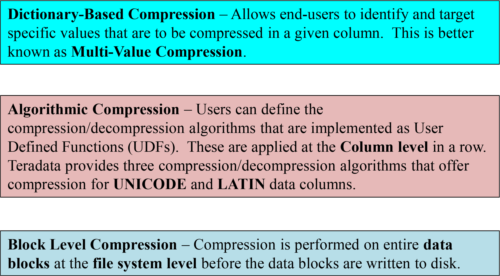
Chapter 16 - Compression

“If you aren’t compressing your Teradata tables using Multi-Value Compression (MVC), you are simply leaving money on the table.”

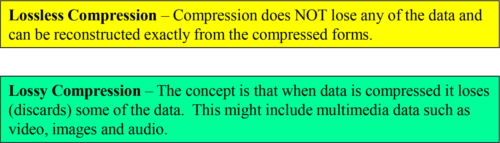
- Tera-Tom Coffing

UserDefined Compression Techniques



All of the above techniques can **also** be used for **Columnar** (Column Partitioned Tables).

Lossyand Lossless Compression



The following are lossless compression techniques:

1. **Row Compression** (Only used in Join Indexes or Hash Indexes)

2. **Multi-Value Compression** (MVC)

3. **Block-Level Compression** (BLC)

**4. Columnar Auto Compression**

Things to Remember:

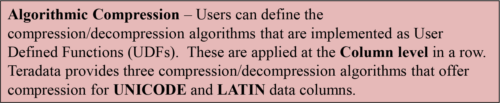
•To compress columns you use either Multi-Value or Algorithmic compression.

•To compress rows you use Compressed Hash or Join Indexes

•To compress data blocks you use Block-Level compression

•To compress containers you use the Auto Compress feature for a column-partitioned table or Join Index.

AlgorithmicCompression



The act of compressing/decompressing column data will require **extra CPU** so consideration must be given to weighing the CPU cost of compression versus the potential space and performance gains.

The algorithms must be defined as regular **scalar UDFs** . The UDFs need to be specified in the column definition during the execution of a CREATE/ALTER TABLE statement.

These algorithms are automatically invoked by the Teradata Database to compress/decompress the column data when the **data is moved into the tables** or when **data is retrieved from the tables**.

In some cases, such as when column values are **mostly unique**, Algorithmic Compression can provide better compression results than Multi-Value Compression.

You can use ALC to compress columns with the following data types:

•BYTE

•CHARACTER

•GRAPHIC

•VARBYTE

•VARCHAR

•VARGRAPHIC

BlockLevel Compression

https://www.safaribooksonline.com/library/view/v14-certification-teradata/9781940540245/img/381_1.png

Block Level Compression performs compression on whole **data blocks** at the **file system level** before the data blocks are actually written to storage.

Only primary data subtable and fallback copies can be compressed. Both objects are either compressed or both are uncompressed.

Secondary Indexes (USI/NUSI) **cannot** be compressed.

Join Indexes can be compressed.

Only the compressed form of the data block will be cached in memory, each block access must perform the data block decompression. This takes up CPU so if your system is CPU saturated, this might not be a good idea.

Temperature-Based Block Level Compression

https://www.safaribooksonline.com/library/view/v14-certification-teradata/9781940540245/img/382_1.png

Temperature-Based Block Level Compression uses the temperature of data that is maintained by Teradata Virtual Storage (TVS) to determine exactly what to compress and what not to compress.

Cold data will be compressed, warm data might also be compressed, but hot data most likely would not be compressed (unless previously compressed).

Only permanent tables are compressed using Temperature-Based Block Level Compression.

When a table is created the BLOCKLEVELCOMPRESSION option is used to control the compression. Here are the options to be set in the BLOCKLEVELCOMPRESSION option:

•**Autotemp** – Allows the system to change the compression state based on the current temperature.

•**Default** – This level of compression is controlled by the DBS Control Parameter DefaultTableMode.

•**Manual** – Identifies tables that are not managed automatically like AUTOTEMP tables are.

•**Never** – Under no circumstances will this table be block-level compressed.

TheTeradata 2690 has Hardware Compression

One important new feature of the 2690 is hardware compression.

