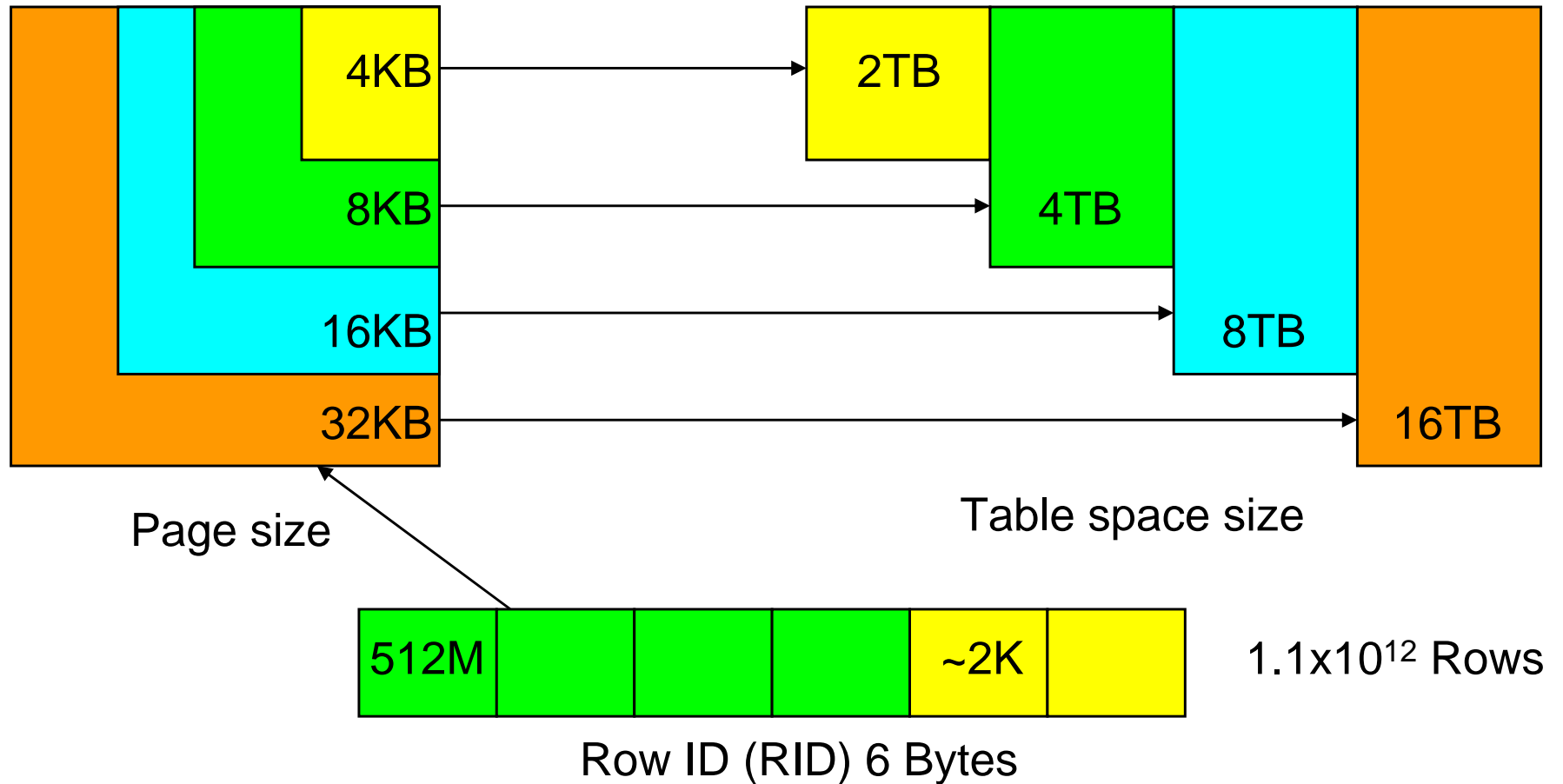
- New 6 byte RID, 4 byte page number and 2 byte slot number
- Infrastructure - runtime, sections, sort, log records, locks – all large RID
- Default table space data type for DMS table spaces is now LARGE
  - ▶ CREATE TABLESPACE <tblspace-name>  
MANAGED BY [DMS | AUTOMATIC STORAGE]
- Tables can now be placed in LARGE table spaces
- Indexes contain regular or large RIDs only, based on the table space type where the table data is stored
  - ▶ Nothing to do with the type of table space where the index resides

