Universal Table Spaces – Matching the Right Type to Our Data

Speaker - Judy Nall
Computer Business International, Inc. (CBI)
DB2 Education and Consulting
IBM Gold Consultant
IBM Data Champion
IBM DB2 DBA, System Administration and Programming Certifications

www.cbi4you.com
866.224.4968 Toll Free
Agenda – Universal Table Spaces

• When do you choose UTS – Range-partitioned vs. UTS Partition by growth?
• Discuss the advantages and disadvantages of each
• What SQL advantages will there be in using UTS’s?
• How to convert to each type?
• How to choose which type to convert to?
• Does using a UTS Partition by growth with a key like ADD-TS to push all new rows to the end hurt when you read?
• Partition-By-Range - Relative Page Numbering (PBR RPN) in Db2 12
Table Space – Which One?

- What type of Tables and table space?
  - Segmented
  - Classic partitioned
  - UTS PBR (Partitioned By Range)
  - UTS PBG (Partitioned By Growth)
  - UTS HASH
  - XML
  - LOB
Well, What Type of Index?

• What type of index?
• NPSI
• DPSI
• PI
• Unique
• Index on expression
Questions?

• How large should the objects be?
  • Should the objects be partitioned?
  • Should the objects be compressed?

• What should the relationship between table space and index attributes be?

• What page size for the table?

• How many indexes if any?
Universal Table Spaces

• Table space that is both segmented and partitioned
  • Bit Maps from Segmented and Partition data

• Two types of universal table spaces are available:
  • The partition-by-growth table space
  • The range-partitioned table space
Universal Table Spaces

• Combination of segmented with partitioning options (hybrid approach)
  • Better space management
  • Support of mass deletes / TRUNCATE
  • One table per table space

• Two options: –

• Range-partitioned (PBR)
  • All the features of classic partitioning • Table controlled partitioning only (no Partitioning Indexes (PI) • Using partition column(s)

• Partition-by-growth (PBG)
  • Partitions added as space is needed • No partitioning key • Partitioned and segmented • DROP / CREATE to migrate existing page sets
  ALTER allowed under some circumstances starting in DB2 V10
Universal Table Space

- The key functions of segmented and partitioned table spaces have been combined in the new universal table space (UTS)

- Reordered Row Format
  - Think about this for a moment (not BRF)
  - DSN1COPY problems we get into

- UTS can be defined with two options:
  - The option to partition by growth (PBG) allows segmented tables to be partitioned as they grow, without needing key ranges, this became the default in DB2 10
  - The option partition by range (PBR) defines the UTS
Catalog Table Information

• SYSIBM.SYSTABLESPACE

• TYPE column
  • G = Partitioned by Growth Universal Table Space
  • O = Table space is a LOB
  • P = Implicit table space created for pureXML columns
  • R = Range Partitioned Universal Table Space

• MAXPARTITIONS
  • Maximum number of partitions
    • Zero if NOT partition by growth

• PARTITIONS
  • The Number of physical partitions (dataset) that currently exist
Universal Table Space Options

• **Partition By Growth – PBG**
  • DSSIZE = Partition Size
  • CREATE TABLESPACE ... (explicit specification)
  • MAXPARTITIONS integer
    • maximum number of partitions in the PBG table space
  • CREATE TABLE ... (implicit specification)
    PARTITIONED BY SIZE EVERY integer G

• **Partition By Range - PBR**
  • CREATE TABLESPACE ... SEGSIZE integer NUMPARTS integer
    • SEGSIZE
Universal Table Spaces

• NUMPARTS and SEGSIZE are specified, the table space that is created is a partition-by-range (UTS) table space.