•With automatic block level compression, customers can get as much as 3x the customer data space.

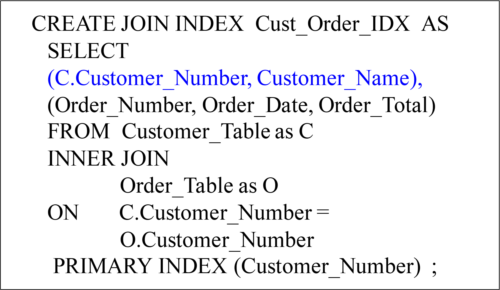
•System level scan rate (what’s also known as effective scan rate), has increased 3x because 3x more data can be scanned.

•Also, hot cache memory, which is where frequently used results are stored until not needed, has tripled as well because the data/results being stored are 3x compressed.

With compression, the system can be pushed higher because compression CPU work has been moved out of the nodes, and that CPU is available for Teradata work.

The advantage to hardware-based block-level compression is that compression and decompression of data presents **very little CPU resource contention**. Hardware-based block-level compression uses the Exar Corporation’s **Lempel-Ziv-Stac** algorithm.

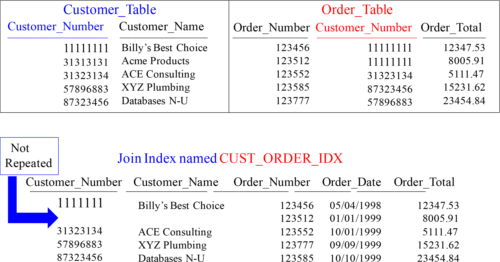
Row-Level Compression Utilizing a Compressed Join Index



A Compressed Multi-Table Join Index won’t keep repeating the same Customer\_Number and Customer\_Name, but only list it once. A visual example follows on the next page.

The point here is that the row is compressed because the repeating values (same customer\_Number) is not repeated in the Join Index. The next page will show this in action. Although you might not think of a Join Index as a form of compression, it is when it eliminates the need for duplicate values (such as a duplicate customer\_number).

AVisual of a Compressed Multi-Table Join Index



Billy’s Best Choice is Customer\_Number 11111111 and they have placed two orders, but the Customer\_Number and Customer\_Name don’t repeat unnecessarily.

ImportantInformation about Multi-Value Compression (MVC)

The COMPRESS Clause in the DDL of a table works in two different ways:

•When issued by itself (without a value), COMPRESS causes all NULL values for the column to be compressed to zero space.

•When issued with a value(s) (e.g., COMPRESS ‘Ohio’, ‘Iowa’, ‘Georgia’), the COMPRESS clause will compress every occurrence of Ohio, Iowa, and Georgia to zero space. It will also automatically compress all NULL values in the column.

•You can compress up to 255 values per column (plus nulls). The cost is that between 1 and 8 bits are placed in front of every row in the table for each column that is compressed.

You cannot compress the following:

•Components of the primary index

•Identity columns

•Volatile tables

•Derived tables

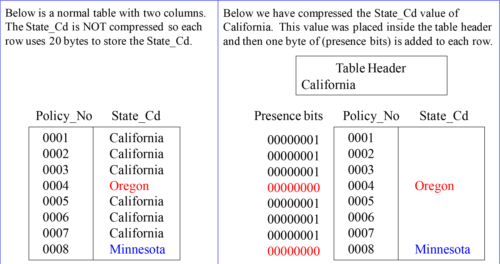
•Columns defined with a UDT, Period, and Geospatial, BLOB or CLOB data type.

•Both the referencing and referenced columns with Standard Referential Integrity cannot be compressed, but Batch and Soft RI columns can be compressed.

You can compress a value as large as 255 characters, but in Teradata 13.10 and beyond, that increases to 510 characters. Before Teradata 13.10, only fixed width columns could be compressed, but not Varchar columns can also be compressed. Compression is case-sensitive!