# Previous Table Space Design



For tables in all table spaces (regular, temporary, DMS, SMS)

# New Large and Temporary Table Space Design



For tables in LARGE table spaces (DMS only)  
 Also all SYSTEM and USER temporary table spaces

# Altering Existing Table Spaces

- ALTER TABLESPACE <name> CONVERT TO LARGE
  - ▶ Table space is locked, definition is modified and catalogues are updated
  - ▶ Indexes will need to be reorganized
    - Every index for every table in the converted table space needs to be reorganized or rebuilt to convert the RID entries from regular to large
  - ▶ Data will need to be reorganized to allow more rows per page

## Larger Index Size

- Support for larger index key parts and number of columns

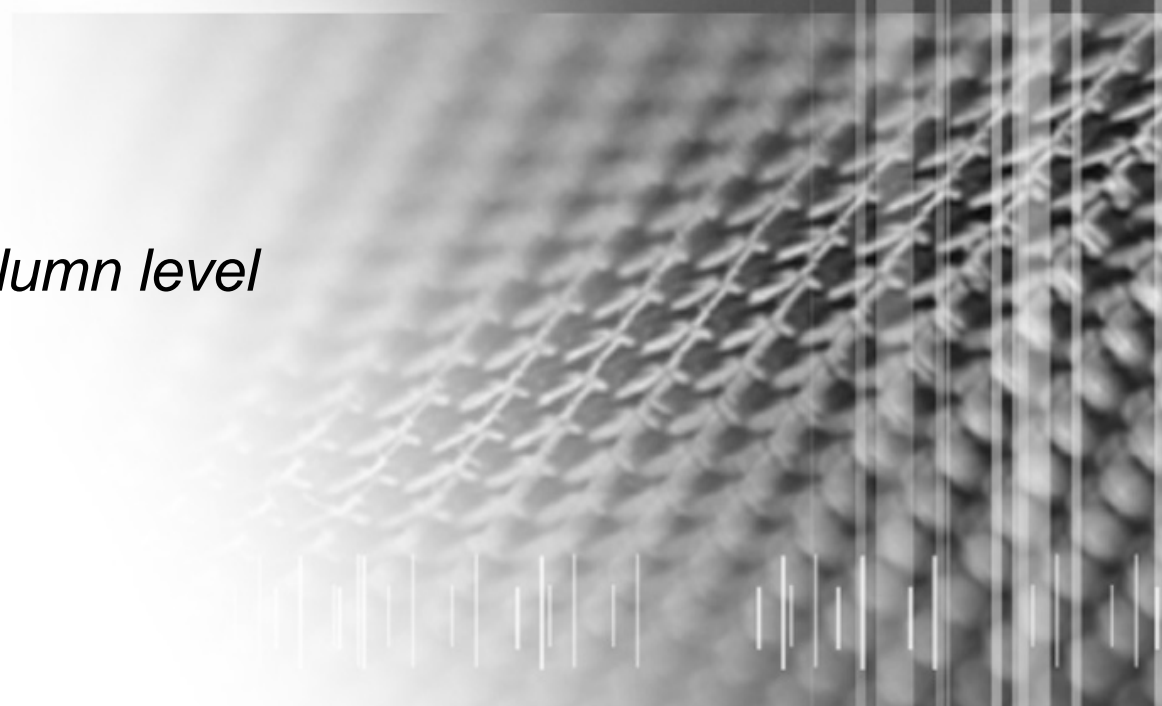
Version	Length of index key parts	# of columns in index key
Pre-Viper	1024	16
Post-Viper	1024 – 4K page	64
	2048 – 8K page	64
	4096 – 16 K page	64
	8192 – 32 k page	64