• If MAXPARTITIONS or MAXPARTITIONS and SEGSIZE are specified, the table space that is created is a partition-by-growth universal table space.
Partition by Growth

- **UTS**
  - Partitioned-by-growth UTS have better space management and improved delete performance than traditional partitioned table spaces due to their segmented space organization.
  - Managed by DB2 depending on the DSSIZE and MAXPARTITIONS chosen.
  - Space must be STOGROUP defined, and only non-partitioning indexes (NPIs) are allowed to be defined on PBG UTS.
CREATE TABLESPACE statement for PBG

CREATE TABLESPACE TS1 IN DB1
MAXPARTITIONS 55
SEGSIZE 64
DSSIZE 8G
LOCKSIZE ANY;

• MAXPARTITIONS –
  • Specifies the maximum # of partition for a table space
  • MAXPARTITIONS can be changed by ALTER TABLESPACE
  • Keep in mind that ALTER MAXPARTITIONS may require down time because it needs to physically close the datasets
Table Size Limits

• DB2 11, a single table is limited to 16TB
• RID changes in DB2 12 have allowed IBM to increase this limit to 4PB
• Customers at or approaching the current size limit,
  • Enhancement can avoid the need to implement costly and intrusive solutions such as splitting large tables into multiple smaller ones

• How large should the objects be?
  • What type of Table Space should I choose?
Partition By Growth

- The Table Space starts with one partition
- Additional partitions will be added on demand until the maximum partition is reached

MAX PART n
ALTER TABLESPACE

• The following ALTER TABLESPACE options can cause pending changes to the definition of the table space under certain conditions:
  • BUFFERPOOL
  • DSSIZE
  • MAXPARTITIONS
  • MEMBER CLUSTER
  • SEGSIZE
Reducing the Size of PBG

PROBLEM DESCRIPTION:
• Users experience storage management issues due to partition-by-growth (PBG) table spaces defined with large MAXPARTITIONS value

• Users may experience difficulty with storage management because internal control blocks were taking up storage for PBG table spaces defined with large MAXPARTITIONS value

• Plans bound with the ACQUIRE ALLOCATE option that refer to the table space will be invalidated when the MAXPARTITIONS value is altered – see APAR PM57001

• You can increase the value of MAXPARTITIONS keyword with the ALTER SQL statement but you CANNOT reduce the value of the MAXPARTITIONS keyword

• The warning will be that if you pick very high values for MAXPARTITIONS, something like 4096, you cannot change it without a drop/create table space
Partition By Growth

• DB2 12 will now allow ALTER TABLESPACE MAXPARTITIONS to a smaller value
  • Previously, SQLCODE -644 was issued to restrict this case altogether
• Now DB2 allows the ALTER, but will still issue SQLCODE -644 if the MAXPARTITIONS value is altered to a number lower than the physical partitions allocated
• Dependent plans defined with the ACQUIRE ALLOCATE option will be invalidated on ALTER TABLESPACE MAXPARTITIONS
• DB2 allocates the storage necessary per thread for the partition-by-growth table space based on the value set in MAXPARTITIONS.
  • Using a value of 4096 caused a good chunk of storage to get initially allocated for each thread
Partition by Range

• UTS have better space management and improved delete performance due to their segmented space organization

• PBR RPN –Db2 12
  • This new type of table space is called Partition-By-Range - Relative Page Numbering
  • **PBR RPN** table spaces can have a maximum partition size of 1 TB
  • New 7 byte RIDs

• New system parameter called PAGESET_PAGENUM is also introduced by this enhancement.
  • It controls whether page numbering of newly created PBR table spaces is relative or absolute by default
Terms

• Data Partitioned Secondary Indexes
  • DPSI’s

• Non-Partitioned Secondary Indexes
  • NPSI’s
Terminology

• Partitioned Table Space
  • Any TS with multiple physical partitions

• Partitioned Index
  • Any index that has multiple physical partitions

• Partitioning Index
  • Left most columns match the partitioning key of the TS
  • The index might, might not be partitioned
Partitioned TS

- Separating the clustering from the partitioning
- Need not be clustered by the partitioning limit key
- We have Table partitioning vs. Index Partitioning
- We will look at both
Table Spaces

- Maximum number of partitions raised in V11
  - From 254 to 4096
  - Table space must have LARGE or DSSIZE to go beyond 254 parts
  - Max table size determined based on page size, DSSIZE and number of parts
DSSIZE

