SAP® MaxDB™ Introduction to Query Optimization Version 7.8

Heike Gursch Christiane Hienger

THE BEST-RUN BUSINESSES RUN SAP"

# **Optimization Overview**

SAP

Introdcution

Explain

Query Rewrite

Single table optimizer

Join Optimizer

Update Statistics

File Directory counter

© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 2



SQL Optimizer is a part of the kernel of the database system. It analyzes SQL queries and selects the best search strategy for accessing the data. You can specify the search condition in the SQL statement in the WHERE clause or via the join condition.

Goals of Optimization:

An SQL performance analysis involves the identification, analysis, and optimization of SQL statements that are responsible for the highest load as regards I/O at the database level. These statements are also called "processing-intensive SQL statements".

Regular analysis and optimization of expensive SQL statements provides THE most important basis for high-performance system operation. Resource-intensive SQL statements are directly responsible for increased I/O and CPU activities, and therefore result in a poor data cache hit ratio.

More information can be found in SAP note 819324: FAQ SAP MaxDB SQL Optimization



Cost-based optimizers determine the best search strategy with the help of information about the size of the table and values within the table columns.

A cost-benefit plan is created for the various access options. The best strategy is chosen to execute the command depending on the values sent in the WHERE condition. Therefore, the eventual search strategy can only be determined at the time of execution.

#### MaxDB supports cost-based optimizers.

Before the optimization **Query Rewrite** checks if the statement can be rewritten in a reasonable way. This check and conversion is done rule-based.



What is a selection or also called search condition?

Selection conditions (search conditions) are specified in the WHERE part of the SQL statement. Within the framework of a selection condition, a column is compared with one or several actual values (for example, "MANDT = :A0", "BDAT greater than '20050821' ").

Search conditions used by the optimizer to determine the optimal search strategy are:

- Equality conditions
- Range conditions
- IN conditions
- LIKE conditions

The best strategy is chosen by the optimizer. The basis of decision making are the cost for each evaluated strategy.

The SQL Optimizer also converts conditions under certain circumstances. If a single value is specified in an IN condition multiple times, the condition is converted into an equality condition.

							S
SE	LECT * FROM	1 zztele WHE	ERE ST	R = ,We	exstr'		
NAME	VORNAME	STR	NR	PLZ	ORT	CODE	ADDINFO
A-J de Groot	Hugo	Dummy	1	10000	Berlin		Growth Mkt-Sapier
A-J de Groot	Hugo	Dummy1	2	10001	Berlin		Growth Mkt-Sapier
A-J de Groot	Hugo	Dummy2	3	10002			Growth Mkt-Sapier
A-J de Groot	Hugo	Dummy3	4	10003			Growth Mkt-Sapier
A-J de Groot	Hugo	Dummy4	5	10004			Growth Mkt-Sapier
A-J de Groot	Hugo	Wexstr	6	10005	Berlin		Growth Mkt-Sapier
A.S.	Raghavendra	Dummy	7	10006			Development India
A.S.	Raghavendra	Dummy1	8	10007			Development India
A.S.	Raghavendra	Dummy2	9	10008	Berlin		Development India
A.S.	Raghavendra	Dummy3	10	10009			Development India
A.S.	Raghavendra	Dummy4	11	10010			Development India
A.S.	Raghavendra	Wexstr	12	10011			Development India
ARARSA	Monia	Dummy	13	10012			GESTION INDUSTR

To fullfill the qualification (WHERE STR = ,Wexstr') the total table can be read and each record can be compared with the qualification.

To minimize the costs the optimizer tries to reduce the area on the table which has to be read to check the qualification and deliver the result.

If the table is sorted by the columns of the qualification, binary search is possible. An area can be found, which includes all the requested result (START and STOP key). You can sort the table by a single column or several columns.

These kind of sorts are called Index. A special kind of index is the primary key.

Primary and Secondary keys (indexes)
<ul> <li>Primary key</li> <li>The primary key is stored in the data tree (clustered)</li> <li>No separate tree for primary key !</li> <li>Parts of the primary keys are used as separator in B*trees</li> <li>The records are stored in primary key order</li> </ul>
<ul> <li>Secondary key (index)</li> <li>Construction of a separate B*tree for the secondary key values</li> <li>A secondary key does not contain physical addresses pointing to the base data but logical addresses in terms of primary keys</li> </ul>
© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 7

Each database table has a primary key (primary index). The primary key is either defined by the user or generated by the system. A user-defined primary key can consist of one or more columns. The primary key must have a unique value for each table row.

The MaxDB primary key is a UNIQUE index that is implemented directly on the data tree. The data is sorted by the primary key.

A separate B\* tree is created for a secondary key (or index). The secondary key (index) contains no physical addresses on the data tree, instead it contains logical addresses in the form of primary keys. An index is a database object that can be created for an individual column or a series of columns in a database table.

The data of the secondary index is sorted by the index column(s).

You can create an index to speed up the search for database records in a table. In technical terms, indexes are data structures (consisting of one or more primary key lists), which store parts of the data of a table in a separate B\* tree structure. This storage sorts the data according to the inverting key fields (index columns). Due to this type of storage, the table data can be accessed faster using the indexed columns.

For more information about indexes use SAP note 928037 FAQ SAP MaxDB Indexes

						С	reate Tab	le ZZTELE	Ŀ.
NAME	VORNAME	STR	NR	PLZ	ORT		NAME	CHAP(40)	
A-J de Groot	Hugo	Dummy	1	10000				CHAR(40)	
A-J de Groot	Hugo	Dummy1	2	10001			VORNAME	CHAR (20) ,	
A-J de Groot	Hugo	Dummy2	3	10002			CUD	CHAR (40)	
A-J de Groot	Hugo	Dummy3	4	10003			STR	CRAR(40),	
A-I de Groot	Hugo	Weystr	6	10004			NR	INT,	
A.S.	Raghavendra	Dummy	7	10006				(UIAD (E))	
A.S.	Raghavendra	Dummv1	8	10007			PLZ	CHAR(5),	
A.S.	Raghavendra	Dummy2	9	10008			ORT	CHAR (25) .	
A.S.	Raghavendra	Dummy3	10	10009					
A.S.	Raghavendra	Dummy4	11	10010			CODE	CHAR(31),	
A.S.	Raghavendra	Wexstr	12	10011			ADDINFO	CHAR (31)	
ABABSA	Monia	Dummy	13	10012			ADDINEO	CIIII((31),	
ABABSA	Monia	Dummy1	14	10013			PRIMARY	KEY	
ABABSA	Monia	Dummy2	15	10014			INAME NO	DNAME CUD) )	
ABABSA	Monia	Dummy3	16	10015				KNAME, SIK) )	
ABABSA	Monia	Dummy4	17	10016					
ABABSA	Monia	Wexstr	18	10017			GESTION INDUSTRIELLE		
ABBARCHI	AUGUSTO	Dummy	19	10018			gam team arienti		
ABBARCHI	AUGUSTO	Dummy1	20	10019			gam team arienti		

In this session, we use the table ZZTELE with approx. 115,000 records for the examples. The primary key is defined on the columns NAME,VORNAME,STR

The uniqueness of the primary key ensures that we only have one entry with the same name, first name and street. The records of the table are sorted in key sequence – name, first name, street

You can get the table and the primary key definition with the following SQL statement: Select \* from domain.columns where tablename = 'ZZTELE'

Ta	able	Exam	ples: ZZSTADTTEIL , ZZMASTER
PLZ	ORT	STADTTEIL	Create Table ZZSTADTTEIL
10950		dummy	
10951		dummy	(PLZ CHAR(5)),
10952		dummy	ORT CHAR(25),
10953		dummy	STADTTETT. CHAR(40)
10954		dummy	
10955		dummy	PRIMARY KEY
10056		dummy	(PLZ) )
10950		dummy	
10058		dummy	
10950		dummy	Create Mable REMACHER
10959		dummy	Create Table ZZMASTER
10960	D	dummy	(YEAR INT, 1988 Teo Hoe Sing LMU München
10961	Berlin	Kreuzberg	NAME CHAR(40) 1999 Doin Xenia FU Berlin
10962		dummy	1999 Toscani Marybeth LMU München
10963		dummy	VORNAME CHAR (20), 2000 Hofmann Martin HILBarlin
10964		dummy	UNI CHAR (40), 2000 Lucck Christina LMU München
10965		dummy	ODE CUAD (2E) 2000 MORONI STEFANO TU Berlin
10966		dummy	OKT CHAR (25), 2000 Reijer Lars TU Berlin
10967	Berlin	Kreuzberg	PRIMARY KEY 2002 Diekmann Burkhard FU Berlin
a	round rec	l 20,000 ords	(YEAR, NAME, VORNAME) ) 10 records
© SAP 2010	/MaxDB 7.8 Inf	ernals – Optimizer Intr	roduction/Page 9

To explain strategies which can be used for subqueries, the examples also refer to the table ZZSTADTTEIL with approx. 20000 records and table ZZMASTER.

The primary key of table ZZSTADTTEIL is defined on column PLZ (zip code). For each zip code there is one entry.

The table is sorted via zip code.

Table ZZMASTER has a multiple key defined on columns YEAR, NAME and VORNAME. The table is sorted by the year of the Master graduation, Name and Vorname.

## Table Examples: ZZCODE

STR	NR	PLZ	ORT	CODE
Dummy	91	10090	Berlin	
Dummy1	92	10091	Berlin	Television
Dummy2	93	10092	Berlin	Cell phone tower
Dummy3	94	10093	Berlin	
Dummy4	95	10094	Berlin	Cable TV
Wexstr	96	10095	Berlin	Cable TV
Dummy	97	10096	Berlin	
Dummy1	98	10097	Berlin	Television
Dummy2	99	10098	Berlin	Cell phone tower
Dummy3	100	10099	Berlin	
Dummy4	101	10100	Berlin	Television
Wexstr	102	10101	Berlin	Cable TV
Dummy	103	10102	Berlin	
Dummy1	104	10103	Berlin	Television
Dummy2	105	10104	Berlin	
Dummy3	106	10105	Berlin	
Dummy4	107	10106	Berlin	Television
Wexstr	108	10107	Berlin	Cable TV
Dummy	109	10108	Berlin	
Dummy1	110	10109	Berlin	Television
Dummy2	111	10110	Berlin	
Dummy3	112	10111	Berlin	
Dummy4	113	10112	Berlin	Television
Wexstr	114	10113	Berlin	Cable TV

C	reate	Table ZZCODE
(	STR	CHAR(40),
	NR	INT,
	PLZ	CHAR(5),
	ORT	CHAR (25) ,
	CODE	CHAR (31)
)		

SAP

### # of records: around 115,000

© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 10

ndexes crea	ted on ta	ble ZZ	TELE			S
년 <u>T</u> able/View <u>E</u> dit <u>G</u> oto System	<u>H</u> elp					
۲ 🖉 🗸	C 😧 🔇   🖵 🖬 🔣   🏖	10 🕄 🔛 🕅	🔞 🖪			
Table / view information						
សិខ្យ						
				_		
😽 🚖 🚺 🖬 System Configuratic 🕨	Table/View Schema	SAPWB5				
System WB5	Table / View Name	ZZTELE				
Current Status     Performance     Space     Dabs     Dapnostics     Missing Tables and Indexes     EXPLAIN     SELECT-Editor     Database Files     Critical Regions     Database console	Image: Topology         Topology <thtopology< th="">         Topology         To</thtopology<>	active Indexes     I       2     Sort     Type       1     ASC       1     ASC       2     ASC	Unused Indexes	Accesses 27.943 0	Bad Indexes           Reset Date         Time           16.03.2010         11:37:45           16.03.2010         11:37:46	Creation D Time 16.03.2010 11:37:45 16.03.2010 11:37:46 16.02.2010 11:37:46
Database Trace     SQLDBC Trace	• STR • NR	1 ASC 2 ASC		83.901	16.03.2010 11:37:44	16.03.2010 11:37:44
Error Codes	▼ ZZTELE~3	1 450		223.696	16.03.2010 11:37:46	16.03.2010 11:37:46
Messages     Database Objects     Tables/Views/Synonyms     Indexes     Database Procedures     Table Stree	✓ ZZTELE~4     ✓ VORNAME	1 ASC		3	16.03.2010 11:37:47	16.03.2010 11:37:47
Admistration     Tools     Documentation						
		SAP	·		⊳ w	/B5 (1) 001 🝸 u252059a INS

Indexes enable faster access to the rows of a table. The indexes of a table can be determined using the system table INDEXCOLUMNS.

SELECT owner, tablename, indexname, type, columnname,

sort, columnno, datatype, len, createdate

FROM domain.indexcolumns

WHERE owner = <owner>

AND schemaname = <schema>

AND tablename = <table\_name>

ORDER BY owner, tablename, indexname, columnno



The data of the base tables and the indexes are stored in B\*Tree format.

Every B\*Tree for a table has one root page with a size of 8 KB.

For more information about B\* Tree see Expert Session no. 15 SAP MaxDB No Reorganization Principle



First, the SQL parser processes an SQL statement. It performs a syntactic and semantic analysis. In the semantic analysis, tables and their column data are checked.

The optimizer determines which primary and secondary keys are available for the table and checks whether a corresponding key can be used to search for values.

The number of pages that have to be read in the primary or secondary index is determined by generating a start and a stop key. Depending on the number of pages of the table or index, it is decided whether it is worthwhile to search using the index. The number of pages of the entire table is located in the so called filedirectory

At the end, the optimizer builds a strategy with which the SQL statement will be executed.

In the R/3 System environment, SQL statements with bind variables are parsed (:A0, :A1, and so on). These bind variables may contain other specific values. All SQL statements that only vary in values are also regarded as different statements. The same SQL statement executed with different values can therefore also have different run schedules. These commands are structurally the same and are listed individually in the Command Monitor with different values.