IBM Software Group

# Granular Security

*Securing tables at the row or column level*

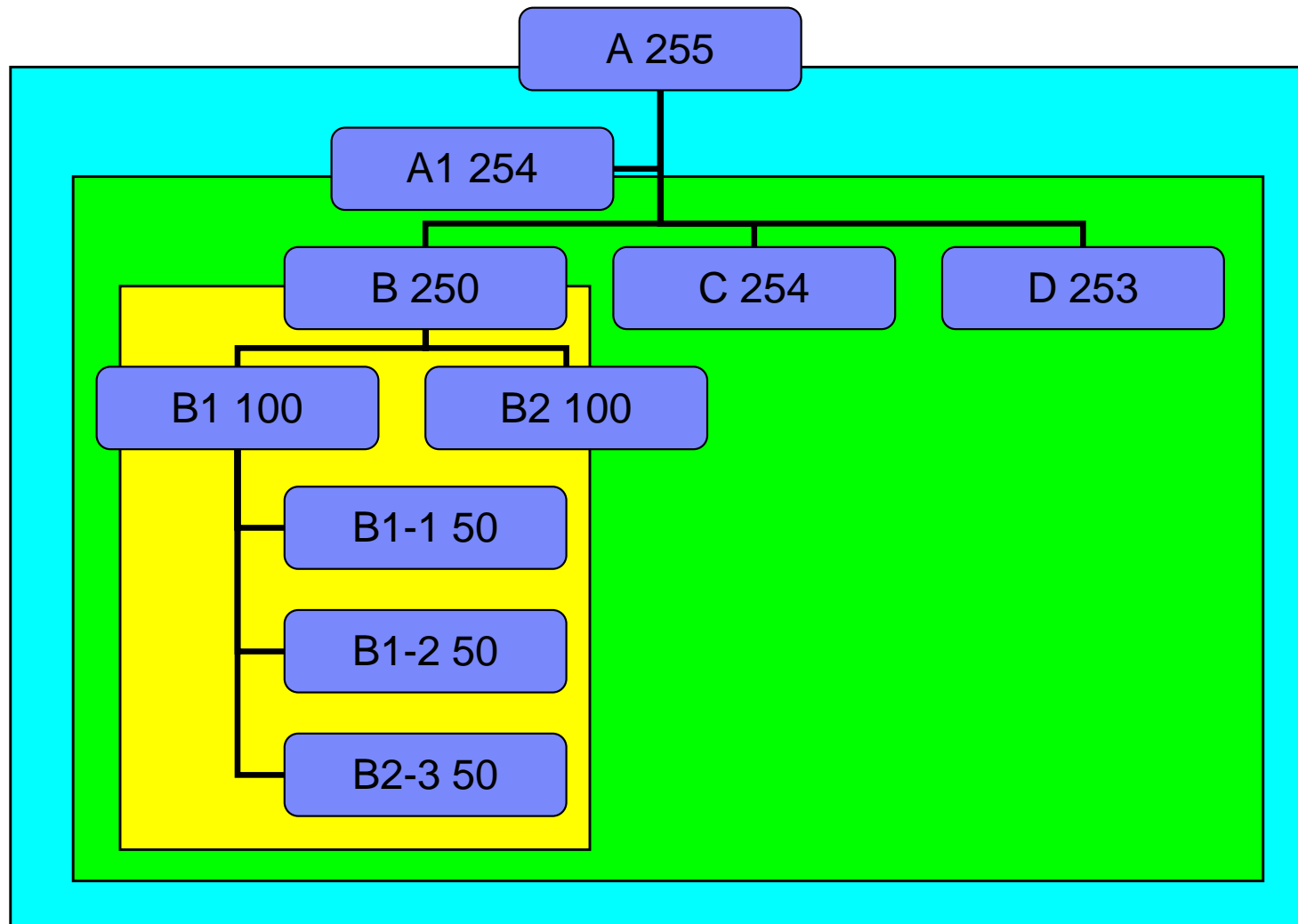


**ON** DEMAND BUSINESS™

# Security - Label Based Access Control

- Label Based Access Control (LBAC)
  - ▶ A “label” is associated with both user sessions and data rows or columns
  - ▶ Rules for comparing users and data labels provide allow access controls to be applied at the row level
- Labels may consist of multiple components
  - ▶ Hierarchical, group or tree types
  - ▶ Row labels appear as a single additional column in a protected table, regardless of the number of label components
  - ▶ User labels are granted by a security administrator
- Similar to the label security support in DB2 for z/OS v8