- Value in gigabytes that indicates the maximum size for each partition
  - If you specify DSSIZE you must also specify NUMPARTS or LOB
- IF DSSIZE or LARGE is omitted then
  - MAX SIZE depends on value of NUMPARTS
• **DSSIZE**
  - Recommended using over LARGE parameter
  - Controls whether ‘LARGE’ or ‘NOT LARGE’
  - Specify the maximum size for each partition, LOB or data set
  - If you use, you must specify
    - NUMPARTS or LOB
  - Specify value > 4GB, TS must be EA-enabled
LARGE - V11

• LARGE
  • Use DSSIZE instead
  • Only for partitioned table spaces
  • When specified more than 64GB of data can be stored in a table space

• RIDs are 5 bytes instead of 4 bytes in TS Increases space of Index as well

• Can have a TS of
  • 128 Terabytes with a page size of 32K
# Db2 12 DSSIZE and NUMPARTS

<table>
<thead>
<tr>
<th>DSSIZE value</th>
<th>4K page size</th>
<th>8K page size</th>
<th>16K page size</th>
<th>32K page size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1G - 4G (1 GB to 4 GB)</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
</tr>
<tr>
<td>8G (8 GB)</td>
<td>2048</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
</tr>
<tr>
<td>16G (16 GB)</td>
<td>1024</td>
<td>2048</td>
<td>4096</td>
<td>4096</td>
</tr>
<tr>
<td>32G (32 GB)</td>
<td>512</td>
<td>1024</td>
<td>2048</td>
<td>4096</td>
</tr>
<tr>
<td>64G (64 GB)</td>
<td>254</td>
<td>512</td>
<td>1024</td>
<td>2048</td>
</tr>
<tr>
<td>128G (128 GB)</td>
<td>128</td>
<td>256</td>
<td>512</td>
<td>1024</td>
</tr>
<tr>
<td>256G (256 GB)</td>
<td>64</td>
<td>128</td>
<td>256</td>
<td>512</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value of NUMPARTS</th>
<th>Maximum partition size (default for DSSIZE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 to 16</td>
<td>4GB (4G)</td>
</tr>
<tr>
<td>17 to 32</td>
<td>2GB (2G)</td>
</tr>
<tr>
<td>33 to 64</td>
<td>1GB (1G)</td>
</tr>
<tr>
<td>65 to 254</td>
<td>4GB (4G)</td>
</tr>
</tbody>
</table>
Db2 12 - Maximum number of partitions table space with absolute numbering

<table>
<thead>
<tr>
<th>Type of RID</th>
<th>Page size</th>
<th>DSSIZE</th>
<th>Maximum number of partitions</th>
<th>Total table space size</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-byte EA</td>
<td>4KB</td>
<td>1GB</td>
<td>4096</td>
<td>4TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>4KB</td>
<td>2GB</td>
<td>4096</td>
<td>8TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>4KB</td>
<td>4GB</td>
<td>4096</td>
<td>16TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>4KB</td>
<td>8GB</td>
<td>2048</td>
<td>16TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>4KB</td>
<td>16GB</td>
<td>1024</td>
<td>16TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>4KB</td>
<td>32GB</td>
<td>512</td>
<td>16TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>4KB</td>
<td>64GB</td>
<td>256</td>
<td>16TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>4KB</td>
<td>128GB</td>
<td>128</td>
<td>16TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>4KB</td>
<td>256GB</td>
<td>64</td>
<td>16TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>8KB</td>
<td>1GB</td>
<td>4096</td>
<td>4TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>8KB</td>
<td>2GB</td>
<td>4096</td>
<td>8TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>8KB</td>
<td>4GB</td>
<td>4096</td>
<td>16TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>8KB</td>
<td>8GB</td>
<td>4096</td>
<td>32TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>8KB</td>
<td>16GB</td>
<td>2048</td>
<td>32TB</td>
</tr>
<tr>
<td>8-byte EA</td>
<td>8KB</td>
<td>32GB</td>
<td>1024</td>
<td>32TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>8KB</td>
<td>64GB</td>
<td>512</td>
<td>32TB</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>------</td>
<td>-----</td>
<td>------</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>8KB</td>
<td>128GB</td>
<td>256</td>
<td>32TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>8KB</td>
<td>256GB</td>
<td>128</td>
<td>32TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>16KB</td>
<td>1GB</td>
<td>4096</td>
<td>4TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>16KB</td>
<td>2GB</td>
<td>4096</td>
<td>8TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>16KB</td>
<td>4GB</td>
<td>4096</td>
<td>16TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>16KB</td>
<td>8GB</td>
<td>4096</td>
<td>32TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>16KB</td>
<td>16GB</td>
<td>4096</td>
<td>64TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>16KB</td>
<td>32GB</td>
<td>2048</td>
<td>64TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>16KB</td>
<td>64GB</td>
<td>1024</td>
<td>64TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>16KB</td>
<td>128GB</td>
<td>512</td>
<td>64TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>16KB</td>
<td>256GB</td>
<td>256</td>
<td>64TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>32KB</td>
<td>1GB</td>
<td>4096</td>
<td>4TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>32KB</td>
<td>2GB</td>
<td>4096</td>
<td>8TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>32KB</td>
<td>4GB</td>
<td>4096</td>
<td>16TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>32KB</td>
<td>8GB</td>
<td>4096</td>
<td>32TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>32KB</td>
<td>16GB</td>
<td>4096</td>
<td>64TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>32KB</td>
<td>32GB</td>
<td>4096</td>
<td>128TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>32KB</td>
<td>64GB</td>
<td>2048</td>
<td>128TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>32KB</td>
<td>128GB</td>
<td>1024</td>
<td>128TB</td>
</tr>
<tr>
<td>5-byte EA</td>
<td>32KB</td>
<td>256GB</td>
<td>512</td>
<td>128TB</td>
</tr>
<tr>
<td>5-byte (non-EA) LARGE</td>
<td>4KB</td>
<td>(4GB)</td>
<td>4096</td>
<td>16TB</td>
</tr>
</tbody>
</table>
Maximum Partition Size V11