PresenceBytes are also used for Multi-Value Compression

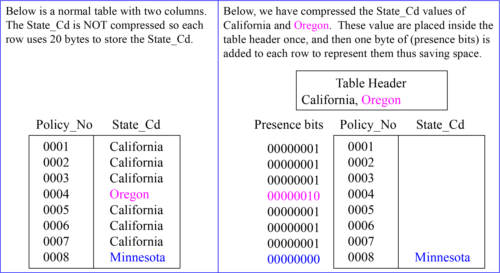
Presence Bytes are used in Multi-Value Compression. Here are the basics of how this compression works. Teradata can place the top 255 most frequent values (per column) in the Table Header, and then up to 8 presence bits can be placed in front of each row of the table (per column) to determine the value inside the row. But that value no longer exists in the row.



The above pictorial shows (logically) a normal table and then that same table compressed.

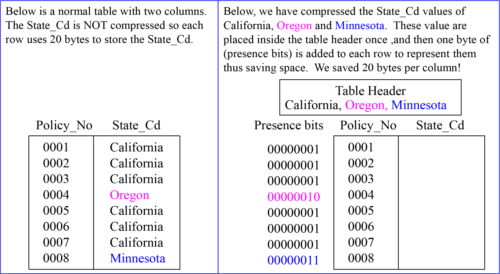
AMVC Compression Example that Compresses Two Values

Notice in our compressed example (on the right), Teradata places one byte (8 bits) in front of the row. Then, Teradata places the State\_Cd of “California” and “Oregon” inside the Table Header. Then, each time the actual row has a State\_Cd equal to “California” they set the bits = 1. This is because “California” is the first value for the column compressed in the Table Header. If the presence bits = 2, they know it is Oregon. If equal to zero, they know the value resides in the row.



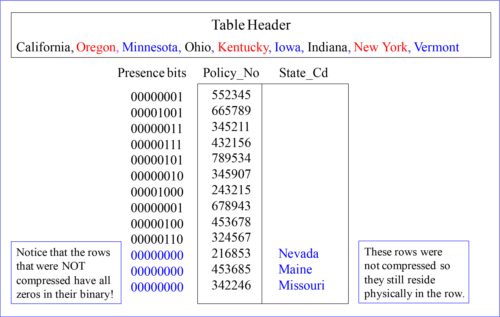
AMVC Compression Example that Compresses Three Values

Notice in our compressed example (on the right), Teradata places one byte (8 bits) in front of the row. Then, Teradata places the State\_Cd of “California” and “Oregon” inside the Table Header. Then, each time the actual row has a State\_Cd equal to “California” they set the bits = 1. This is because “California” is the first value for the column compressed in the Table Header. If the presence bits = 2, they know it is Oregon. If equal to zero, they know the value resides in the row.



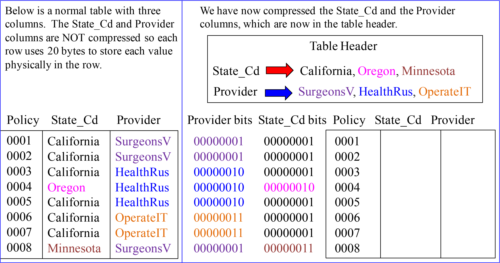
Quiz– Name that MVC Compression Value

In the slide below on the right, you get the opportunity to name the value inside the row. Use your binary skills, the Table Header, and the Presence bits to put in the correct value! Good luck.



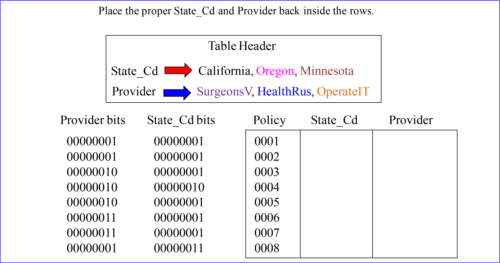
TheNext Important Concept in MVC Compression

The next two concepts are vital to getting a full understanding of compression. I will take these concepts in simple steps. The first is easy, but the second is amazing. Here is the first. Teradata can compress many columns in a table! Check out the Table Header and the column values!