Explain (1)	SAP
Input : EXPLAIN <select command=""></select>	
Output : Description of search strategy	
EXPLAIN is used with SELECT commands that access tables and views	
EXPLAIN does not execute the specified SELECT command.	
The explain command cannot be used with UPDATE, DELETE or INSERT commands	
© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 14	

An explain plan or access path shows how MaxDB accesses the requested data (index access, table scan, key range, key equal, index equal, and so on). An EXPLAIN plan displays the strategy the Optimizer selects to run a special SQL statement. These EXPLAINs are used to analyze long running SQL statements. An EXPLAIN plan can only be displayed for SELECT statements. Other SQL statements must be rewritten to display an explain plan. For example, an UPDATE statement can be converted into a SELECT FOR REUSE. Example:

UPDATE ZZTELE SET ADDINFO = 'ledig' WHERE NAME = 'Mueller' AND VORNAME = ' Egon' AND STR = ' Wexstraße' SELECT \* FROM ZZTELE WHERE NAME = 'Mueller' AND VORNAME = ' Egon' AND STR = ' Wexstraße' FOR REUSE

In the ABAP-based SAP application server, EXPLAIN is available in transactions ST05, DB50 and DBACockpit (in the command monitor). The SQL editor of the Database Studio can send an EXPLAIN via context menu (right mouse click) to the database. The output is shown in a separate window.

Expla	in (2)			SAP
SCHEMANAME	TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
Schema	Table 1	Names of key or index columns	Name of chosen strategy for this table	Number of pages In system table Optimizerstatistics
Schema	Table 2	Names of key or index columns	Name of chosen strategy for this table	Number of pages in system table Optimizerstatistics
	Result name		RESULT IS (NOT) COPIED, COSTVALUE IS	Estimated costs
			Applied Query Rewrite rules	1
© SAP 2010 /MaxDB 7.8	nternals – Optimizer Introduction	/Page 15		

EXPLAIN shows:

- one block for each table from the SELECT-FROM list
- the order of the strategies reflects the order of execution
- COPIED / NOT COPIED --> Result set is generated/not generated
- "Estimated costs" provides an estimation about the number of read/write accesses
- Applied Query Rewrite rules



Here is an example for an explain plan on a single table access. The Optimizer is using the strategy *In Condition for key* on column *plz.* A temporary result is created – *Result is copied.* The estimated costs have a value of 4.

Select plz from zzstadtteil where plz = '12345' or plz = '10000'

Furthermore the explain tells us the usage of Queryrewrite rules.

DistinctpullUp does not have any effects on the execution, but is required for other rules to work.

ConvertOrToIn rewrites OR predicates as IN statements.

### Query Rewrite

Query Rewrite rebuilds SQL statements by the use of rules to enable the optimizer to find the best strategy.

```
Example: ConvertOrToIn
select * from zztele
where PLZ = '10967' or PLZ = '15099' or PLZ = '12047'
SELECT "NAME", "VORNAME", "STR", "NR",
"PLZ", "ORT", "CODE", "ADDINFO"
FROM "SAPR3". "ZZTELE" AS "_T1"
WHERE PLZ in ('10967', '15099', '12047')
Parameter: EnableQueryRewrite=YES, QueryAnalysisMode=EXTENDED
```

Query Rewrite investigates the statement after the syntactical analysis.

Query Rewrite does a semantical analysis and rebuilds the statement if rules can be applied. Several rules can be applied to one query.

The rearranged statement with the possible execution plans is stored in internal format within Shared SQL or the catalog cache, respectively. The optimizer determines the best execution plan for the rearranged statement.

Query Rewrite works rule-based. Statistical data is not taken into account. There is no evaluation of data.

For more information check FAQ note 1368477 about QueryRewrite.