# LBAC Hierarchy – Tree



# LBAC Query

```
SELECT * FROM EMP
WHERE
    SALARY >= 50000
```

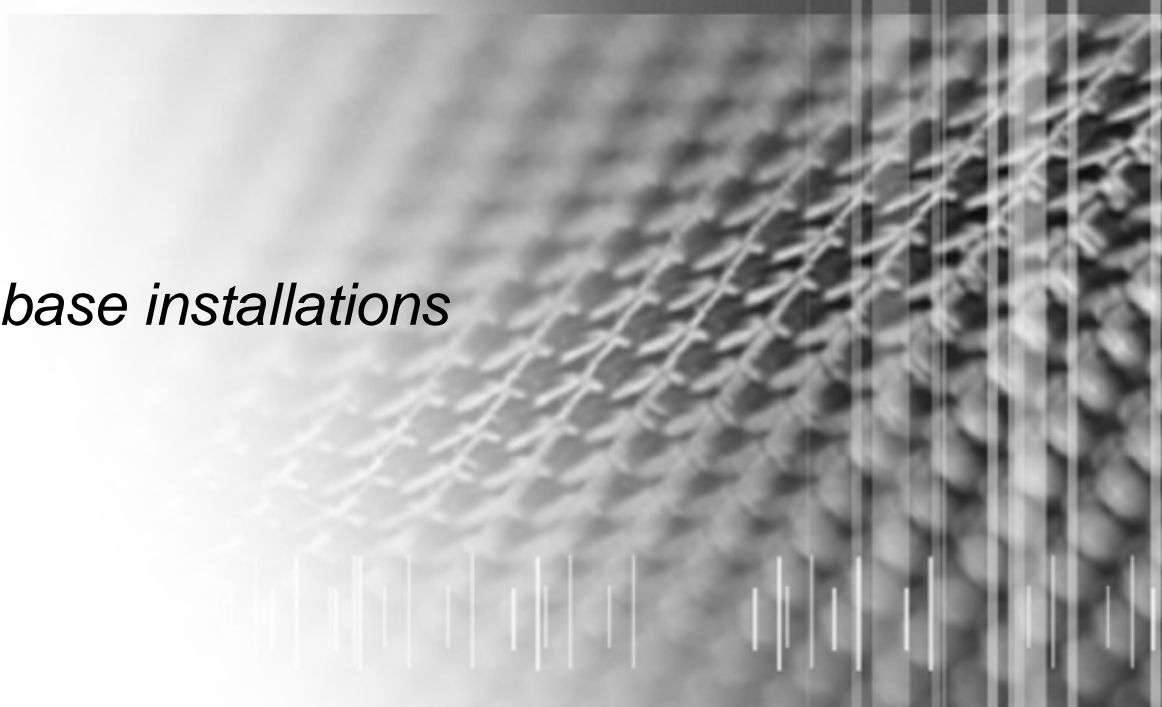
No LBAC	SEC=254	SEC=100	SEC=50	ID	SALARY
				255	60000
				100	50000
				50	70000
				50	45000
				60	30000
				250	56000
				102	82000
				100	54000
				75	33000
				253	46000
				90	83000
				200	78000



