Support Workshop
BW Feature Pack

SAP AG
Berlin, June 2007
Chapter

New DB Features for BI

BI / Netweaver Integration

Support Issues
BW Features in MaxDB 7.6.

Table Clustering
- Logical Table Clustering
- Physical Table Clustering

Table Compression

Join Optimization

The BW Feature Pack is available for BI systems as of MaxDB Version 7.6 Support Package 01. This BW Feature Pack improves the performance of BI systems.

Logical Cluster (see next slide)

Physical table cluster
The CLUSTER table attribute is included in the tables. As a result of this attribute, the table contents are no longer distributed evenly among all volumes in individual blocks of eight kilobyte pages. Instead, if possible, they are stored among the volumes in larger connected blocks.

Table Compression
The PACKED table attribute causes integer values to be compressed and stored and as a result, they require less memory space on the volumes. The results in a faster I/O access and reduced displacement in the data cache.
Logical Table Clustering

- Use CLUSTER columns internally as primary key
- Ensure uniqueness of the primary key by adding a postfix sequence
- Only for tables with no user defined primary key

```
CREATE TABLE clu
  col1        VARCHAR(20),
  col2        VARCHAR(20),
  created     VARCHAR(8 )
CLUSTER (created)
```

A virtual key allows you to create a key for non-unique columns. From a technical point of view, this virtual key consists of a non-unique key field and a unique sequential number (SEQUENCE). If a table has a virtual key, logical clustering of the table also takes place.

This type of virtual key is created in the BW environment in connection with the physical clustering of the fact tables.

The report from Note 983845 adds the CLUSTER attribute to the fact tables and creates the virtual key.
Benefit: Logical Table Clustering

- Join via primary key instead of join via index
- Field experience: 50% faster join (precondition: without IO)
Challenge: Physical Table Clustering

- Database tasks read blocks from disks in 8KB units
- Scans don’t benefit from larger block sizes of storage systems and read-ahead options
Physical Table Clustering (Write)

- Sort pages in cache via the right page pointer and write them down in clusters
- Write big clusters to the end of the volume. They fill the volume from the end to the beginning. 8 KB clusters fill the volumes from start to the end

### Data Cache

<table>
<thead>
<tr>
<th>Offset</th>
<th>Volume</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4711</td>
<td>1</td>
<td>9857</td>
</tr>
<tr>
<td>7598</td>
<td>1</td>
<td>9858</td>
</tr>
<tr>
<td>9328</td>
<td>1</td>
<td>9859</td>
</tr>
<tr>
<td>9587</td>
<td>1</td>
<td>9860</td>
</tr>
</tbody>
</table>

### Converter
Physical Table Clustering (Read)

- Read page numbers from separators in index level (1) and check the block positions in converter
- Read pages of one cluster with one I/O

<table>
<thead>
<tr>
<th>Data Cache</th>
<th>Converter</th>
</tr>
</thead>
<tbody>
<tr>
<td>root level (2)</td>
<td></td>
</tr>
<tr>
<td>Index level (1)</td>
<td></td>
</tr>
<tr>
<td>leave level (0)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Page</th>
<th>Volume</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>4711</td>
<td>1</td>
<td>9857</td>
</tr>
<tr>
<td>7598</td>
<td>1</td>
<td>9858</td>
</tr>
<tr>
<td>9328</td>
<td>1</td>
<td>9859</td>
</tr>
<tr>
<td>9587</td>
<td>1</td>
<td>9860</td>
</tr>
</tbody>
</table>

© SAP 2007 / SWS 2007 BW Feature Pack / Page 8
Physical Table Clustering (System Views)

- The view DATAVOLUMES shows the CLUSTERAREASIZE
- The view FILES shows the CLUSTER attribute
- The views TABLESTORAGEDETAILS and INDEXSTORAGEDETAILS show the number of clusters in the column CLUSTERCOUNT (Select only the columns SCHEMANAME, TABLENAME, INDEXNAME and CLUSTERCOUNT to prevent a full table scan)
- Optimum number of clusters would be:
  \[
  \frac{\text{TREELEAVESSIZE}}{8} / \frac{\text{DATA_IO_BLOCK_COUNT}}{}
  \]
  e.g. 260448 / 8 / 64 ~ 509

You can check whether a table has the CLUSTER attribute by looking at the CLUSTERED column in the FILES system table.

You use selects for tables TABLESTORAGEDETAILS or INDEXSTORAGEDETAILS to evaluate the cluster quality of a table. Note that SQL accesses to these system tables where no qualification is specified in the WHERE condition for table or index names or one of the columns can lead to long runtimes.

You can use the following select to quickly determine the cluster properties for tables:
```sql
select SCHEMANAME, TABLENAME, CLUSTERCOUNT from TABLESTORAGEDETAILS
where SCHEMANAME = <schemaname> and TABLENAME = <tablename>
```

Compared with TREELEAVESSIZE:
```sql
select TREELEAVESSIZE from FILES, ROOTS
where FILES.FILEID = ROOTS.TABLEID
and ROOTS.TABLENAME = <tablename>
and ROOTS.SCHEMANAME = <schemaname>
```

The CLUSTERCOUNT value counts the data block areas that are connected and sorted in ascending order; however, these data block areas do not necessarily have to be located in the clustered area of a volume.