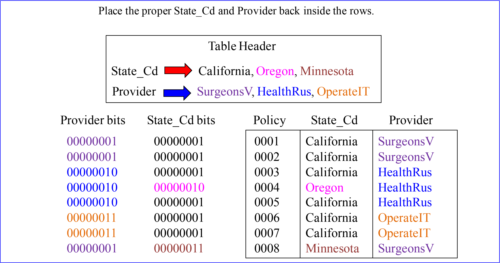
Quiz– Can you Fill in the MVC Compression Values?

Use your binary skills, the Table Header, and both the Provider bits and the State\_Cd bits to show your intelligence. If you fill this out correctly, you only have one more concept to understand!



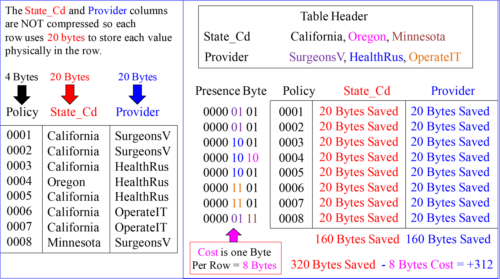
Answer– Can you Fill in the MVC Compression Values?

Use your binary skills, the Table Header, and both the Provider bits and the State\_Cd bits to show your intelligence. If you fill this out correctly, you only have one more concept to understand!



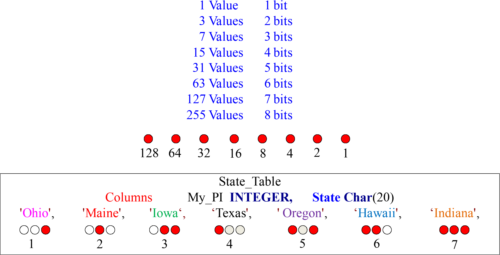
TheMulti-Value Compression (MVC) Cost vs. the Savings

OK! There are still a few things to learn, but you are on the Tera-Tom team because you proved yourself. I am proud you have arrived. Now comes another very important concept. This is your first introduction to Cost Vs. Savings. This concept will also be hammered home with advanced exercises later, but for now just understand exactly how we saved 312 Bytes!



TheCost List of MVC Compression

Below is the cost list for compression. At a maximum, you can compress up to 255 values per column. Each value is placed in the table header, and then a maximum of up to 8 bits can be placed in front of each row of the table (per column). If you have two columns being compressed, then up to 16 bits can be placed in front of each row, etc.



The above pictorial uses only three bits to represent up to seven values.

AutoCompress in Columnar Tables

Compressed unless NO AUTO COMPRESS stated.

Teradata uses many different Compression techniques.

Teradata will decide NOT to Compress some Partitions.

Decompression automatic on column retrieval.

Compression at its best for single -column partitions.

Overhead in deciding best compression techniques.

No Overhead with NO AUTO COMPRESS in CREATE.

Teradata will automatically Compress each container if applicable!

AutoCompress Techniques in Columnar Tables

Trim Compression will compress leading zeros and trailing spaces to reduce space.

NULL Compression will compress NULL Values if the column is defined as NULLABLE.

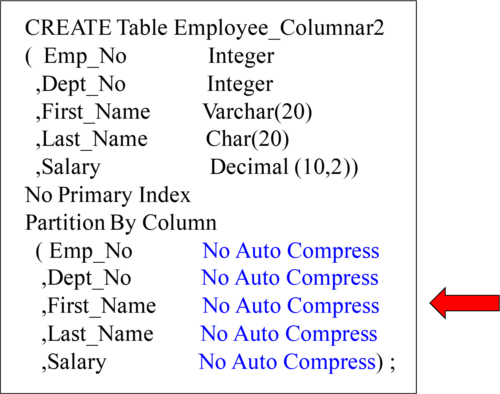
Local Dictionary Compression is similar to Multi-Value Compression where a list of values are compressed.

Run Length Encoding Compression will have values only once and then maintain an associated count.

Unicode to UTF8 Compression for all Unicode (2-byte) characters that are ASCII so only (1-Byte) is needed.

Delta on Mean Compression will get the MEAN or AVERAGE and store -1, -2, + 3 to show the difference.

ColumnarTables Automatically Compress unless NO AUTO COMPRESS



Teradata compresses Columnar columns unless NO AUTO COMPRESS is stated.