IBM Software Group

# Table Compression

*Saving disk space for large database installations*



**ON DEMAND BUSINESS**

# DB2 Compression

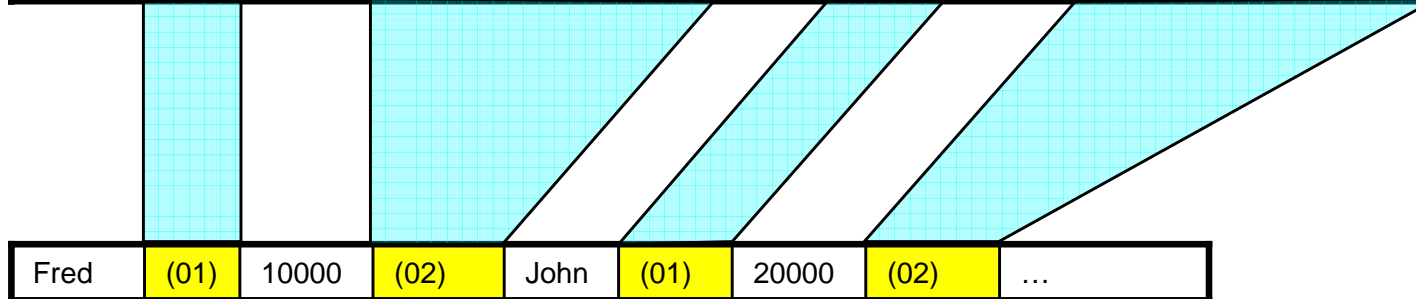
- NULL and Default Value Compression (V8 GA)
  - ▶ No disk storage consumed for NULL column values, zero length data in variable length columns and system default values
- Multidimensional Clustering (V8 GA)
  - ▶ Significant index compression can be achieved through block indexes
    - One key per thousands of records (vs one key per record with traditional indexes)
- Database Backup Compression (V8 FP4)
  - ▶ Smaller backup images; compress index and lf/lob tablespaces
- Data Row Compression (Viper)

# Row Compression Using a Compression Dictionary

- Repeating patterns within the data (and just within each row) is the key to good compression. Text data tends to compress well because of reoccurring strings as well as data with lots of repeating characters, leading or trailing blanks

Name	Dept	Salary	City	State	ZipCode
Fred	500	10000	Plano	TX	24355
John	500	20000	Plano	TX	24355

Fred	500	10000	Plano	TX	24355	John	500	20000	Plano	TX	24355	...
------	-----	-------	-------	----	-------	------	-----	-------	-------	----	-------	-----



Fred	(01)	10000	(02)	John	(01)	20000	(02)	...
------	------	-------	------	------	------	-------	------	-----

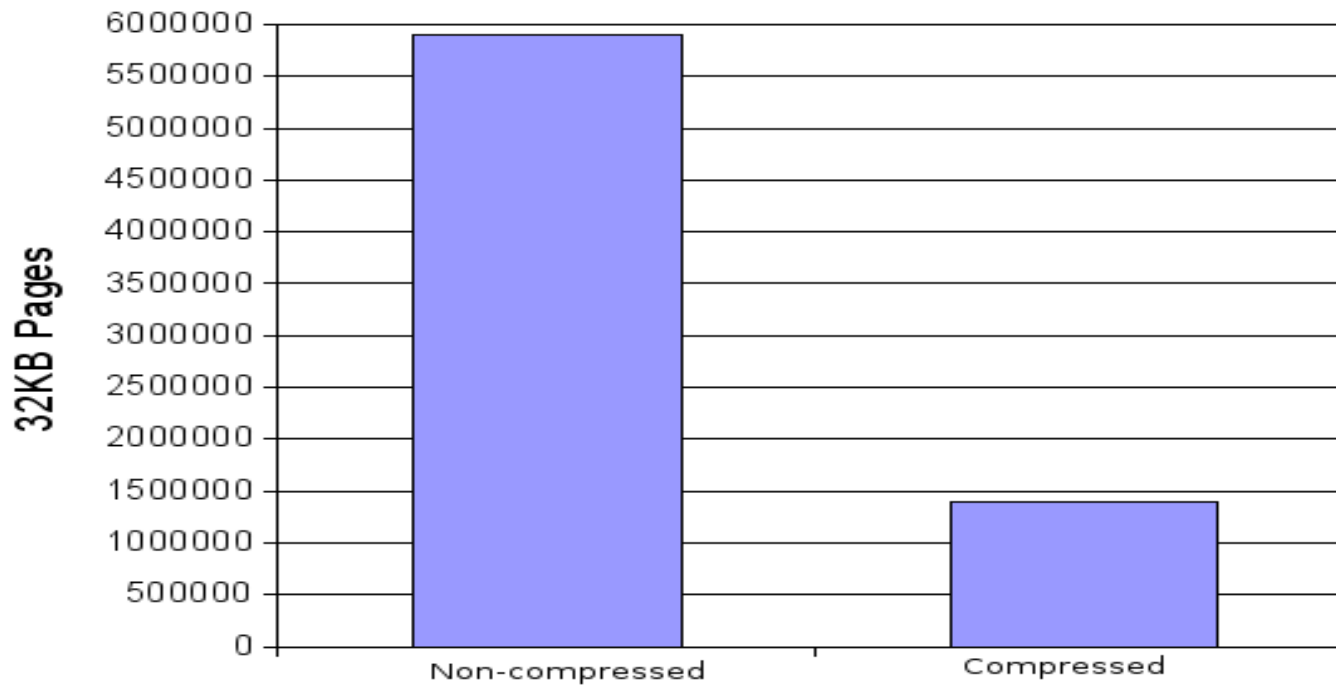
01	Dept 500
02	Plano, TX, 24355
...	...