ging the column ACTIVE in the view QUERYREWRITERULES effect tched on or off.					
ging the column ACTIVE in the view QUERYREWRITERULES effect tched on or off.					
ging the column Active in the view Gold KTREEWER COLES energy         tched on or off.         sel. 5QL Result (1)         pelect * from queryrewriterules         view         Queryrewriterules         view         Result (1)         view         View         Combine ToAntyOrAll         YES         Convert Linn Challe external selects         Convert Linn Challe extremation         Soconvert Linn Challe extremation         BlistinctPushDown         YES       Pull up distinct information         Pill up distinct information         Pilleresuburies         VES       Pull up distinct information         Pilleresuburies       YES         Optimize Aggregates       YES         Optimize Aggregates <th>the c</th> <th></th> <th>n the</th> <th></th> <th>cte if a</th>	the c		n the		cte if a
Sol. Sol.       Sol. Sol.         Belect ** from queryrewriterules       Image: Solid content of the s			ii uie		
SQL SQL mesult (1)         Select * from queryrewriterules         Image: Solution of the second s	d on d	or off			
Set SQL Im Result (1)         Select ** from queryrewriterules         Image: Select ** from query ** from query ** from		51 011.			
RULENAME       ACTIVE       COMMENT         1       AddiocalPredicates       YES         2       CombineExternalSelects       YES         3       CombineToAnyOrAll       YES         4       ConvertExceptToAntiSemiJoin       YES         5       ConvertIntersectToSemiJoin       YES         6       ConvertInionToORQuery       YES         7       DistinctPullup       YES         8       DistinctPullup       YES         9       FlattenSubqueries       YES         10       MergeQueries       YES         12       OptimizeApredicates       YES         13       OptimizeApredicates       YES         14       OptimizeAgregates       YES         13       OptimizeAgregiates       YES         14       OptimizeAgregiates       YES         15       OptimizeAgregiates       YES         16       OptimizeAgregiates       YES         17       PushDownPredicates       YES         18       PushDownPredicates       YES         19       Optimize Predicates       YES         16       Optimize Predicates       YES         17       PushDownPredicate	SOL SOL	Besult (1)			
Select * from queryrewriterules         RULENAME       ACTIVE       COMMENT         1       AddiocalPredicates       YES       Add local predicates         2       CombineToAnyOrAll       YES       Combine external selects       Test combine ToAnyOrAll         3       CombineToAnyOrAll       YES       Combine toAnyOrAll       YES         4       ConvertExceptToAntiSemiJoin       YES       Convert IntersectToSemiJoin test combine other semi-join         5       ConvertInionToORQuery       YES       Convert Intersect ToSemiJoin       Test convert Inion to OR-query         7       DistinctPullup       YES       Pull up distinct information       Plattersubqueries         8       DistinctPushDown       YES       Platter subqueries       Test convert Inion ToRomation         9       Flattersubqueries       YES       Merge queries       Test convert Inion ToRomation         10       MergeQueries       YES       Normalize predicates       Test convert Inion ToRomation         11       Normalize predicates       YES       Optimize aggregates       Test convert Inion ToRomation         13       OptimizeApredicates       YES       Optimize aggregates       Test convert Inion ToRomation         14       OptimizeApredicates       YES       Push d	040 540				-
RULENAME       ACTIVE       COMMENT         1       AddLocalPredicates       YES       Add local predicates         2       CombineExternalSelects       YES       Combine external selects         3       CombineToAnyOrAll       YES       Combine ORed/ANDed predicates to ANY/ALL predicate         4       ConvertExceptToAntiSemiDoin       YES       Convert Intersect ToSemiJoin       YES         5       ConvertUnionToORQuery       YES       Convert Intersect to semi-join       6         6       ConvertUnionToORQuery       YES       Pull up distinct information       7         9       FlattenSubqueries       YES       Pull up distinct information       9         10       MergeQueries       YES       Normalize predicates       10         11       NormalizePredicates       YES       Optimize aggregates       11         12       OptimizeAggregates       YES       Optimize aggregates       12         13       OptimizeExpressions       YES       Optimize predicates       12         14       OptimizeExpressions       YES       Push down predicates       13         13       PushDownPredicates       YES       Push down predicates       14         14       OptimizeSubqueries <th></th> <th></th> <th></th> <th></th> <th></th>					
RULENAME         ACTIVE         COMMENT           1         AddLocalPredicates         YES         Add local predicates           2         CombineExternalSelects         YES         Combine external selects           3         CombineToAnyOrAll         YES         Combine ORed/ANDed predicates to ANY/ALL predicate           4         ConvertExceptToAntiSemiJoin         YES         Convert Except to anti-semi-join           5         ConvertIntersect ToSemiJoin         YES         Convert Union to OR-query           7         DistinctPullup         YES         Pull up distinct information           8         DistinctPullup         YES         Pull up distinct information           9         FlattenSubqueries         YES         Merge queries           10         MergeQueries         YES         Normalize predicates           12         OptimizeAggregates         YES         Optimize aggregates           13         OptimizeAggregates         YES         Optimize poins           14         OptimizePoins         YES         Push down proins           18         PushDownPredicates         YES         Push down proins           18         PushDownPredicates         YES         Push down proins           19 <td< td=""><td>select *</td><td>from queryrewriterules</td><td></td><td>A</td><td></td></td<>	select *	from queryrewriterules		A	
RULENAME         ACTIVE         COMMENT           1         AddLocalPredicates         YES         Add local predicates           2         CombineExternalSelects         YES         Combine external selects           3         CombineToAnyOrAll         YES         Combine external selects           4         ConvertExceptToAntSemiJoin         YES         Convert Except to anti-semi-join           5         ConvertUnionToORQuery         YES         Convert Intersect to semi-join           6         ConvertUnionToORQuery         YES         Convert Intersect to semi-join           7         DistinctPullUp         YES         Pull up distinct information           8         DistinctPullUp         YES         Push down distinct information           9         FlattenSubqueries         YES         Merge queries           10         MergeQueries         YES         Normalize predicates           12         OptimizeAggregates         YES         Optimize aggregates         12           13         OptimizeAggregates         YES         Optimize predicates         12           14         OptimizeAgregates         YES         Optimize predicates         13           15         OptimizePredicates         YES         Optimize s					
RULENAME         ACTIVE         COMMENT           1         AddLocalPredicates         YES         Add local predicates           2         CombineExternalSelects         YES         Combine external selects           3         CombineToAnyOrAll         YES         Combine Query AntiSemiJoin           4         ConvertExceptToAntiSemiJoin         YES         Convert Except to anti-semi-join           5         ConvertUnionToORQuery         YES         Convert Union to OR-query           7         DistinctPullup         YES         Pull up distinct information           8         DistinctPullup         YES         Pull up distinct information           9         FlattenSubqueries         YES         Plul up distinct information           10         MergeQueries         YES         Normalize predicates           11         Normalize Predicates         YES         Optimize aggregates           12         OptimizeAggregates         YES         Optimize aggregates           13         OptimizeAggregates         YES         Optimize adgregates           14         OptimizePredicates         YES         Push down joins           15         OptimizePredicates         YES         Push down predicates           16					
ROLENAME         Act Intel         Comment           1         AddiocalPredicates         YES         Addiocal predicates           2         Combine ExternalSelects         YES         Combine ORed/ANDed predicates to ANY/ALL predicate           3         Combine ToAnyOrAll         YES         Convert Except To AntiSemiJoin         YES           4         Convert Except To AntiSemiJoin         YES         Convert Except To anti-semi-join           5         Convert Intersect To SemiJoin         YES         Convert Union to OR-query           7         DistinctPullUp         YES         Pull up distinct information           8         DistinctPullUp         YES         Pull up distinct information           9         FlattenSubqueries         YES         Merge queries           10         MergeQueries         YES         Normalize Predicates           11         NormalizePredicates         YES         Optimize aggregates           12         OptimizeAggregates         YES         Optimize aggregates           13         OptimizeAggregates         YES         Optimize predicates           14         OptimizeAggregates         YES         Optimize subqueries           15         Optimize Subqueries         YES         Push down proins </th <th></th> <th></th> <th>ACTUC</th> <th>COMMENT</th> <th></th>			ACTUC	COMMENT	
1     Additional predicates     YES     Additional predicates       2     Combine External Selects     YES     Combine ORed/ANDed predicates to ANY/ALL predicate       3     Combine ToAnyOrAll     YES     Combine ORed/ANDed predicates to ANY/ALL predicate       4     ConvertExceptToAntiSemiJoin     YES     Convert Except to anti-semi-join       5     ConvertIntersectToSemiJoin     YES     Convert Union to OR-query       6     ConvertUnionToORQuery     YES     Pull up distinct information       8     DistinctPullup     YES     Push down distinct information       9     Flattens/ubqueries     YES     Merge queries       10     MergeQueries     YES     Normalize predicates       11     NormalizePredicates     YES     Optimize aggregates       12     OptimizeAggregates     YES     Optimize aggregates       13     OptimizeAggregates     YES     Optimize joins       14     OptimizePredicates     YES     Optimize joins       15     OptimizePredicates     YES     Push down projection       18     PushDownPredicates     YES     Push down projection       19     PushDownPredicates     YES     Push down projection       18     PushDownPredicates     YES     Push down projection       19 </td <td></td> <td>RULENAME</td> <td>ACTIVE</td> <td></td> <td></td>		RULENAME	ACTIVE		
2       Combine External selects       YES       Combine Oxeq/ANDed predicates to ANY/ALL predicate         3       Combine ToAnyOrAll       YES       Convert Except to anti-semi-join         4       ConvertUnion ToSenUoin       YES       Convert Except to anti-semi-join         5       ConvertUnion ToRQuery       YES       Convert Intersect to semi-join         6       ConvertUnion ToRQuery       YES       Convert Intersect to semi-join         7       DistinctPulDup       YES       Pull up distinct information         8       DistinctPushDown       YES       Push down distinct information         9       Flatten subqueries       YES       Merge queries         10       MergeQueries       YES       Mormalize predicates         11       NormalizePredicates       YES       Optimize aggregates         12       OptimizeAggregates       YES       Optimize aggregates         13       OptimizePredicates       YES       Optimize predicates         14       OptimizeByteres       YES       Optimize subqueries         17       PushDownPredicates       YES       Optimize subqueries         18       PushDownPredicates       YES       Remove dispensable constants         19       PushDownPredicates	1	AddLocalPredicates	YES	Add local predicates	
3     Combine ToAnt/SemUoin     YES     Comvert Except ToAnt/SemUoin       4     Convert Except ToAnt/SemUoin     YES     Convert Intersect to anti-semi-join       5     ConvertUnionToORQuery     YES     Convert Intersect to semi-join       6     ConvertUnionToORQuery     YES     Convert Union to OR-query       7     DistinctPullup     YES     Pull up distinct information       8     DistinctFushDown     YES     Flatten subqueries       9     FlattenSubqueries     YES     Flatten subqueries       10     MergeQueries     YES     Optimize predicates       11     NormalizePredicates     YES     Optimize aggregates       12     OptimizeAggregates     YES     Optimize expressions       13     OptimizeAggregates     YES     Optimize predicates       14     OptimizePredicates     YES     Optimize subqueries       15     OptimizeSubqueries     YES     Optimize subqueries       16     OptimizeSubqueries     YES     Push down predicates       18     PushDownProjection     YES     Remove dispensable Constants       20     RemoveDispensableConstants     YES     Remove dispensable GroupBy-columns       21     RemoveDispensableConderBy     YES     Remove dispensable GroupBy-columns	2	CombineExternalSelects	YES	Combine external selects	
4       Convert Except to Anticemuloin       YES       Convert Except to anticemuloin         5       Convert Intersect To Semuloin       YES       Convert Intersect to semi-join         6       Convert Union To OR Query       YES       Convert Intersect to semi-join         7       DistinctPullUp       YES       Pull up distinct information         8       DistinctPullUp       YES       Platten subqueries         9       FlattenSubqueries       YES       Merge queries         10       MergeQueries       YES       Merge queries         11       NormalizePredicates       YES       Optimize aggregates         12       OptimizeAggregates       YES       Optimize aggregates         13       OptimizeExpressions       YES       Optimize predicates         14       OptimizePredicates       YES       Optimize subqueries         16       OptimizeSubqueries       YES       Push down joins         18       PushDownProjection       YES       Push down predicates         20       RemoveDispensableConstants       YES       Remove dispensable GroupBy-columns         21       RemoveDispensableConstants       YES       Remove dispensable OrderBy-columns	3	CombineToAnyOrAll	YES	Combine Oked/ANDed predicates to ANY/ALL predicate	
5       Convert Union ToORQuery       YES       Convert Union to OR-query         7       DistinctPullUp       YES       Pull up distinct information         8       DistinctPullUp       YES       Pull up distinct information         9       FlattensUbqueries       YES       Flatten subqueries         10       MergeQueries       YES       Merge queries         11       NormalizePredicates       YES       Optimize aggregates         12       OptimizeAggregates       YES       Optimize aggregates         13       OptimizeDirs       YES       Optimize predicates         14       OptimizeDirs       YES       Optimize predicates         15       OptimizeSubqueries       YES       Optimize predicates         16       OptimizeSubqueries       YES       Push down joins         18       PushDownProdicates       YES       Push down projection         20       RemoveDispensableConstants       YES       Remove dispensable GroupBy-columns         21       RemoveDispensableConderby       YES       Remove dispensable GroupBy-columns	4	ConvertExcept I oAntiSemiJoin	YES	Convert Except to anti-semi-join	
6       ConvertUnion to ORQuery       YES       ConvertUnion to OR-query         7       DistinctPullUp       YES       Pull up distinct information         8       DistinctPushDown       YES       Pull up distinct information         9       FlattenSubqueries       YES       Flatten subqueries         10       MergeQueries       YES       Merge queries         11       NormalizePredicates       YES       Optimize predicates         12       OptimizeAggregates       YES       Optimize aggregates         13       OptimizeExpressions       YES       Optimize predicates         14       OptimizePredicates       YES       Optimize predicates         15       OptimizePredicates       YES       Optimize predicates         16       OptimizePredicates       YES       PushDownJoins       YES         18       PushDownPredicates       YES       Push down projection         20       RemoveDispensableConstants       YES       Remove dispensable constants         21       RemoveDispensableGroupBy       YES       Remove dispensable constants         22       RemoveDispensableGroupBy       YES       Remove dispensable Constants	5	ConvertIntersectToSemiJoin	YES	Convert Intersect to semi-join	
7       Distinct/VillUp       YES       Pull up distinct information         8       Distinct/PullUp       YES       Pull down distinct information         9       FlattenSubqueries       YES       Flatten subqueries         10       MergeQueries       YES       Merge queries         11       NormalizePredicates       YES       Normalize predicates         12       OptimizeAggregates       YES       Optimize aggregates         13       OptimizeExpressions       YES       Optimize expressions         14       OptimizePredicates       YES       Optimize predicates         16       OptimizeSubqueries       YES       Optimize subqueries         17       PushDownPredicates       YES       Push down joins         18       PushDownProjection       YES       Push down predicates         19       PushDownProjection       YES       Remove dispensable Constants         21       RemoveDispensableConderBy       YES       Remove dispensable GroupBy-columns         22       RemoveDispensableCorderBy       YES       Remove dispensable ConderBy       YES	6	ConvertUnionToORQuery	YES	Convert Union to OR-query	
8     Distinct/UshDown     YES     Push down distinct information       9     Flatten subqueries     YES     Flatten subqueries       10     MergeQueries     YES     Merge queries       11     NormalizePredicates     YES     Mormalize predicates       12     OptimizeAggregates     YES     Optimize aggregates       13     OptimizeDirpredicates     YES     Optimize predicates       14     OptimizePredicates     YES     Optimize predicates       15     OptimizeSubqueries     YES     Optimize predicates       16     OptimizeSubqueries     YES     Optimize predicates       17     PushDownJoins     YES     Push down predicates       18     PushDownPredicates     YES     Push down predicates       19     PushDownProjection     YES     Remove dispensable GroupBay       20     RemoveDispensableConstants     YES     Remove dispensable GroupBay-columns       21     RemoveDispensableCorderBay     YES     Remove dispensable Conternsto	/	DistinctPullUp	YES	Pull up distinct information	
9       Platten Subqueries       YES       Platten Subqueries         10       MergeQueries       YES       Merge queries         11       NormalizePredicates       YES       Normalize predicates         12       OptimizeAggregates       YES       Optimize aggregates         13       OptimizeExpressions       YES       Optimize expressions         14       OptimizeDians       YES       Optimize predicates         15       OptimizeEvalueries       YES       Optimize subqueries         16       OptimizeSubqueries       YES       Push down joins         18       PushDownPredicates       YES       Push down predicates         19       PushDownProjection       YES       Remove dispensable GroupBy         20       RemoveDispensableConstants       YES       Remove dispensable GroupBy         21       RemoveDispensableCorderBy       YES       Remove dispensable OrderBy-columns         22       RemoveDispensableCorderBy       YES       Remove dispensable OrderBy-columns	8	DistinctPushDown	YES	Push down distinct information	
10     MergeQueries     YES     MergeQueries       11     NormalizePredicates     YES     Optimize aggregates       12     OptimizeExpressions     YES     Optimize aggregates       13     OptimizeExpressions     YES     Optimize predicates       14     OptimizePredicates     YES     Optimize predicates       16     OptimizeExpressions     YES     Optimize predicates       17     PushDownPorions     YES     Push down predicates       18     PushDownProjection     YES     Push down predicates       20     RemoveDispensableConstants     YES     Remove dispensable GroupBy       21     RemoveDispensableCorduptery     YES     Remove dispensable GroupBy-columns       22     RemoveDispensableCorderBy     YES     Remove dispensable OrderBy-columns	9	FlattenSubqueries	YES	Flatten subqueries	
11     NormalizePredicates     YES     Normalize predicates       12     OptimizeAggregates     YES     Optimize aggregates       13     OptimizeJoins     YES     Optimize organize       14     OptimizeJoins     YES     Optimize organize       15     OptimizeJoins     YES     Optimize predicates       16     OptimizeSubqueries     YES     Optimize predicates       17     PushDownPredicates     YES     Push down joins       18     PushDownPredicates     YES     Push down predicates       19     PushDownProtection     YES     Push down projection       20     RemoveDispensableConstants     YES     Remove dispensable GroupBy-columns       21     RemoveDispensableCorderBy     YES     Remove dispensable OrderBy-columns       22     RemoveDispensableCorderBy     YES     Remove dispensable OrderBy-columns	10	MergeQueries	YES	Merge queries	
12       OptimizeAggregates       YES       Optimize aggregates         13       OptimizeExpressions       YES       Optimize expressions         14       OptimizeDins       YES       Optimize expressions         15       OptimizeEvence       YES       Optimize predicates         16       OptimizeSubqueries       YES       Optimize subqueries         17       PushDownJoins       YES       Push down joins         18       PushDownPredicates       YES       Push down predicates         19       PushDownProjection       YES       Push down projection         20       RemoveDispensableConstants       YES       Remove dispensable GroupBy         21       RemoveDispensableGroupBy       YES       Remove dispensable GroupBy-columns         22       RemoveDispensableGroupBy       YES       Remove dispensable OrderBy-columns	11	NormalizePredicates	YES	Normalize predicates	
13     Optimize:xpressions     YES     Optimize prints       14     Optimize:predicates     YES     Optimize prints       15     Optimize:Subqueries     YES     Optimize predicates       16     Optimize:Subqueries     YES     Optimize subqueries       17     PushDownDoins     YES     Push down joins       18     PushDownPredicates     YES     Push down predicates       19     PushDownProjection     YES     Push down projection       20     RemoveDispensableConstants     YES     Remove dispensable GroupBy-columns       21     RemoveDispensableCorderBy     YES     Remove dispensable OrderBy-columns       22     RemoveDispensableCorderBy     YES     Remove dispensable OrderBy-columns	12	OptimizeAggregates	YES	Optimize aggregates	
14     Optimize Joins     YES     Optimize predicates       15     OptimizePredicates     YES     Optimize predicates       16     OptimizeSubqueries     YES     Optimize predicates       17     PushDownJoins     YES     Push down joins       18     PushDownPredicates     YES     Push down predicates       19     PushDownProjection     YES     Push down projection       20     RemoveDispensableConstants     YES     Remove dispensable GroupBy-columns       21     RemoveDispensableOrderBy     YES     Remove dispensable OrderBy-columns       22     RemoveDispensableOrderBy     YES     Remove dispensable OrderBy-columns	13	OptimizeExpressions	YES	Optimize expressions	
15     OptimizeFredicates     YES     Optimize subqueries       16     OptimizeSubqueries     YES     Optimize subqueries       17     PushDownJoins     YES     Push down proins       18     PushDownProdicates     YES     Push down projection       19     PushDownProjection     YES     Push down projection       20     RemoveDispensableGroupBy     YES     Remove dispensable GroupBy-columns       21     RemoveDispensableGroupBy     YES     Remove dispensable GroupBy-columns       22     RemoveDispensableOrderBy     YES     Remove dispensable OrderBy-columns	14	OptimizeJoins	YES	Optimize joins	
16     Optimize Subqueries     YES     Optimize Subqueries       17     PushDownJoins     YES     Push down joins       18     PushDownPredicates     YES     Push down predicates       19     PushDownProjection     YES     Push down projection       20     RemoveDispensableConstants     YES     Remove dispensable GroupBy-columns       21     RemoveDispensableGroupBy     YES     Remove dispensable GroupBy-columns       22     RemoveDispensableGroupBy     YES     Remove dispensable OrderBy-columns	15	OptimizePredicates	YES	Optimize predicates	
17     PoshDownProdicates     YES     Posh down points       18     PushDownProdicates     YES     Push down predicates       19     PushDownProjection     YES     Push down projection       20     RemoveDispensableConstants     YES     Remove dispensable constants       21     RemoveDispensableConderBy     YES     Remove dispensable GroupBy-columns       22     RemoveDispensableConderBy     YES     Remove dispensable OrderBy-columns	10	OptimizeSubqueries	YES	Optimize subqueries	
10     PushDownProjection     Push down projection       19     PushDownProjection     YES       20     RemoveDispensableConstants     YES       21     RemoveDispensableGroupBy     YES       22     RemoveDispensableConderBy     YES       23     RemoveDispensableConderBy     YES	1/	PushDownDoins	VES	Push down joins	
10         Positiowin Pojection         Positiowin Pojection           20         RemoveDispensableConstants         YES         Remove dispensable constants           21         RemoveDispensableGroupBy         YES         Remove dispensable GroupBy-columns           22         RemoveDispensableOrderBy         YES         Remove dispensable OrderBy-columns	10	PushDownProjection	VES	Push down projection	
21         RemoveDispensableConupBy         YES         Remove dispensable GroupBy-columns           22         RemoveDispensableOrderBy         YES         Remove dispensable OrderBy-columns	20	Pusi DownProjection RemoveDispensableConstants	VES	Pash down projection Remove dispensable constants	
22 RemoveDispensableOrderBy YES Remove dispensable OrderBy-columns	20	RemoveDispensableConstants	VES	Remove dispensable Constants	
22 Removed spensable or derby TES Removed spensable or derby-coldnins	22	RemoveDispensableOrderBy	VES	Remove dispensable Groupby-countris	
23 Peorder Joins VES Peorder joins	22	ReinovebisperisableOrderby Reorder Joins	VES	Reinove dispensable order by-columns	
24 Depredicables VFS Depredicables	24	PeorderDredicates	VES	Peorder predicates	
25 Reorder Inductes IES Reorder under	25	Reordert Ipions	VES	Reorder unions	
26 SubstituteBuchvions VES Substitute right cide of a buchvion by a from-select	25	SubstituteBushy Joins	VES	Substitute right side of a busby join by a from-select	
27 Substitute/Substitu	27	SubstituteViews	VES	Substitute complex views and join views by the corresponding from-select	

You can influence the use of Query Rewrite by setting the parameter EnableQueryRewrite.

Furthermore you have the possibility to switch single rules on or off. Use an UPDATE statement on table QUERYREWRITERULES to set the attribute ACTIVE for the corresponding rule to YES or NO.