<table>
<thead>
<tr>
<th>DSSIZE value</th>
<th>4K page size</th>
<th>8K page size</th>
<th>16K page size</th>
<th>32K page size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1G - 4G (1 GB to 4 GB)</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
</tr>
<tr>
<td>8G (8 GB)</td>
<td>2048</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
</tr>
<tr>
<td>16G (16 GB)</td>
<td>1024</td>
<td>2048</td>
<td>4096</td>
<td>4096</td>
</tr>
<tr>
<td>32G (32 GB)</td>
<td>512</td>
<td>1024</td>
<td>2048</td>
<td>4096</td>
</tr>
<tr>
<td>64G (64 GB)</td>
<td>254</td>
<td>512</td>
<td>1024</td>
<td>2048</td>
</tr>
<tr>
<td>128G (128 GB)</td>
<td>128</td>
<td>256</td>
<td>512</td>
<td>1024</td>
</tr>
<tr>
<td>256G (256 GB)</td>
<td>64</td>
<td>128</td>
<td>256</td>
<td>512</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value of NUMPARTS</th>
<th>Maximum partition size (default for DSSIZE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 to 16</td>
<td>4GB (4G)</td>
</tr>
<tr>
<td>17 to 32</td>
<td>2GB (2G)</td>
</tr>
<tr>
<td>33 to 64</td>
<td>1GB (1G)</td>
</tr>
<tr>
<td>65 to 254</td>
<td>4GB (4G)</td>
</tr>
</tbody>
</table>

NUMPARTS integer must be a value between 1 and 4096 inclusive and must be less than or equal to the value that is specified for the MAXPARTITIONS clause.
Maximum partition size depending on page size - V11

<table>
<thead>
<tr>
<th>Page size</th>
<th>Maximum partition size (default for DSSIZE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4K</td>
<td>4GB (4G)</td>
</tr>
<tr>
<td>8K</td>
<td>8GB (8G)</td>
</tr>
<tr>
<td>16K</td>
<td>16GB (16G)</td>
</tr>
<tr>
<td>32K</td>
<td>32GB (32G)</td>
</tr>
</tbody>
</table>
Data Set Naming

• VCAT.DSNDBC.DBNAME.TSNAME.I0001.A001 (A001 - A999)

• Dataset naming convention
  • ‘Axxx’ – partitions 1- 999
  • ‘Bxxx’ - partitions 1000 – 1999
  • ‘Cxxx’ - partitions 2000 – 2999
  • ‘Dxxx’ - partitions 3000 – 3999
  • ‘Exxx’ – partitions 4000 – 4096
The Real Size of a Table Space?