# More Compression Ratios (Customer Data)

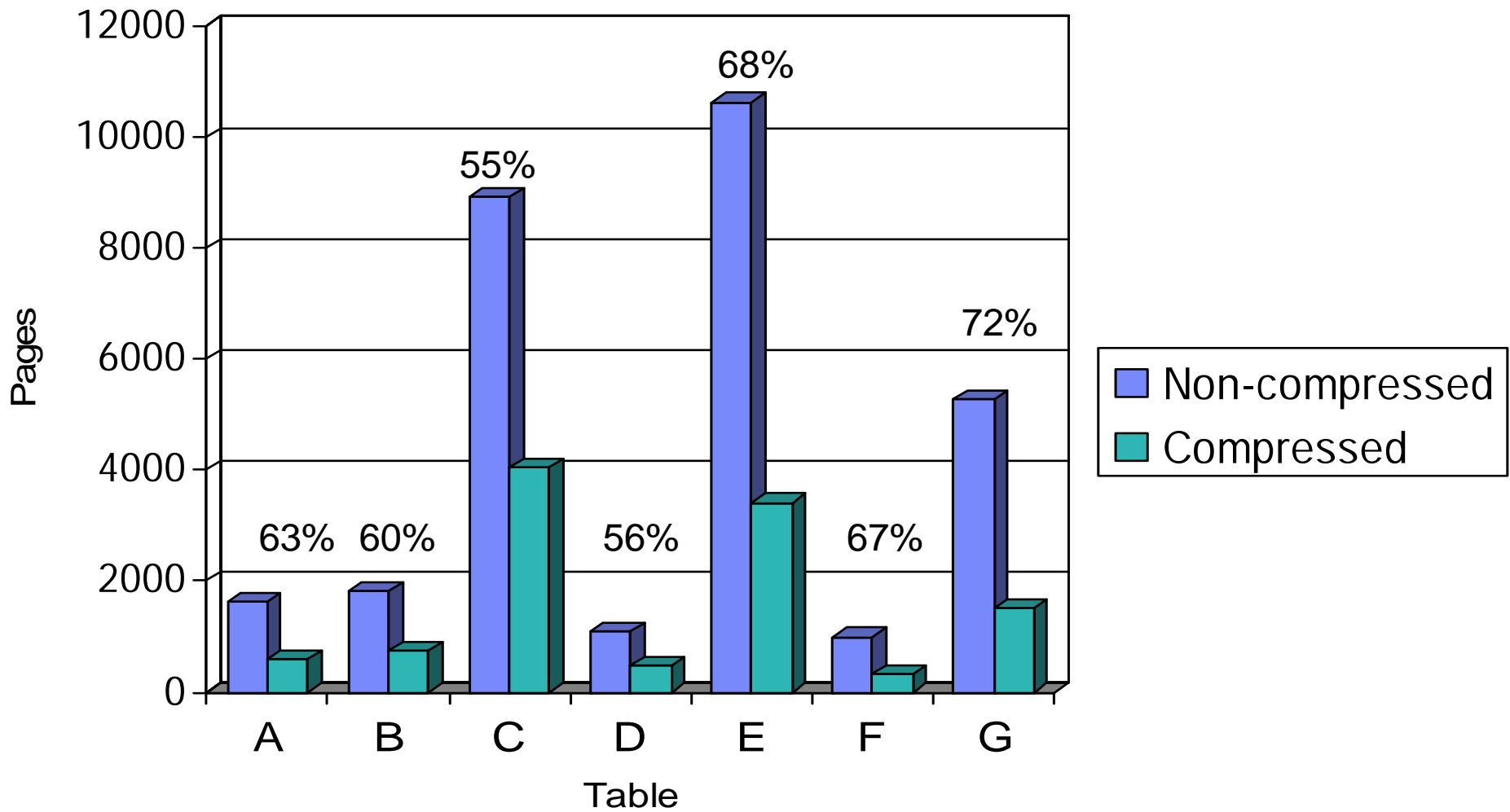
<u>Compression Type</u>	<u>32KB Page Count</u>	<u>Space Required on Disk</u>
No compression	5893888	179.9GB
Row compression	1392446	42.5GB