#### UPDATE queryrewriterules

SET ACTIVE = 'YES' WHERE RULENAME = 'AddLocalPredicates'

To activate the rule changes to all applications execute: diagnose share parse reject

	itorin	ng Query R	lewr	ite SAP
EXPLA	AIN QU	ERYREWRITE	show	s the result of Query Rewrite as SQL statement.
explain	quervrewrite			
select d	listinct *			
Troth 22	r			
	L			
	STATEMEN	т		
1	SELECT "N	AME", "VORNAME", "STR", "NR", "	"PLZ", "ORT",	"CODE", "ADDINFO" FROM "SAPA1S"."ZZTELE" AS "_T1"
1				
he view	N MON	ITOR indicates	s how c	often rules were applied
select *	from monitor	and i		
select * where t	from monitor type = 'REWRI'	re'		
select * where t	* from monitor type = 'REWRI'	ΓΕ'		
select * where t	from monitor type = 'REWRI'		VALUE	
select *	from monitor type = 'REWRI'	DESCRIPTION	VALUE	
select * where t	from monitor type = 'REWRI'	DESCRIPTION SubstituteViews	VALUE	
select " where I	from monitor type = 'REWRI' REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPull In	VALUE 1960631 0 1767894	
select " where I 1 2 3 4	from monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPullUp DistinctPulsDown	VALUE 1960631 0 1767894 114373	
select * where I 1 2 3 4 5	from monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteUshyJoins DistinctPullUp DistinctPushDown OntimizeSuboueries	VALUE 1960631 0 1767894 114373 0	
select * where 1 1 2 3 4 5 6	from monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPullup DistinctPushDown OptimizeFudqueries OptimizeFedicates	VALUE 1960631 0 1767894 114373 0 8543	
select * where I 1 2 3 4 5 6 7	from monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPullUp DistinctPushDown OptimizeSubqueries OptimizeFredicates OptimizeFreesions	VALUE 1960631 0 1767894 114373 0 8543 0	
select 4 where 1 1 2 3 4 5 6 7 8	From monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteUshyJoins DistinctPullUp DistinctPushDown OptimizeSubqueries OptimizeFxploates OptimizeFxpressions OptimizeFxpressions OptimizeFxpressions	VALUE 1960631 0 1767894 114373 0 8543 0 8543 8	
select 4 where 1 1 2 3 4 5 6 7 7 8 9	From monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPullup DistinctPushDown OptimizePoedicates OptimizePedicates OptimizePoins OptimizePoins OptimizePains	VALUE 1960631 0 1767894 114373 0 8543 0 8543 0 8 0 0	
select 4 where 1 1 2 3 4 5 6 7 8 9 10	From monikor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPullUp DistinctPushDown OptimizeSubqueries OptimizePredicates OptimizeExpressions OptimizeJoins OptimizeAggregates ReorderPredicates	VALUE 1960631 0 1767894 114373 0 8543 0 8 0 0 0 0 0 0 0 0 0 0 0 0 0	
select 4 where 1 2 3 4 5 6 7 8 9 9 10	From monitor type = 'REWRIT REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteViews DistinctPullUp DistinctPushDown OptimizeSubqueries OptimizeDubqueries OptimizeDubqueries OptimizeJoins OptimizeJoins OptimizeQgregates ReorderPredicates ReorderPredicates ReorderPredicates	VALUE 1960631 0 1767894 114373 0 8543 0 8 0 0 8 0 0 43	
select 4 where 1 2 3 4 5 6 7 7 8 9 10 11 12	From monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPuBlUp DistinctPuBlUp OptimizeForedicates OptimizePredicates OptimizePagregates ReorderPredicates ReorderPredicates ReorderPions ReorderLions	VALUE 1960631 0 1767894 114373 0 8543 0 8543 0 8 0 0 0 0 0 0 0 0 0 0 0 0 0	
select <sup>4</sup> where 1 1 2 3 4 5 6 7 8 9 10 11 11 12 13	From monikor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPuBlUp DistinctPuBlUp DistinctPushDown OptimizePredicates OptimizeAggregates ReorderPredicates ReorderPredicates ReorderInions PushDownPredicates	VALUE 1960631 0 1767894 114373 0 8543 0 8 0 0 43 0 1828128	
select 4 where 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14	From monitor type = 'REWRIT REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteViews DistinctPusIDown OptimizeSubqueries OptimizeDubqueries OptimizeDubqueries OptimizeJoins OptimizeJoins OptimizeJoins OptimizeJoins ReorderPredicates ReorderPredicates ReorderDins ReorderUnions PushDownPredicates PushDownPredicates	VALUE 1960631 0 1767894 114373 0 8543 0 8 0 0 43 0 1828128 1813928	
select <sup>4</sup> where 1 2 3 4 5 5 6 7 7 8 9 10 11 12 13 14 15	From monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPuBlUp DistinctPuBlUp OptimizeFredicates OptimizePredicates OptimizeAgregates ReorderPredicates ReorderPredicates ReorderPrioins PushDownProjection PushDownProjection	VALUE 1960631 0 1767894 114373 0 8543 0 8 0 0 43 0 1828128 1813928 0	
select <sup>4</sup> where 1 2 3 4 5 6 7 8 9 9 10 11 12 13 14 15 16	From monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPushDown OptimizeFudqueries OptimizePerdicates OptimizeAggregates ReorderPredicates ReorderPredicates ReorderPredicates ReorderUnions PushDownPredicates PushDownProjection PushDownDoins FlattenStituteries	VALUE 1960631 0 1767894 114373 0 8543 0 8543 0 0 43 0 1826128 1813928 0 1360	
select 4 where 1 2 3 4 5 6 7 7 8 9 10 11 11 12 13 14 15 16 17	From monitor type = 'REWRIT REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteUshyJoins DistinctPullUp DistinctPushDown OptimizeDubqueries OptimizeDubqueries OptimizeJoins OptimizeAggregates ReorderPredicates ReorderPredicates ReorderPredicates ReorderDioins ReorderUnions PushDownPredicates PushDownPredicates PushDownDoins FlattenSubqueries	VALUE 1960631 0 1767894 114373 0 8 8 0 43 0 1828128 1813928 0 1360 1950877	
select 4 where 1 2 3 4 5 6 7 7 8 9 10 11 12 13 14 15 16 17 19	From monitor type = 'REWRI' REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE REWRITE	DESCRIPTION SubstituteViews SubstituteBushyJoins DistinctPuBlUp OptimizeSubqueries OptimizePeredicates OptimizePagregates ReorderPredicates ReorderPredicates ReorderPredicates ReorderUnions PushDownProjection PushDownProjection PushDownProjection PushDownProjection PushDownSis FlattenSubqueries MeraeQueries	VALUE 1960631 0 1767894 114373 0 8543 0 8 0 0 43 0 1828128 1813928 0 1360 1959877 0	

EXPLAIN QUERYREWRITE displays the SQL statement after it has been converted by QueryRewrite.

SELECT \* FROM MONITOR WHERE Type = ,REWRITE'

eryrewrite se = '12345' or	elect plz from zzstadtteil or plz = '10000'	
STATEMEN	IT	
SELECT T2.	"PLZ" FROM "SAPWB5"."ZZSTADTTEIL" AS T2 W	/HERE T2."PLZ" IN ('12345', '10000')
ECT T2." )00')	"PLZ" FROM "SAPWB5"."ZZSTADTT	EIL" AS T2 WHERE T2."PLZ" IN ('1234
ECT T2." 000')	"PLZ" FROM "SAPWB5"."ZZSTADTT	EIL" AS T2 WHERE T2."PLZ" IN ('12345
ECT T2." 000')	"PLZ" FROM "SAPWB5"."ZZSTADTT	EIL" AS T2 WHERE T2."PLZ" IN ('1234
ECT T2." 000')	"PLZ" FROM "SAPWB5"."ZZSTADTT	EIL" AS T2 WHERE T2."PLZ" IN ('1234

Use the statement EXPLAIN QUERYREWRITE <select> to display the result of the rewrite. In SAP MaxDB versions below 7.8, this output is restricted to 2500 characters. This means that truncated rewrites may occur.

Note that the result of EXPLAIN QUERYREWRITE is not an SQL statement that can necessarily be executed using the Database Studio. You can use the right mouse to expand the complete SQL statement.

Further information about QueryRewrite: SAP note: 1368477 FAQ: SAP MaxDB QueryRewrite

SELEC	WHERE	ME,NA FRO ADDIN	AME,PL M ZZTI Fo Lik	.Z,ORT,S ELE <mark>(E 'Traini</mark>	TR,ADDINFO <b>ng%'</b>		
VORNAME	NAME	PLZ	ORT	STR	ADDINFO		
MICHELA	ANDREONI	10108	Berlin	Dummy	Training Administration		
MICHELA	ANDREONI	10109	Berlin	Dummy1	Training Administration		
MICHELA	ANDREONI	10110	Berlin	Dummy2	Training Administration		
MICHELA	ANDREONI	10111	Berlin	Dummy3	Training Administration		
MICHELA	ANDREONI	10112	Berlin	Dummy4	Training Administration		
MICHELA	ANDREONI	10113	Berlin	Wexstr	Training Administration	Search Strategy	
Svein	Aasen	10186	Berlin	Dummy	Training - NEW EMP	STRATEGY	PAGECOUN
Svein	Aasen	10187	Berlin	Dummy1	Training - NEW EMP	TABLE SCAN	3212
Svein	Aasen	10188	Berlin	Dummy2	Training - NEW EMP	RESULT IS NOT COPIED , COSTVALUE IS	3212
Svein	Aasen	10189	Berlin	Dummy3	Training - NEW EMP	QUERYREWRITE : APPLIED RULES:	1
Svein	Aasen	10190	Berlin	Dummy4	Training - NEW EMP	DistinctPullOp	1
Svein	Aasen	10191	Berlin	Wexstr	Training - NEW EMP		
Keiichiro	Abe	10258	Berlin	Dummy	Training Mgmt & Admin		
Keiichiro	Abe	10259	Berlin	Dummy1	Training Mgmt & Admin		
Keiichiro	Abe	10260	Berlin	Dummy2	Training Mgmt & Admin		

**Search conditions** are specified in the WHERE part of an SQL statement. The WHERE part is used by the optimizer to find the best **search strategy** to deliver the result.

Note: The **order** of the column specification in the SELECT list only influences the optimizer search strategy for DISTINCT statements.

Without an explicit order by option in the WHERE condition the result is sorted by the primary key *Name, Vorname, Str.* 

Exception: When an Index only strategy is used the result is sorted by the index order. Always ORDER BY should be used if a special sort of the result is requested.



An index contains the data of the secondary key as well as the respective primary key. Using the primary key, the data can be found in the base table. For each index, a B\* tree is created, which is sorted according to the values of the secondary key.

There is no record ID or anything similar. The unique ID of a record is the primary key.

If no primary key was specified with the table creation, the database generates the internal field SYSKEY of the type CHAR(8) BYTE. This field is filled with unique values.

Searching via an index is relatively costly.

On the following slides you will find examples of search strategies. The list of strategies is not complete. A complete list of search strategies can be found in the documentation.

Basic Information -> Background Knowledge -> SQL Optimizer -> Search Strategy -> List of all search strategies



Remember the key definition of table ZZTELE is Name, Vorname, Str

EQUAL CONDITION FOR KEY provides an efficient access path through "direct access" to the base table.

The optimizer takes decision for this strategy already at the time of the parsing because, independent of the data in the search conditions, no better search strategy is possible.

SELECT * WHERE 1 AND 7 AND 3	FROM zzt Name = Vorname = Str =	ele 'Aa 'An 'Al	ron' ton' t Moabit	±'				SA
CHEMANAME	TABLENAME	-	COLUMN O	R INDEX	STRATEGY			PAGECOUNT
APWR5	77TELE				FOUAL CON	3200		
	and I blue		NAME		(USED KE	5200		
					(USED KE			
				STR		(USED KEY COLUMN)		
	JDBC_CURSO	DR_44			RESULT IS	IS 1		
	JDBC_CURSOR_44 JDBC_CURSOR_44				QUERYREWR			
					DistinctPu	1		
	ORNAME	STR	2	NR	PLZ	ORT	CODE	ADDINFO
aron /	nton	Alt	Moabit	96	10559	Berlin	X	Testperson



If a portion of the start of the primary key is specified in the WHERE condition, the strategy RANGE CONDITION FOR KEY will be executed.

A special case of key range is the table scan. The start key is located at the beginning of the table and the stop key at the end of the table.

The base table will be searched completely (TABLE SCAN).

An intermediate result set is not generated.

				SA
SELECT * F	ROM zztele W	HERE Name = '	Schmidt'	
	1	AND Vorname	like 'A%`	
SCHEMANAME	TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
SAPWB5	ZZTELE		RANGE CONDITION FOR KEY	3212
		NAME	(USED KEY COLUMN)	
		VORNAME	(USED KEY COLUMN)	
	JDBC_CURSOR_28		RESULT IS NOT COPIED , COSTVALUE IS	1
	JDBC_CURSOR_28		QUERYREWRITE : APPLIED RULES:	
	JDBC_CURSOR_28		DistinctPullUp	1
SELECT *	FROM zztele	COLUMN OR INDEX	STRATEGY	PAGECOUNT
	77TELE		TABLESCAN	3212
SAPWR5			RESULT IS NOT COPIED COSTVALUE IS	3212
SAPWB5	IDRC CURSOR 29			JLIL
SAPWB5	JDBC_CURSOR_29		OUERYREWRITE : APPLIED RULES:	

primary key order



The IN condition can be placed on each field of a primary key.

Only one IN condition is taken into account.

The primary key fields that precede the field with the IN condition may only be specified in an EQUAL condition.

An intermediate result set is generated. The result set is sorted according to the primary key.

As of version 7.4, the optimizer checks whether the RANGE CONDITION FOR KEY is advantageous. This happens if the values in the IN condition are close to each other. Example:

SELECT \* FROM zztele WHERE name IN ('Scheu', 'Schmidt')

There are additional names in the table that are located between the values 'Scheu' and 'Schmidt'. Thus, using this search condition, records are also included that do not belong to the results set. However, the strategy is more favorable since only one start and stop key have to be determined.