• Row Length and Number of Rows
  • For a 4K page size
  • EX: 400 bytes * 2M rows / bytes in page size - 22
  • 197,238.65 pages just to load the data
    • Number of rows + the number of rows(100*PCTFREE)
    • Add the number of free pages
    • Add the Header, Bit Map, System page
  • Now get this into PRIQTY value with growth

• Growth - Need to know
  • Insert/Delete activity, how it grows over time
  • Get this to a PCTFREE value to add back
  • You will also have pages for Header, several Bit Maps, System page, and if you use compression, 16 pages max.
Running Out of Space

• Table space fills, no space available

• Why?
  • Page is Hot
  • Really is Full!

• Secondary Extent is Built for Data
  • These keep being allocated at the same size
How to Make TS Larger?

• PRIQTY
  • ALTER command to increase
  • Use ADM Tool

• Do you have the disk space?

• What about PCTFREE and FREEPAGE?

• Make the change
  • Does not go into effect immediately
  • REORG uses the new parameter
Standards?

• Do you have a defaults, for
  • PRIQTY or SECQTY
  • Do you estimate growth rates?
    • Your Small, Medium, Large tables?
    • For FREESPACE, PCTFREE?
Extent Sizes

• ZParm’s parameter MGEXTSZ
  • For Storage Group Managed TS’s
  • Default is NO
  • Enable Sliding Secondary Quantity

• TSQTY / IXQTY
  • Specify default primary allocation for TS / IX
Sliding SECQTY

DSNZPAPRMS

• TSQTY
  • TSQTY specifies the number of space in KB for the primary space allocation quantity for DB2-managed table spaces that are created without the USING clause.
  • A value of 0 (zero) indicates that you want to use standard defaults (now 1 cylinder)

• IXQTY
  • IXQTY fulfils a similar role but for index spaces.
Secondary Sliding Scale

• Sliding Scale
  • A sliding scale means that the first secondary extent allocations are smaller than later secondary allocations

• This feature delivers autonomic selection of data set extent sizes with a goal of preventing extent errors before reaching maximum data set size
Secondary Extents

• OPTIMIZE EXTENT SIZING
  • specifies whether secondary extent allocations for DB2-managed data sets are to be sized according to a sliding scale that optimizes the likelihood of reaching the maximum data set size before secondary extents are exhausted.
  • If you select NO, the default value, you will manage secondary extent allocations manually
  • YES, DB2 will automatically optimize the secondary extent allocations
PBG – When and Limitations

When?

- No obvious partitioning column exists
- Space on Demand
- Table requiring > 64G

Limits?

- No shrinking of partitions
  - Even if there are only empty partitions at the end of the table space
  - These could have header page, space map pages, dictionary page and system pages
COPY

• Copies made at the part level or the table space level
• Empty partitions will be copied
PBG - DSN1COPY

• Partition number may be inconsistent between
  • DSN1COPY and Target table space

• Partition number of Target Table Space > Number of Source table space
• We use TRUNCATE TABLE on the target before DSN1COPY – Now Empty

• Partition number of Target < partition number of Source table space
• DSN1COPY cannot be used!
• UNLOAD/LOAD can be used
Well, Are We There Yet?

• When do you choose UTS – Range-partitioned vs. UTS Partition by growth?
• Discuss the advantages and disadvantages of each
• What SQL advantages will there be in using UTS’s?
• How to convert to each type?
• How to choose which type to convert to?
• Does using a UTS Partition by growth with a key like ADD-TS to push all new rows to the end hurt when you read?
• Partition-By-Range - Relative Page Numbering (PBR RPN) in Db2 12
Thanks!

Hope you enjoyed the presentation !!!!
• Let me know if you have additional questions
  • www.cbi4you.com
  • judynall@cbi4you.com

• Check out our schedule of classes at CBI
• We have ‘real-time’ instructor led virtual classes with hands-on labs
• Consulting services on Db2