% Pages Saved: 76.4%

## T1 Compression - 179.9GB Initial Size



# Compression Ratio – Customer Data

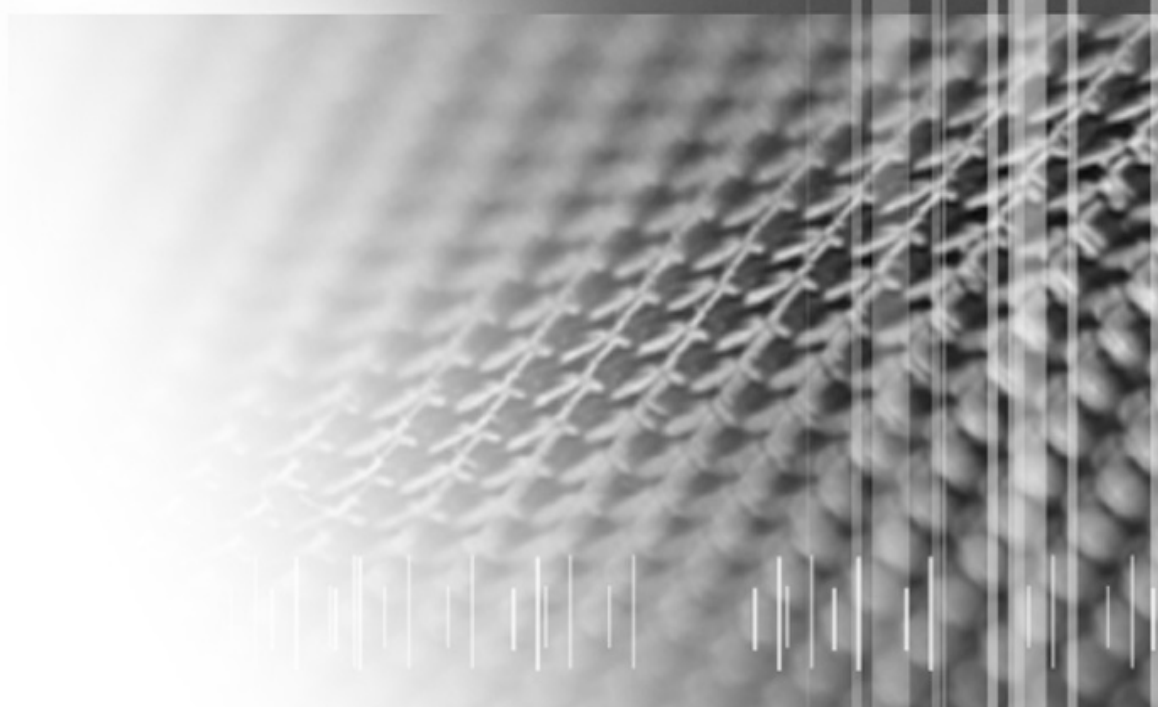




IBM Software Group

## SQL Enhancements

- *Unicode Character Functions*
- *Administrative Functions*
- *CLP Limits*
- *SQL Error Message Function*
- *Alter Table Enhancements*
- *MQT Enhancements*