	S				
SELECT * WHERE Nam	FROM zztele me IN ('Muell	er', 'S	Schmidt'	')	
SCHEMANAME	TABLENAME	COLUMN	OR_INDEX	STRATEGY	PAGECOUNT
SAPWB5	ZZTELE			IN CONDITION FOR KEY	3212
		NAME		(USED KEY COLUMN)	
	JDBC_CURSOR_45			RESULT IS COPIED , COSTVALUE IS	69
	JDBC_CURSOR_45			QUERYREWRITE : APPLIED RULES:	
SELECT *	JDBC_CURSOR_45 JDBC_CURSOR_45 FROM zztele			QUERYREWRITE : APPLIED RULES: DistinctPullUp	1
SELECT * : WHERE Nam	JDBC_CURSOR_45 JDBC_CURSOR_45 FROM zztele e IN ('Scheu COLUMN_O	', 'Schr R_INDEX	midt') STRATEG	QUERYREWRITE : APPLIED RULES: DistinctPullUp	1 PAGECOUNT
SELECT * 3 WHERE Nam TABLENAME ZZTELE	JDBC_CURSOR_45 JDBC_CURSOR_45 FROM zztele e IN ('Scheu COLUMN_O	', 'Schr R_INDEX	midt') STRATEG RANGE C	QUERYREWRITE : APPLIED RULES: DistinctPullUp Y ONDITION FOR KEY	1 PAGECOUNT 3212
SELECT * 3 WHERE Nam TABLENAME ZZTELE	JDBC_CURSOR_45 JDBC_CURSOR_45 FROM zztele e IN ('Scheu COLUMN_O NAME	', 'Schr R_INDEX	midt') STRATEG RANGE C (USED	QUERYREWRITE : APPLIED RULES: DistinctPullUp Y ONDITION FOR KEY KEY COLUMN)	1 PAGECOUNT 3212
SELECT * : WHERE Nam TABLENAME ZZTELE	JDBC_CURSOR_45 JDBC_CURSOR_45 FROM zztele e IN ('Scheu COLUMN_O NAME	', 'Schr R_INDEX	midt') STRATEG RANGE C (USED RESULT	QUERYREWRITE : APPLIED RULES: DistinctPullUp Y ONDITION FOR KEY KEY COLUMN) IS NOT COPIED , COSTVALUE IS	1 PAGECOUNT 3212 32
SELECT * 3 WHERE Nam TABLENAME ZZTELE DBC_CURSOF	JDBC_CURSOR_45 JDBC_CURSOR_45 FROM zztele e IN ('Scheu COLUMN_O NAME	', 'Schr R_INDEX	midt') STRATEG RANGE C (USED RESULT QUERYRE	QUERYREWRITE : APPLIED RULES: DistinctPullUp Y ONDITION FOR KEY KEY COLUMN) IS NOT COPIED , COSTVALUE IS WRITE : APPLIED RULES:	PAGECOUNT 3212 32

An intermediate result is created for IN CONDITIONs.

No intermediate result is created for RANGE CONDITIONs.

Primary key order



When determining the strategy, additional costs (index\_overhead) for accessing the base data via the index are taken into account.

The optimizer also selects the strategy EQUAL CONDITION FOR INDEX, if all fields of a multiple index in the WHERE condition are specified with an equal condition.

An intermediate result set is not generated.

## **EQUAL CONDITION FOR INDEX - Example**

CREATE INDEX "ZZTELE~3" ON ZZTELE(PLZ)

### SELECT \* FROM zztele WHERE plz = '12345'

SCHEMANAME	TABLENAM	TABLENAME		UMN_OR	_INDEX	STRATEG	PAGECOUNT		
SAPWB5		ZZT	ELE~3		EQUAL C	2105			
			PLZ			(USED	INDEX CO	LUMN)	
	JDBC_CURS	JDBC_CURSOR_55				RESULT	IS NOT C	OPIED , COSTVALUE IS	2
	JDBC_CURS				QUERYR				
	JDBC_CURS	SOR_55				Distinc	tPullUp		1
NAME	VORNAME	STR		NR	PLZ	ORT	CODE	ADDINFO	
Antao	Sanjiv	Wexst	r	51	12345			Consulting-Bomba	у
Dry	Marie	Dumn	ny2	162	12345			Central Registration	1
Huebel	Ralf	Wexst	r	18	12345			FI Dev. Internationa	1
Marek	Peter	Dumn	ny2	129	12345			RIVA Entwicklung R	/2
Ringling	Sven	Wexst	r	240	12345			Test R/3-HR	
	Takayashi	Dump	0.12	96	12345			Log - 2	

SAF

© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 30



If a part of the start of the secondary key is specified in the WHERE condition, the strategy RANGE CONDITION FOR INDEX will be executed.

An intermediate result set is not generated. The result is sorted by the secondary key.

The index scan is a special index range with start key at the beginning of the index and stop key and the end of the index.

The index scan is only used for ORDER BY.

During an INDEX SCAN, all entries are read via the index in the order of the secondary key. An intermediate result set is not generated.

As of version 7.4, NULL values are also included in single indexes. Thus, this strategy can be used on all indexes.

<b>RANGE CONDITION FOR INDEX - Example</b>									
CREATE II SELECT * WHERE pl:	NDE FF z E	IX "Z: ROM Z BETWE	ZTELE~3" ztele EN '1010(	ол 1А 'С	ZZTE ND '10	LE (PL2	3)		
SCHEMANA	ME	TABLE	ENAME	COL	UMN_	DR_INDEX	STR	ATEGY	PAGECOUNT
SAPWB5	APWB5 ZZTELE		.E	ZZTELE~3			RAN	NGE CONDITION FOR INDEX	2105
				PLZ			(	USED INDEX COLUMN)	
			CURSOR 72				RE	ESULT IS NOT COPIED COSTVALUE IS	245
		IDBC	CURSOR 72				011		
		JDBC_	CURSOR_72				Di	istinctPullUp	1
NAME	VO	RNAME	STR	NR	PLZ	ORT	CODE	ADDINFO	
ALZIATI	DO	RIANO	Dummy4	101	10100	Berlin		ISU Finance Sector	
Del Rio	Ma	rcela	Dummy1	212	10100	Berlin		Presales ARG	
Hirn	Ma	nfred	Dummy4	68	10100	Berlin		Log.Entw. Auftr/Vers/Faktura	
Luengen	Eric		Dummy1	179	10100	Berlin		Log.Entw. Grunddaten	
Rasanayagam	Raji	iv	Dummy4	35	10100	Berlin		Dallas - Early Watch	
Teusch	Pat	ricia	Dummy1	146	10100	Berlin		Communications Media - Train	
ALZIATI	DO	RIANO	Wexstr	102	10101	Berlin		ISU Finance Sector	
Del Rio	Ma	rcela	Dummy2	213	10101	Berlin		Presales ARG	
Hirn	Ma	nfred	Wexstr	69	10101	Berlin		Log.Entw. Auftr/Vers/Faktura	
Luengen	Eric		Dummy2	180	10101	Berlin		Log.Entw. Grunddaten	

© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 32

The index scan is used if an ORDER BY is added to the SQL statement.



A secondary key can be taken into account for an IN CONDITION. Only one IN CONDITION is taken into account.

The secondary key fields that precede the field with the IN condition may only be specified in an EQUAL CONDITION.

The result set is sorted according to the secondary key.

An intermediate result set is generated.

	r = 'We	eie exstr'	AND	nr '	—— 、 тм (З	255	5)	
CHEMANAME	TABLEN	AME	COL		R_INDE	STR	ATEGY	PAGECOUNT
SAPWB5	ZZTELE		ZZTELE~2			IN C	ONDITION FOR INDEX	2048
			STR			(1	ISED INDEX COLUMN)	
			NR			(	ISED INDEX COLUMN)	
						P		52
	JDBC_CU	IDCOD 24				0115	ESOLITIS COPIED , COSTVALOE IS	, J2
	JDBC_CO	IKSUK_24				QUE	RYREWRITE : APPLIED ROLES:	
	JDBC_CU	IRSOR_24				Dis	stinctPullUp	1
NAME	VORNAME	STR	NR	PLZ	ORT	CODE	ADDINFO	
Abe	Etsuko	Wexstr	3	10257	Berlin		Best Practice Library	
Agnes	Darci	Wexstr	3	10767	Berlin		Pre-Sales-Atlanta	
Alexander	William	Wexstr	3	11277			Performance & Tuning - ATAC	
Amidei	Lester	Wexstr	3	11787			O&G Sales	
Annweiler	Denise	Wexstr	3	12297			Empfang EVZ	
Armstrong	Patricia	Wexstr	3	12807			HR Operations	
Axmann	Bernd	Wexstr	3	13317			Vertrieb München	
BORTOLAN	EGIDIO	Wexstr	3	13827			Training Administration	
Baethke	David	Wexstr	3	14337			Atlanta - Consulting - Malon	
Bandula	Diane	Wexstr	3	14847			Sales Health care	
Bartel-Moufang	Diana	Wexstr	3	15357			Communications Media - Knowl	



The list above shows the properties which are additionally checked by the optimizer and therefore lower the costs of an SQL statement.

Examples:

*Index only* strategy can be used if a SELECT statement only addresses columns that are also contained in an index (SELECT list and WHERE condition)

An intermediate result is not necessary to create if the ORDER BY is in key order or in order of an index

If the SQL statement specifies a MIN or MAX operation of a key or index column no intermediate result has to be created.

If the SQL statement specifies a DISTINCT on a key or index column the distinct can be verified directly on the key or index.



Remember: in each index the primary key is also part of the data. The primary key is used as separator in the index B\* tree.

If a SELECT statement only addresses columns that are also contained in an index (SELECT list, WHERE clause), then only this index will be accessed for the execution of the command.

Advantage:

- In some cases, significantly fewer pages that have to be read
- Optimal usage of sorting of secondary and primary keys in the index
- No additional access to the base table
## **ONLY INDEX ACCESSED - Example**

```
SELECT name, plz
FROM zztele
WHERE plz = '12345'
```

DELIVANIE		COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
TELE		ZZTELE~3	EQUAL CONDITION FOR INDEX	2105
			ONLY INDEX ACCESSED	
		PLZ	(USED INDEX COLUMN)	
BC_CURSOR	108		RESULT IS NOT COPIED , COSTVALUE IS	1
NAME	PLZ			
A	10550			
Aaron	10228			
Aaron Adams	10559			
Adams Dettmann	10559 10559			
Aaron Adams Dettmann Hoehn	10559 10559 10559			
Aaron Adams Dettmann Hoehn Lyngstad	10559 10559 10559 10559			
Adams Dettmann Hoehn Lyngstad Redmond	10559 10559 10559 10559 10559			

SAP



An ORDER BY specification influences the choice of the Optimizer strategy.

The above SQL statement without the ORDER BY specification would be executed via a key range directly on the table. But with the ORDER BY specification (by secondary key - zztele~2) the strategy will change to an INDEX SCAN.

It is more expensive to read the data via key range and do the sort afterwards then to use an index which is already sorted like the ORDER BY specification and access the rows in the specified order in the primary table.

During the INDEX SCAN, all entries are read via the index in the order of the secondary key. An intermediate result set is not generated.

SELEC	T * F BETW C	'ROI IEEI ORDI	M zztele N 'A' and ER BY str	WHEI l'T c, n:	RE nam '	ne			
TABLENAM	E	CO		DEX	STRATEG	βY			PAGECOUNT
ZZTELE		ZZI	TELE~2		INDEX SO	CAN			2048
		NA	ME		(USED	KEY COL	UMN)		
JDBC_CURS	OR_40				RESULT	IS NOT	COPIED,	COSTVALUE IS	5260
JDBC CURS	OR 40				QUERYR	EWRITE :	APPLIED	RULES:	
JDBC CURS	OR 40				Distinct	PullUp			1
NAME	VORNA	ME	STR	NR	PLZ	ORT	CODE	ADDINFO	
Aaron	Anton		Alt Moabit	96	10559	Berlin	Х	Testperson	
A-J de Groot	Hugo		Dummy	1	10000			Growth Mkt-Sa	pient/CA
Adametz	Hannel	ore	Dummy	1	10510	Berlin		Sekretariat Gesc	hf.
Akin	Nursen		Dummy	1	11020			Beratung RW3	
Alper	Carol		Dummy	1	11530			Pre-Sales-Bosto	n
Andersson	Lisa		Dummy	1	12040			Consulting Basi	s
Aprisnik	Thorste	n	Dummy	1	12550			<b>BA-Industrie</b>	
Asik	Isiltan		Dummy	1	13060			Verrechnungsk.	Tuerkei
	11.1	a al	Dumanaut	1	12570			ALITOMOTIVE	



The WHERE condition specifies columns of index ZZTELE~2.

The result should be sorted according to the primary key.

Using the additional strategy TEMPORARY INDEX CREATED, the primary keys are sorted in a merge list. The optimum cache usage is guaranteed using access to the base data in the order of the primary keys.

Note: During index merge the index is locked therefore it is useful to define a limit.

The maximum size of the merge lists that are generated can be configured using the parameter IndexlistsMergeThreshold (OPTIM\_MAX\_MERGE). When the number of index pages involved is less than or equal to IndexlistsMergeThreshold ( Default 500 Pages) the strategy eg. RANGE CONDITION FOR INDEX is used.

As an alternative the strategy which only works on the base table is used.

An intermediate result set is generated (merge list) but result is not copied.

SELECT * FRO WHERE BETWEEN 20 <i>I</i> ORDER	DM zztele str = 'Wexstr' AND 23 BY name, vornan	AND nr ne, str		SAP
TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT	
ZZTELE	ZZTELE~2	RANGE CONDITION FOR INDEX	2048	
		TEMPORARY INDEX CREATED		
	STR	(USED INDEX COLUMN)		
	NR	(USED INDEX COLUMN)		
JDBC_CURSOR_76		RESULT IS NOT COPIED, COSTVALUE IS	22	
JDBC_CURSOR_76		QUERYREWRITE : APPLIED RULES:		
JDBC_CURSOR_76		DistinctPullUp	1	

TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
ZZTELE	ZZTELE~2	RANGE CONDITION FOR INDEX	2048
		ONLY INDEX ACCESSED	
		MIN/MAX OPTIMIZATION	
	STR	(USED INDEX COLUMN)	
JDBC_CURSOR_19		RESULT IS COPIED , COSTVALUE IS	3

If possible an index is used to compute the min max value of the specified column in the select list.

In this example the multiple index on columns str and nr is used to find the highest number of street Wexstr.

Access on table zztele is not necessary to deliver the result.