TREELEAVESSIZE (in kilobytes) is divided by the page size (in kilobytes), which is divided by CLUSTERCOUNT, and this determines the cluster quality. The closer the result is to the DATA_IO_BLOCK_COUNT, the better the clustering. A result with a value of 1 shows that the table is not clustered and stored.
Benefit: Physical Table Clustering

- Database tasks read blocks from disks in cluster units
- Scans benefit from larger block sizes of storage systems and read-ahead options
- Field experience: 5-6 times in average (20 times) faster scan
Table Compression: numeric columns

- Compression of numeric columns
  - In general fact tables have only numeric columns
  - If fact table has no key figures with data type FLOAT:
    Length indicator occupies 4 Bit instead of 1 Byte, because maximum value length is 15
  - Two digits are stored in 7 Bits instead of 1 Byte

You can check whether a table has the PACKED attribute and therefore, whether integer values are packed and stored by looking at the CLUSTERED column in the FILES system table.
Table Compression: 0-values and NULL

- Additional compression of frequent 0-values
  - Length occupies 4 Bit instead of 1 Byte

  Non Packed:  
  
  ![](length 1 0)  

  Packed:  0

- Omit Defined Bytes for NULL values
  - Non packed fields occupy a Defined Byte to identify NULL values
  - This Defined Byte is not needed if the database knows that records will not have NULL values in the column
  - Works only with NOT NULL
Table Compression: Requirements + Benefit

- **Requirements**
  - All non primary key columns are numeric
  - NOT NULL
  - Maximum 15 Byte per column
  - PACKED attribute will be ignored if requirements are not fulfilled

- **Benefit:**
  - Field Experience: 60-70% smaller tables and 30-40% smaller indexes
Hash Join Optimization

- Joins via hash tables make sense, if the ratio between the size of the intermediate join result and the size of the next join table exceeds a certain value.
- The parameter OPTIMIZE_JOIN_HASH_MINIMAL_RATIO defines the minimal ratio to use hashing for the next table.
- New: The decision if a hash join should be executed takes into account the intermediate result size instead of the last table.
- You can override this parameter with the hint HASHRATIO(<unsigned int>)

```sql
SELECT /* HASHRATIO(1) */ …
FROM tab1, tab2
WHERE tab1.key = tab2.field
```

- Benefit: 4-5 Seconds for each 1 Mio rows in result sets.
- Field Experience: select 2 times faster
Move Clouds of Dimensions in Optimizer Strategy

- Use knowledge about fact tables
  The MaxDB optimizer is able to identify a fact table:
  \[
  \text{ALTER TABLE } \text{<tablename> } \text{FACT | DIMENSION}
  \]

- Join from first dimension cloud to fact table
  The join between two dimension clouds can significantly increase the temporary result to be joined with the fact table.
  The system now ensures the join between one dimension cloud to the fact table before joining the other dimension clouds.

- Benefit: Prevent from long runtimes of joins due to undesirable joins between dimension clouds
Chapter

New DB Features for BI

BI / Netweaver Integration

Support Issues
Activation of BW Features

- Requirements
  - MaxDB 7.6.01
  - ABAP corrections for BW 3.x and BI 7.x, see CSN note 983845
    - Available in Support Packages (Basis and BW)
    - in urgent cases use the correction instruction
  - New cube and aggregate tables (fact tables) will be created automatically with BW Feature Pack attributes (clustered, packed)
  - Previous fact tables have to be converted with the report RSDU_CLUSTER_FACT_ADA

To use the BW Feature Pack, you must upgrade to MaxDB Version 7.6.01 or higher. For more information, see Note 983845. We recommend that you use the Save Data and Restore Data functions for the database before converting the fact tables.

If you use the Restore Data function, the tables are evenly distributed among the volumes. Fact table clustering can then be optimized.
Monitoring of packed and clustered tables

- Transaction DB50 -> Problem Analysis -> Tables/Views/Synonyms
Cluster attribute in table definition

- Only the cluster key field is visible, not the sequence field (virtual key)
Storage details of clustered table

- Factor: value corresponds with the benefit for scan
Recommended MaxDB Version

- Min. Version to activate BW Feature Pack: 7.6.01
- Recommended min. version: 7.6.02.x, x >= 10
  - Miscellaneous error corrections for Hash Join, Parallel Join
  - Hash Join on Unique Index
  - Packed table with columns of datatype FLOAT
Volatile clustering

- Remember: page clusters are assembled during save point
- Updates and inserts inside the cluster split the cluster
- Short page clusters get lost after RESTORE DATA
- No automatic and dynamic reclustering at the moment (planned for 7.6.04)
Execution Plan of a BW Query

- Only 1 dimension cloud before fact table
- Join transition to fact table via clustered key
- All tables behind the fact table are hashed
Only EXPLAIN SEQUENCE provides information about Star Schema Search
Preconditions for Star Schema Search

- Parameter OPTIMIZE_STAR_JOIN = YES
- Fact table has FACT attribute
  - FACT attribute get lost by CREATE TABLE ... LIKE
  - Set FACT attribute with report RSDU_SET_FACT_ATTR_ADA
- Only MaxDB 7.6.1: fact table has to be joined directly only with dimension tables (this precondition is dropped in MaxDB 7.6.2)
Thank you!