**ON** DEMAND BUSINESS™

# Unicode Character Functions

- The current SQL functions in DB2 UDB that deal with text strings work on a byte or double-byte basis and ignore character boundaries
  - ▶ A function applied to a character string or graphic string that will treat the character string as an uninterpreted sequence of bytes or byte pairs
- New functions
  - ▶ Find the length of a string
    - CHAR\_LENGTH (Length of a string)
  - ▶ Locate a string in another string
    - LOCATE/POSITION
  - ▶ Find the number of bytes a string takes up
    - OCTET\_LENGTH
  - ▶ Get a substring of a string
    - SUBSTRING
  - ▶ Strip or Trim characters from a String
    - STRIP/TRIM

## Additional Administrative Commands

- CALL ADMIN\_CMD(..) function enhanced to support the following existing commands
  - ▶ IMPORT
  - ▶ UPDATE DBM CFG
  - ▶ UPDATE HISTORY
  - ▶ LOAD
  - ▶ QUIESCE / UNQUIESCE DATABASE
  - ▶ QUIESCE TABLESPACES FOR TABLE
  - ▶ FORCE APPLICATIONS
  - ▶ DESCRIBE TABLE/INDEX/DATA PARTITION
  - ▶ EXPORT (XML support)
  - ▶ REDISTRIBUTE

# Command Line Processor Limits

- Historically, the maximum SQL statement length that could be processed internally has been 64K bytes
  - ▶ SQL Statements in DB2 8.2 are now allowed to exceed 64K bytes in length
  - ▶ SQL statements up to 2M can now be processed internally
- The CLP application has not kept pace and SQL statements passing through the CLP are still limited to a 64K maximum
  - ▶ This feature removes the CLP limitation of 64K SQL statements and allows up to 2M SQL statement lengths

# SQLERRM Function

- There is a heavily used function in PL/SQL (SQLERRM), that returns a varchar message for a specified error code
- Two forms of the scalar function are available
  - ▶ The short form takes the SQLSTATE and returns the short message
  - ▶ The long form takes an locale input and returns the message in the specified locale if the locale is supported at the server
    - Can return a short or long version of the message

# Alter Table Command

- One of the highest administrative costs is making changes to tables such as dropping a column, changing column type, or changing column nullability
  - ▶ To modify these attributes, the user must:
    - unload all table data
    - Drop the table
    - Recreate it along with all its previous authorities, table related objects
    - Reload the table data
- The ALTER command will be modified to allow:
  - ▶ alter table drop column
  - ▶ alter table alter column type
  - ▶ alter table alter column nullability
- When an alter of any of these types is performed, the table is updated to the new design, but only SELECT scans will be allowed (no inserts or updates) until a REORG is completed
  - ▶ No more than 3 separate ALTERs will be allowed before a REORG will be required

# Materialized Query Table Improvements

- Explain Un-used MQTs
  - ▶ Lists MQTs which were not used and the reason for their exclusion
- Mismatched Elements and Expression Support
  - ▶ Allow for expressions with different order of operation to be considered for MQT selection
  - ▶  $C = A + B$  is equivalent to  $C = B + A$
- Maintenance of NULLable MQT columns
- Efficient handling of  $A=B$  OR  $(A \text{ IS NULL AND } B \text{ IS NULL})$  predicates

## Explain Unused MQT's

- Currently, DB2 gives no indication whatsoever if an MQT was even considered during the compilation process
  - ▶ makes it extremely difficult to figure out if a MQT is right or if it is a configuration or costing issue
- Goal is to provide diagnostic messages indicating which MQTs were not used during the query rewrite process
  - ▶ extremely common requirement from customers
- Include high-level reasons why the MQT was disqualified
  - ▶ message provided will be understandable in terms of SQL
- Functionality provided with the following table function:

```
EXPLAIN_GET_MSGS ( )
```



IBM Software Group

# DB2 Viper Technology Preview

*Worldwide Information Management Technical Presales*



**ON DEMAND BUSINESS**