<b>Optimizatio</b> "ZZTELE~1" ON z nct (ort) tele	o <b>n - Example</b> ztele ( ort, str )	SA
COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
ZZTELE~1	INDEX SCAN	2273
	ONLY INDEX ACCESSED	
	DISTINCT OPTIMIZATION (P)	
	RESULT IS NOT COPIED, COSTVALUE IS	2273
ORT		
	Optimizatio	Optimization - Example         "ZZTELE~1" ON zztele (ort, str)         oct (ort)         tele         COLUMN_OR_INDEX         STRATEGY         ZZTELE~1         INDEX SCAN         ONLY INDEX ACCESSED         DISTINCT OPTIMIZATION (P)         RESULT IS NOT COPIED, COSTVALUE IS

DISTINCT eliminates Duplicates.

Internally MaxDB creates a temporary b\*tree. The key of this temporary B\* tree is defined on the DISTINCT fields of the SELECT list.

To compute the result each record which is read is copied into this temporary table. When a record with the same distinct value will be inserted twice an error avoids the insert.

This expensive procedure is not necessary if an index exists on the columns of the select list or parts of the select list. If this optimization can be used only the keys of the secondary index will be checked but not the primary key list.

In this example there is no access to the primary table necessary; the DISTINCT can be provided with the index zztele~1by the secondary key.

# NO STRATEGY NOW (ONLY AT EXECUTION TIME)

Strategy will be determined first during execution of the command

AP 2010 /MaxDB 7.8 Internals – Optimizer Introducti

This is valid for queries if the access path will be determined first when they are executed.

SAF

Usually this output is used for queries containing sub-queries or correlated subqueries: strategy will first be determined when interim results become available.



With a subquery you can generate the input values for WHERE condition of the query.

In this example all names of persons who made the master in year 2000 should be listed.

First the subquery will be executed to get the results as input for the IN condition.

The strategy which is used for the ,external' SELECT on ZZTELE can only be chosen during execution. The values of the IN-clause are not known yet.

An intermediate result set is generated. The result set contains key values, in this example the first key column name, ordered by the primary key of the base table. The Optimizer is doing an IN or RANGE STRATEGY FOR KEY COLUMN on table zztele to find those entries which belong to the result with PLZ = 10967.

If a subquery returns values which can be compared with key values, EQUAL CONDITION FOR KEY, IN CONDITION FOR KEY or RANGE CONDITION FOR KEY is used on the base table. The result set is sorted according to primary key values.

If a subquery returns values which can be compared with index values, EQUAL CONDITION FOR Index , IN CONDITION FOR Index or RANGE CONDITION FOR Index is used.

## SUBQUERY – Example (1)

SAP

EXPLAIN SELECT \* FROM zztele WHERE name IN (SELECT name FROM zzmaster WHERE year = 2000)

1
1
1
1



The result set is sorted according to the secondary key sequence. If only values from the index are queried, the Only Index strategy is used.

An intermediate result set is generated.

### **SUBQUERY – Example (2)**



#### SELECT \* FROM zztele WHERE plz IN (SELECT plz FROM zzstadtteil WHERE stadtteil = 'Kreuzberg')

ZZSTADTTEIL     ZZSTADTTEIL~1     EQUAL CONDITION FOR INDEX       STADTTEIL     (USED INDEX COLUMN)       ZZTELE     NO STRATEGY NOW (ONLY AT EXECUTION TIME)       JDBC_CURSOR_47     RESULT IS COPIED , COSTVALUE IS	41
STADTTEIL     (USED INDEX COLUMN)       ZZTELE     NO STRATEGY NOW (ONLY AT EXECUTION TIME)       JDBC_CURSOR_47     RESULT IS COPIED , COSTVALUE IS	41
ZZTELE NO STRATEGY NOW (ONLY AT EXECUTION TIME) JDBC_CURSOR_47 RESULT IS COPIED , COSTVALUE IS	
JDBC_CURSOR_47 RESULT IS COPIED , COSTVALUE IS	
JDBC_CURSOR_47 QUERYREWRITE : APPLIED RULES:	
JDBC_CURSOR_47 DistinctPullUp	1
JDBC_CURSOR_47 DistinctPushDown	1

SAP

© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 48

Nice to know		SAP
SELECT plz FROM zzstadtteil WHER	E stadtteil like ,%berg'	
SELECT * FROM zzmaster WHERE vo	orname = 'Lars'	
SELECT * FROM zztele WHERE name IN ('Aimonsri', 'Hofma AND vorname IN ( 'Jan', 'Walter')	nn', 'Lueck', 'MORONI', 'Reijer')	
SELECT * FROM zzmaster WHERE year + 1 = 2001	SELECT * FROM zzmaster WHERE year = 2001 - 1	
© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 49		

#### Wildcards at the beginning of a column specification

Using wildcards at the beginning of a column specification cannot be optimized. All rows of the table can be part of the result.

Such queries will be processed with table scans and can result in terrible performance.

Solution: Teach your end users not to start the specification with %

#### WHERE qualification specifies only some columns at the end of an index or the primary key

As the first key filed (Name) was not specified, MaxDB is not able to use the primary key of ZZTELE and performs a table scan which can result in terrible performance.

Solution: Teach your end users to specify as many values as possible Create secondary index

Only one IN condition can be optimized, if there are more than one IN qualifications the first is optimized. All followed IN conditions will be processed via range.

**Do not use functions in where column qualification.** This cannot be optimized. Always try to use the function in the value specification.



Nested OR terms are analyzed down to the third level.

If the costs of the strategy search exceed the costs determined for the highest level, the strategy search is discontinued.

An intermediate result set is generated.

Within the SAP environment, similar statements are also generated by SELECTS with RANGES.

Table/View Edit Goto System	Help	的品质	9   🗺 🖬	<b>(</b> )					
Table / view information									
🗿 🤽   💷									
😒 🚓 🚺 🗊 System Configuratic 🕨	Table/View Schema	SAPW	B5			1			
System WB5	Table / View Name	ZZTE	LE			ī			
SAP MaxDB Database Administration	Properties Definition	Indexes	Optimize	r Statistics	Exact	t Sizes	Data Stor. 👔	Data Brows	ser
Current Status								_	
	1 1 🔄 🗟 🛄	Inactive In	dexes 📘	Unused	Indexes		Bad Indexes		
) Dibbs	중 숲 읍 , ▦ ,								
Alerts	Table / Jadex / Column	2	Cort Type	C A		Accossos	Recet Date	Time	Creation D. Time
		F	зотс туре	C h	5	Accesses	Reset Date	Time	creation b Time
Missing Tables and Indexes	CODE	-				27 042	16.02.2010	11.27.45	16.02.2010.11:27:45
FXPLAIN	* CODE	1	ASC		•	27.943	10.03.2010	11.57.45	10.03.2010 11.37.45
SELECT-Editor	▼ 77TELE~1	-	730	• •	Δ	0	16.03.2010	11:37:46	16.03.2010 11:37:46
<ul> <li>Database Files</li> </ul>	· ORT	1	ASC		-		1010012010	1110/110	1010012010 1110/110
<ul> <li>Critical Regions</li> </ul>	• STR	2	ASC						
Database Console	▼ ZZTELE~2			•		83.901	16.03.2010	11:37:44	16.03.2010 11:37:44
Database Trace	• STR	1	ASC						
SQLDBC Trace     SVSINEO Vious	• NR	2	ASC						
Frror Codes	▼ ZZTELE~3			•••		223.696	16.03.2010	11:37:46	16.03.2010 11:37:46
Messages	• PLZ	1	ASC		-				
<ul> <li>Database Objects</li> </ul>	▼ ZZTELE~4		100			3	16.03.2010	11:37:47	16.03.2010 11:37:47
Tables/Views/Synonyms	VORNAME	1	ASC						
<ul> <li>Indexes</li> </ul>									
<ul> <li>Database Procedures</li> </ul>									
Table Sizes									
Administration									
• Tools									
Documentation									

Indexes enable faster access to the rows of a table. The indexes of a table can be determined using the system table INDEXCOLUMNS.

SELECT owner, tablename, indexname, type, columnname,

sort, columnno, datatype, len, createdate

FROM domain.indexcolumns

WHERE owner = <owner>

AND schemaname = <schema>

AND tablename = <table\_name>

ORDER BY owner, tablename, indexname, columnno

You can create an index (also known as secondary key) to speed up the search for database records in a table. In technical terms, indexes are data structures (consisting of one or more inverting lists), which store parts of the data of a table in a separate B\* tree structure. This storage sorts the data according to the inverting key fields that were used. Due to this type of storage, the table data can be accessed faster using the indexed columns than without the relevant index.

For more information about indexes use SAP note 928037 FAQ SAP MaxDB Indexes

Introduction Join (1)	Ş,
SQL joins are used to query data from two or more tables, based on a relationship between certain columns in these tables.	
ANSI Syntax	
SELECT reservation.rno, customer.name, reservation.arrival, reservation.departure FROM hotel.customer JOIN hotel.reservation ON customer.cno = reservation.cno WHERE customer.name = 'Porter' AND ROWNO <= 6 Local predicate Join predicate	
© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 52	

A join is an SQL statement that links multiple tables with each other. A result table is created.

An **inner join** is the most common join operation. Inner join creates a result by combining column values of two tables (A and B) based upon the **join predicate**.

The join predicate is defined in an ON clause and specifies a comparison between two values or lists of values of both tables.

MaxDB handles four types of JOIN: INNER, OUTER (Full, LEFT and RIGHT), UNION

An **outer join** does not require each record in the joined tables to have a matching record. The result contains each record—even if no other matching record exists. We distinguish between left and right outer join.

A **left outer join** returns all the values from an inner join plus all values in the left table that do not match to the right table added by NULL values for the left table.

A **right outer join** returns all the values from the right table and matched values from the left table added by NULL values for the right table.

XPLAIN SELE				0
ROM hotel.cust VHERE custome ND ROWNO <=	CT customer.name tomer JOIN ho er.name = 'Por = 6	ə, reservation.arriv Itel.reservation <b>ON</b> ter''	al, reservation.departure customer.cno = reservation.cno	
SCHEMANAME T	ABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
HOTEL C	USTOMER	FULL_NAME_INDEX	RANGE CONDITION FOR INDEX	1
			ONLY INDEX ACCESSED	
		NAME	(USED INDEX COLUMN)	
HOTEL R	RESERVATION		JOIN VIA KEY RANGE	1
			TABLE TEMPORARY SORTED	
		CNO	(USED COLUMN)	
J	DBC_CURSOR_15		RESULT IS COPIED , COSTVALUE IS	3
J	DBC_CURSOR_15		QUERYREWRITE : APPLIED RULES:	
JC	DBC_CURSOR_15		PushDownPredicates	1

When a join is optimized, first the optimal access strategy for each single table is calculated.

Then the optimizer decides which order of the tables will be processed in the join executation. The calculation of the costs are based on the optimizer statistics.

Outdated optimizer statistics may have an extreme influence on the chosen access strategy and therefore on the runtime of the SQL command.

E.g. After a dataload the statistics are outdated. But only if the relationship of the data (Distinct values) was changed new optimizer statistics are necessary to find the best strategy.



For a JOIN, the optimizer looks for the most suitable access path for each table.

Then the join optimizer decides in which order the tables will be processed and connected with each other. For the join columns, the values are unknown before the execution. Therefore, the join optimizer works with statistical values for columns.

Update Statistics (1)
UPDATE STAT[ISTICS] [ <owner>.]<table_name> [ESTIMATE [SAMPLE <unsigned_integer> <percent,rows>]]</percent,rows></unsigned_integer></table_name></owner>
To determine the best possible access path, in particular for joins, the Optimizer requires statistical information. If such information is not up-to-date, the system may make erroneous strategic decisions.
<b>UPDATE STATISTICS</b> determines values about the size of a table as well as the size and value distribution of indexes.
<b>UPDATE STATISTICS</b> should be executed following large-scale change transactions (INSERT/LOAD, UPDATE, DELETE).
Start using the DBM command sql_updatestat and sql_updatestat_per_systemtable or via the CCMS (transactions DB13, DB21, DBACOCKPIT).
© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 55

For the table itself, Update Statistics only determines data if the current size information is not already in the file directory. This does not apply to tables created with databases of versions < 7.6 and for which no size information could yet be determined in the file directory.

Update Statistics determines statistics data for all columns that are primary key or index columns. It also determines the statistics data for all columns outside of the primary key and the index, if statistics are available. Additionally it determines the statistics data of all entries in system table SYSUPDSTATWANTED.

If the Optimizer discovers tables with outdated statistics data, they are inserted into in the table SYSUPDSTATWANTED. The DBM command sql\_updatestat\_per\_systemtable executes Update Statistics for all tables listed in SYSUPDSTATWANTED.

The DBM command sql\_updatestat executes Update Statistics for all tables in the database.

Update Statistics imports the data for a table from all data volumes in parallel for update statistics computed (not estimate). This makes it very speedy.

As of version 7.6, the sampling procedure in the standard uses a new algorithm for calculating the statistics data. You can determine the algorithm to be used with the parameter UPDATESTAT\_SAMPLE\_ALGO. The new algorithm generates more accurate statistics with fewer records read.

#### The programs "xpu" and "updcol" are no longer available as of version 7.6.

Additional information about Update Statistics: FAQ note 927882

Update Statistics (2)	SAP
ALTER TABLE <table_name> SAMPLE <unsigned_integer> <percent,rows></percent,rows></unsigned_integer></table_name>	
The default value for the number of rows to be included when determini statistics is stored in the database catalog.	ng the
This value can be changed either directly with ALTER TABLE or using t DBACOCKPIT -> Diagnostics-> Database Objects ->Tables/Views/Syne	ransaction onyms

For tables that grow and shrink very quickly, such as spool tables, for example, it is a good idea to set the sampling rate to 0. This prevents Update Statistics from being requested and executed for these tables.

With the following command dbmcli starts an Update Statistics with sampling for all tables of one schema:

sql\_updatestat SAP<SID>.\* estimate

pdate Statis	stics (3)						S
ਟੇ <u>T</u> able/View <u>E</u> dit <u>G</u> oto S <u>v</u> sten	n <u>H</u> elp						
Ø 🔹 🗸 🖉	C 🐼 😪   🗅 M 👪   21 10	li 🎗   💥 🏹	2 🖪				
Table / view information							
តា <u>ខ</u> ា ប							
	Table / Kanna Cabaana	Ca pupe		p			
System WB5	Table / View Name	ZZTELE		,			
SAP MaxDB Database Administra Current Status Performance	Properties Definition I	ndexes Optimizer	Statistics Exact	Sizes Data	Stor. Data Br	owser	
Space	Requested Updates		date (Standard)	🔄 🖬 Up	date (Column Sel	ection)	
Jobs     Alerts		Different Values	Entries (Exact)	Dagos	Dagos (Evact)	Ctatistics	Time
<ul> <li>Diagnostics</li> </ul>	<ul> <li>SAPWB5.ZZTELE</li> </ul>	114199	114199	3200	3212	20.06.2012	16:41:5 +
Missing Tables and Indexes	<ul> <li>Columns</li> </ul>						-
EXPLAIN     SELECT-Editor	ADDINFO     CODE	1033					
Database Files	• NAME	1803					
Critical Regions	• NR	255					
Database Console	• ORT	2					
Database Trace     SOLDRC Trace	• PLZ	48281					
SYSINFO Views	· STR · VORNAME	1297					
Error Codes	✓ Indexes	1207					
<ul> <li>Messages</li> </ul>	ZZTELE~2	513	513	2048	2048		
Database Objects	CODE	3	3	1984	1984		
Indexes	• 22 TELE~1 • 77TELE~3	20001	20001	22/3	22/3		<b>^</b>
Database Procedures		20001	20001	2105	2105		<b>4</b> b
Table Sizes							
Administration							
Tools							
		10110			D. Lawrence (1)		

*Requested Updates* shows if an Update Statistics is requested for this table. It shows the content of system table *SYSUPDSTATWANTED*.

Update Standard executes an Update Statistics table.

You can use Update (Column Statistics) to create column statistics for specified columns.

In the Optimizer Statistics view the column and table statistics are listed.

pa	ate Stat	istics (4)			SA
			mt emt ce		
IERE	tablename	$e = 1 \dots$	1151105		
Show	s the current	statistic values	that will be used by th	e optimizer to dete	rmine the
strate					
Suat	-gy.				
					1
	TABLENAME	INDEXNAME	COLUMNNAME	DISTINCTVALUES	PAGECOUNT
	ZZTELE	?	ADDINFO	1969	?
	ZZTELE	?	CODE	2	?
	ZZTELE	?	NAME	13363	?
	ZZTELE	?	NR	255	?
	ZZTELE	?	ORT	2	?
	ZZTELE	?	PLZ	20001	?
	ZZTELE	?	STR	8	?
	ZZTELE	?	VORNAME	5156	?
	ZZTELE	CODE	?	?	1155
	ZZTELE	ZZTELE~1	?	?	1165
		ZZTELE~3	?	?	1112
	ZZIELE			2	1001
	ZZTELE	ZZTELE~4	?	ſ	1334
	ZZTELE	ZZTELE~4 ZZTELE~2	?	?	1334
	ZZTELE ZZTELE ZZTELE ZZTELE	ZZTELE~4 ZZTELE~2 ?	? ? TABLE STATISTICS	? 114199	1334 1548 1800
	ZZTELE ZZTELE ZZTELE ZZTELE	ZZTELE~4 ZZTELE~2 ?	? ? TABLE STATISTICS	? 114199	1334 1548 1800
	ZZTELE ZZTELE ZZTELE ZZTELE	ZZTELE~4 ZZTELE~2 ?	? ? TABLE STATISTICS	? 114199	1334 1548 1800

The one table Optimizer only uses the statistics data for tables if the counters for size data are not in the file directory.

The join optimizer uses the column statistics created with Update Statistics in the system table *OPTIMIZERSTATISTICS*.

FROM       Files f, roots r         WHERE       f.fileid = r.tableid         AND       r.tablename = ('ZZTELE')         Displays the current counter values in the file directory.         TYPE       TABLENAME       INDEXNAME       ENTRYCOUNT       TREEINDEXSIZE       TREELEAVESSIZE       LOBSIZE         TABLE       ZZTELE       ?       114199       144       14400       0         INDEX       ZZTELE       CODE       2       9240       9240       ?         INDEX       ZZTELE       CODE       2       9240       ?       ?         INDEX       ZZTELE       ZZTELE       2001       8896       ?       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       21515       10672       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       21672       ?       ?	FROM FILES F, FOOTS FWHEREf.fileid = r.tableidANDr.tablename = ('ZZTELE')Displays the current counter values in the file directory.TYPE TABLENAME INDEXNAME ENTRYCOUNT TREEINDEXSIZE TREELEAVESSIZE LOBSIZETABLEZZTELETABLEZZTELEProductCODEProductProductProductProductTABLEZZTELECODE2ProductProductProductProductTABLEZZTELECODE2ProductProductProductProductProductProductTABLEZZTELECODE2ProductProductProductProductProductProductProductProductTABLEZZTELEProductProductTABLEZZTELEProduct	FROM FILES F, FOOTS FWHEREf.fileid = r.tableidANDr.tablename = ('ZZTELE')Displays the current counter values in the file directory.TYPE TABLENAME INDEXNAME ENTRYCOUNT TREEINDEXSIZE TREELEAVESSIZE LOBSIZETYPETABLEINDEXZZTELEProductProductINDEXZZTELECODE2P240P240P320<	FROM       Files F, roots r         WHERE       f.fileid = r.tableid         AND       r.tablename = ('ZZTELE')         Displays the current counter values in the file directory.         TYPE       TABLENAME       INDEXNAME       ENTRYCOUNT       TREEINDEXSIZE       TREELEAVESSIZE       LOBSIZE         TABLE       ZZTELE       ?       114199       144       14400       0         INDEX       ZZTELE       CODE       2       9240       9240       ?         INDEX       ZZTELE       ZZTELE       ZODO1       9320       9320       ?         INDEX       ZZTELE       ZZTELE       ZIELE       216672       10672       ?         INDEX       ZZTELE       ZZTELE       ZIELE       513       12384       ?	LECT f	type, r.	tablename ksize, f.	Director e, r.index treeleave	<b>ry</b> kname, f.en ssize, f.lc	trycount, bbsize	SA
AND       r.tablename = ('ZZTELE')         Displays the current counter values in the file directory.         TYPE       TABLENAME       INDEXNAME       ENTRYCOUNT       TREEINDEXSIZE       TREELEAVESSIZE       LOBSIZE         TABLE       ZZTELE       ?       114199       144       14400       0         INDEX       ZZTELE       CODE       2       9240       9240       ?         INDEX       ZZTELE       ZZTELE       20001       8896       8896       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       210672       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       20011       8896       8896       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       ZZTELE       ZZTELE       ZZTELE       2         INDEX       ZZTELE       ZZTELE       ZZTELE       ZZTELE       2       12384       2	ANDr.tablename = ('ZZTELE')Displays the current counter values in the file directory.TYPETABLENAMETABLEZITELERAND114199INDEXZITELECODE292409240INDEXZITELECODE292409320INDEXZITELEZITELEZITELE~11093209320?INDEXZITELEZITELEZITELE~32000188968896?INDEXZITELEZITELEZITELE~25131238412384?	ANDr.tablename = ('ZZTELE')Displays the current counter values in the file directory.TYPETABLENAMETABLEZZTELE7114199144144001NDEXZZTELE22TELECODE292409240?1NDEXZZTELE22TELEZZTELE1093209320?1NDEXZZTELE22TELEZZTELE~32000188968896?1NDEXZZTELEZZTELEZZTELE~45156106721NDEXZZTELEZZTELEZZTELE~25131238412384?	ANDr.tablename = ('ZZTELE' )Displays the current counter values in the file directory. <td< th=""><th>FROM WHERE</th><th>files f, f.fileid</th><th>roots r = r.tabl</th><th>eid</th><th></th><th></th><th></th></td<>	FROM WHERE	files f, f.fileid	roots r = r.tabl	eid			
Displays the current counter values in the file directory.         TYPE TABLENAME INDEXNAME ENTRYCOUNT TREEINDEXSIZE TREELEAVESSIZE LOBSIZE         TABLE       ZZTELE       ?       114199       144       14400       0         INDEX       ZZTELE       CODE       2       9240       9240       ?         INDEX       ZZTELE       CODE       2       9240       9320       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       20001       8896       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       ZZTELE       20001       10672       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       ZZTELE       2       12384       ?	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELECODE292409320?INDEXZZTELEZZTELE~11093209320?INDEXZZTELEZZTELE~32000188968896?INDEXZZTELEZZTELE~4515610672??INDEXZZTELEZZTELE~251312384??	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELEZZTELE200193209320?INDEXZZTELEZZTELEZZTELE2000188968896?INDEXZZTELEZZTELEZZTELE216672?10672?INDEXZZTELEZZTELEZZTELE~251312384?	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELEZZTELE-v11093209320?INDEXZZTELEZZTELE-v32000188968896?INDEXZZTELEZZTELE-v451561067210672?INDEXZZTELEZZTELE-v251312384?	AND	r.tablen	ame = ('2	ZZTELE')			
Type TABLENAME INDEXNAME ENTRYCOUNT TREEINDEXSIZE TREELEAVESSIZE LOBSIZE         TABLE       ZZTELE       ?       114199       144       14400       0         INDEX       ZZTELE       CODE       2       9240       9240       ?         INDEX       ZZTELE       CODE       2       9240       9320       ?         INDEX       ZZTELE       ZZTELE       20001       8896       8896       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       21516       10672       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       2156       10672       ?	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELECODE292409320?INDEXZZTELEZZTELE~11093209320?INDEXZZTELEZZTELE~32000188968896?INDEXZZTELEZZTELE~451561067210672?INDEXZZTELEZZTELE~25131238412384?	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE29240?INDEXZZTELEZZTELE~11093209320?INDEXZZTELEZZTELE~32000188968896?INDEXZZTELEZZTELE~451561067210672?INDEXZZTELEZZTELE~25131238412384?	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE29240?INDEXZZTELEZZTELE200193209320?INDEXZZTELEZZTELE200188968896?INDEXZZTELEZZTELEZZTELE~3200018896?INDEXZZTELEZZTELEZZTELE~25131238412384?			•	997 Alexandra (1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997			
Displays the current counter values in the file directory.         TYPE       TABLENAME       INDEXNAME       ENTRYCOUNT       TREEINDEXSIZE       TREELEAVESSIZE       LOBSIZE         TABLE       ZZTELE       ?       114199       144       14400       0         INDEX       ZZTELE       CODE       2       9240       9240       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       20001       8896       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       21516       10672       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       2156       10672       ?	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELECODE292409320?INDEXZZTELEZZTELE~11093209320?INDEXZZTELEZZTELE~32000188968896?INDEXZZTELEZZTELE~4515610672?10672INDEXZZTELEZZTELE~251312384?	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELEZZTELE~11093209320?INDEXZZTELEZZTELE~32000188968896?INDEXZZTELEZZTELE~451561067210672?INDEXZZTELEZZTELE~25131238412384?	Displays the current counter values in the file directory. TYPE       TABLENAME       INDEXNAME       ENTRYCOUNT       TREEINDEXSIZE       TREELEAVESSIZE       LOBSIZE         TABLE       ZZTELE       ?       114199       144       14400       0         INDEX       ZZTELE       CODE       2       9240       9240       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       20001       8896       8896       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       10672       ?       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       ?       ?         INDEX       ZZTELE       ZZTELE       ?       ?							
Displays the current counter values in the file directory.         TYPE       TABLENAME       INDEXNAME       ENTRYCOUNT       TREEINDEXSIZE       TREELEAVESSIZE       LOBSIZE         TABLE       ZZTELE       ?       114199       144       14400       0         INDEX       ZZTELE       CODE       2       9240       9240       ?         INDEX       ZZTELE       CODE       2       9240       9320       ?         INDEX       ZZTELE       ZZTELE       20001       8896       ?       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       2156       10672       ?       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       2156       12384       ?	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELECODE292409320?INDEXZZTELEZZTELE~11093209320?INDEXZZTELEZZTELE~32000188968896?INDEXZZTELEZZTELE~451561067210672?INDEXZZTELEZZTELE~25131238412384?	Displays the current counter values in the file directory.TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELEZZTELE~11093209320?INDEXZZTELEZZTELE~32000188968896?INDEXZZTELEZZTELE~451561067210672?INDEXZZTELEZZTELE~25131238412384?	Displays the current counter values in the file directory. TYPE       TABLENAME       INDEXNAME       ENTRYCOUNT       TREEINDEXSIZE       TREELEAVESSIZE       LOBSIZE         TABLE       ZZTELE       ?       114199       144       14400       0         INDEX       ZZTELE       CODE       2       9240       9240       ?         INDEX       ZZTELE       CODE       2       9240       9320       ?         INDEX       ZZTELE       ZZTELE       ZZTELE       20001       8896       ?       ?         INDEX       ZZTELE       ZZTELE       ZITELE       5156       10672       ?       ?         INDEX       ZZTELE       ZZTELE       ZITELE       513       12384       ?       ?							
TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELEZZTELEZZTELE?1093209320?INDEXZZTELEZZTELEZZTELE2000188968896?INDEXZZTELEZZTELEZZTELE21ELE~4515610672?INDEXZZTELEZZTELEZZTELE251312384?	TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELEZZTELEZZTELE29320?INDEXZZTELEZZTELE2000188968896?INDEXZZTELEZZTELEZZTELE10672?INDEXZZTELEZZTELE515610672?INDEXZZTELEZZTELE51312384?	TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELEZZTELE~11093209320?INDEXZZTELEZZTELE~32000188968896?INDEXZZTELEZZTELE~451561067210672?INDEXZZTELEZZTELE~25131238412384?	TYPETABLENAMEINDEXNAMEENTRYCOUNTTREEINDEXSIZETREELEAVESSIZELOBSIZETABLEZZTELE?114199144144000INDEXZZTELECODE292409240?INDEXZZTELEZZTELEZZTELE29320?INDEXZZTELEZZTELEZZTELE293668896?INDEXZZTELEZZTELEZZTELEZZTELE7IINDEXZZTELEZZTELEZZTELE51561067210672?INDEXZZTELEZZTELEZZTELEZITELE251312384?	)ienlave '	the current c	ounter valu	os in tho filo	directory		
TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         ZZTELE~3         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE         513         12384         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         20001         9320         9320         ?           INDEX         ZZTELE         ZZTELE         20001         8896         ?         ?           INDEX         ZZTELE         ZZTELE         5156         10672         ?         ?           INDEX         ZZTELE         ZZTELE         513         12384         ?         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE         513         12384         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         2001         9320         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE         ZITELE~2         513         12384         ?	Jispiays	the current c	ounter value	es in the me	directory.		
TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         ZZTELE~3         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         5154         12384         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         5156         10672         ?         ?           INDEX         ZZTELE         ZZTELE         513         12384         ?         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         ZZTELE         7         10         9320         ?           INDEX         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?         ?							
TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         ZZTELE~3         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         5153         12384         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         2001         9320         ?         ?           INDEX         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE         513         12384         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         20001         8396         8896         ?           INDEX         ZZTELE         ZZTELE         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         2166         10672         10672         ?           INDEX         ZZTELE         ZZTELE         S156         10672         12384         ?           INDEX         ZZTELE         ZZTELE         S13         12384         ?         ?							
TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         ZZTELE~3         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE         5153         12384         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE~1         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE~3         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9320         ?         ?           INDEX         ZZTELE         ZZTELE         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?	TYPE         TABLENAME         INDEXNAME         ENTRYCOUNT         TREEINDEXSIZE         TREELEAVESSIZE         LOBSIZE           TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         CODE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         20001         8996         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         20001         8996         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?         ?							
TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE         2ZTELE         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         2ZTELE         10672         10672         ?           INDEX         ZZTELE         ZZTELE         515         1034         12384         ?	TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~3         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?	TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE~3         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?	TABLE         ZZTELE         ?         114199         144         14400         0           INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         2         9240         9320         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         10672         10672         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         513         12384         ?		TABLENAME	INDEXNAME	ENTRYCOUNT	TREEINDEXSIZE	TREELEAVESSIZE	LOBSIZE
INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         516         10672         10672         ?           INDEX         ZZTELE         ZZTELE         513         12384         ?	INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE         2001         8996         8896         ?           INDEX         ZZTELE         ZZTELE         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?	INDEX         ZZTELE         CODE         2         9240         9240         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         7         10672         ?           INDEX         ZZTELE         ZZTELE         ZZTELE         513         12384         ?	TYPE			114199	144	14400	0
INDEX         ZZTELE         ZZTELE~1         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE~3         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         2         ?	INDEX         ZZTELE         ZZTELE~1         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE~3         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	INDEX         ZZTELE         ZZTELE~1         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE~3         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~3         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	INDEX         ZZTELE         ZZTELE~1         10         9320         9320         ?           INDEX         ZZTELE         ZZTELE~3         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	TYPE	ZZTELE	1				•
INDEX         ZZTELE         ZZTELE~3         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	INDEX         ZZTELE         ZZTELE~3         20001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	INDEX         ZZTELE         ZZTELE~3         2001         8896         8896         ?           INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         ?	INDEX         ZZTELE         ZZTELE         22001         8896         8896         ?           INDEX         ZZTELE         ZZTELE         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE         513         12384         ?	TYPE TABLE INDEX	ZZTELE	CODE	2	9240	9240	?
INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	INDEX         ZZTELE         ZZTELE~4         5156         10672         10672         ?           INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	TYPE TABLE INDEX INDEX	ZZTELE ZZTELE ZZTELE	CODE ZZTELE~1	2 10	9240 9320	9240 9320	? ?
INDEX 77TELE 77TELE~2 513 12384 12384 2	INDEX ZZTELE ZZTELE~2 513 12384 12384 ?	INDEX ZZTELE ZZTELE~2 513 12384 12384 ?	INDEX         ZZTELE         ZZTELE~2         513         12384         12384         ?	TYPE TABLE INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	CODE ZZTELE~1 ZZTELE~3	2 10 20001	9240 9320 8896	9240 9320 8896	? ? ?
				TYPE TABLE INDEX INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	CODE ZZTELE~1 ZZTELE~3 ZZTELE~4	2 10 20001 5156	9240 9320 8896 10672	9240 9320 8896 10672	? ? ? ? ?
				TYPE TABLE INDEX INDEX INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	7 CODE ZZTELE~1 ZZTELE~3 ZZTELE~4 ZZTELE~2	2 10 20001 5156 513	9240 9320 8896 10672 12384	9240 9320 8896 10672 12384	? ? ? ? ?
				TYPE TABLE INDEX INDEX INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	7 CODE ZZTELE~1 ZZTELE~3 ZZTELE~4 ZZTELE~4 ZZTELE~2	2 10 20001 5156 513	9240 9320 8896 10672 12384	9240 9320 8896 10672 12384	? ? ? ? ?
				TYPE TABLE INDEX INDEX INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	7 CODE ZZTELE~1 ZZTELE~3 ZZTELE~4 ZZTELE~4 ZZTELE~2	2 10 20001 5156 513	9240 9320 8896 10672 12384	9240 9320 8896 10672 12384	? ? ? ? ?
				TYPE TABLE INDEX INDEX INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	7 CODE ZZTELE~1 ZZTELE~3 ZZTELE~4 ZZTELE~4 ZZTELE~2	2 10 20001 5156 513	9240 9320 8896 10672 12384	9240 9320 8896 10672 12384	? ? ? ? ?
				TYPE TABLE INDEX INDEX INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	7 CODE ZZTELE~1 ZZTELE~3 ZZTELE~4 ZZTELE~2	2 10 20001 5156 513	9240 9320 8896 10672 12384	9240 9320 8896 10672 12384	? ? ? ? ?
				TYPE TABLE INDEX INDEX INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	7 CODE ZZTELE~1 ZZTELE~3 ZZTELE~4 ZZTELE~2	2 10 20001 5156 513	9240 9320 8896 10672 12384	9240 9320 8896 10672 12384	? ? ? ? ?
				TYPE TABLE INDEX INDEX INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	7 CODE ZZTELE~1 ZZTELE~3 ZZTELE~4 ZZTELE~2	2 10 20001 5156 513	9240 9320 8896 10672 12384	9240 9320 8896 10672 12384	? ? ? ? ?
				TYPE TABLE INDEX INDEX INDEX INDEX INDEX	ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE ZZTELE	7 CODE ZZTELE~1 ZZTELE~3 ZZTELE~4 ZZTELE~2	2 10 20001 5156 513	9240 9320 8896 10672 12384	9240 9320 8896 10672 12384	? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?

For tables that were created with versions < 7.6, the counters for size data in the file directory after upgrade to version 7.5 are not yet available. You can determine the counters with a CHECK DATA in the ADMIN state or with CHECK TABLE WITH SHARE LOCK. CHECK TABLE sets a share lock for the duration of the check.

After the upgrade from versions < 7.6 to versions >= 7.6, all table names are transferred to the table SYSUPDATECOUNTERWANTED. With every restart and in periodic intervals, the database attempts to determine the counters for all remaining tables in SYSUPDATECOUNTERWANTED for the file directory. A share lock is set on a table during processing. Determination of the counters is immediately terminated for a table if the share lock causes a lock collision.

The values for TREENINDEXSIZE, TREELEAVESIZE and LOBSIZE are shown in KB.

For tables, ENTRYCOUNT shows the number of records per table. For indexes, ENTRYCOUNT shows the number of different values for the secondary key.



An executation plan or access path shows how MaxDB accesses the requested data (index access, table scan, key range, key equal, index equal, and so on). An EXPLAIN plan (execution plan) displays the strategy the Optimizer selects to run a special SQL statement. These EXPLAINs are used to analyze long running SQL statements. An EXPLAIN plan can only be displayed for SELECT statements.

In the ABAP-based SAP application server, EXPLAIN is available in transactions ST05, DB50 and DBACockpit (in the command monitor). The SQL editor of the Database Studio can send an EXPLAIN via context menu (right mouse click) to the database. The output is shown in a separate window.

There are additional EXPLAIN statements which are useful for join analysis.

EXPLAIN JOIN and EXPLAIN SEQUENCE are used by the development to find optimizer problems.

Interested people can find additional information can be found in the SCN using the following links: Explain JOIN -> http://wiki.sdn.sap.com/wiki/pages/viewpage.action?pageId=13230&bc=true

EXPLAIN SEQUENCE -> https://wiki.sdn.sap.com/wiki/display/MaxDB/MaxDB+Explain+SEQUENCE

Expla	in (2)			SAP
SCHEMANAME	TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
Schema	Table 1	Names of key or index columns	Name of chosen strategy for this table	Number of pages In system table Optimizerstatistics
Schema	Table 2	Names of key or index columns	Name of chosen strategy for this table	Number of pages in system table Optimizerstatistics
	Result name		RESULT IS (NOT) COPIED, COSTVALUE IS	Estimated costs
			Applied Query Rewrite rules	1
© SAP 2010 /MaxDB 7.8	nternals – Optimizer Introduction	/Page 61		

EXPLAIN shows:

- one block for each table from the SELECT-FROM list
- the order of the strategies reflects the order of execution
- COPIED / NOT COPIED --> Result set is generated/not generated
- "Estimated costs" provides an estimation about the number of read/write accesses
- Applied Query Rewrite rules

Which condition will be evaluated ?	
	S
Join select	
Table1_column = table2_column	
<ul> <li>Condition has to be on the ,Top-AND-Level of the <search condition="">,(Or terms are not relevant)</search></li> </ul>	

Search conditions used by the optimizer to determine the optimal search strategy are:

- Equality conditions
- Range conditions

© SAP 2010 /MaxDB 7.8 Intern

Optimizer Introd

IN conditions

I

The best strategy is chosen by the Optimizer. The basis of decision making is the cost for each evaluated strategy.

The SQL Optimizer also converts conditions under certain circumstances. If a single value is specified in an IN condition multiple times, the condition is converted into an equality condition.



Joins are executed with the Nested Loop method. In doing so for the single join transitions **no result sets are built**. The nested loop join uses one join input as the outer input table and one as the inner input table. The outer loop consumes the outer input table row by row. The inner loop, executed for each outer row, searches for matching rows in the inner input table.

Only the final result is fully created before the first row is delivered. -> this is a advantage for SQL commands with restriction of ROWNO

As of version 7.7 there is no more possibility to choose between **Sorted Merge** or **Nested Loop** by a parameter setting (JOIN\_OPERATOR\_IMPLEMENTATION). There are only marginal disadvantages concerning CPU usage for Nested Loop with the current algorithms. Therewith the Nested Loop can deliver the result faster and with the use of less resources.

The Optimizer starts with that table which related to the total execution plan results in the lowest total costs. You should take care that convenient indexes exist.

In the example the Optimizer starts with a large table customer.

For each hit in customer (outer table) the inner table *reservation* is read. Each hit in reservation is inserted immediately into the final result.

As soon as the number of requested rows (rowno = 6) has been reached the join process stops and the result can be delivered to the application.



Here is an example for nested loop join processed via index strategies.

Join Key Strategies	SAP
SELECT * FROM scantab JOIN jointal AND scantab.B = jointab.Col2	b ON scantab.A = jointab.Coll
Join Strategy	Meaning
JOIN VIA KEY COLUMN	Join table has a single key column key column is part of the join
JOIN VIA MULTIPLE KEY COLUMNS	Join table has multiple key columns all key columns are part of the join
JOIN VIA KEY RANGE	Join table has multiple key columns the first key column is part of the join
JOIN VIA RANGE OF MULTIPLE KEY COLUMNS	Join table has multiple key columns some key columns are part of the join
© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 65	

The analysis and optimization of complex joins is one of the most difficult tasks in the SQL statement analysis.

For the access to the first table have a closer look to the local predicates. Can the primary key be used to access the table or can the access be optimized with an additional index.

For each join with MaxDB it is very important to have good join transition. The number of records read can be reduced by creating convenient indexes for the join transition. During join performance analysis a focus should always be if the best join transition is used.

	ey columr	า (1)	
zstadtteil.plz is	s the <b>sole</b> prin	nary key column	
zstadtteil.ort is	a standard co	olumn	
AND zztele.Ort =	zzstadtteil.Ort		
WHERE zztele.n	ame = 'Mueller'		
WHERE zztele.n	ame = 'Mueller'	RANGE CONDITION FOR KEY	3200
WHERE zztele.n	ame = 'Mueller'	RANGE CONDITION FOR KEY (USED KEY COLUMN)	3200
WHERE zztele.n ZZTELE ZZSTADTTEIL	ame = 'Mueller' NAME PLZ	RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA KEY COLUMN	3200
WHERE zztele.n ZZTELE ZZSTADTTEIL	ame = 'Mueller' NAME PLZ	RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA KEY COLUMN NO TEMPORARY RESULTS CREATED	3200 98
WHERE zztele.n ZZTELE ZZSTADTTEIL IDBC_CURSOR_15	ame = 'Mueller' NAME PLZ	RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA KEY COLUMN NO TEMPORARY RESULTS CREATED RESULT IS COPIED , COSTVALUE IS	3200 98 79
WHERE zztele.n ZZTELE ZZSTADTTEIL IDBC_CURSOR_15 IDBC_CURSOR_15	ame = 'Mueller' NAME PLZ	RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA KEY COLUMN NO TEMPORARY RESULTS CREATED RESULT IS COPIED , COSTVALUE IS QUERYREWRITE : APPLIED RULES:	3200 98 79
WHERE zztele.n ZZTELE ZZSTADTTEIL DBC_CURSOR_15 DBC_CURSOR_15 DBC_CURSOR_15	ame = 'Mueller' NAME PLZ	RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA KEY COLUMN NO TEMPORARY RESULTS CREATED RESULT IS COPIED , COSTVALUE IS QUERYREWRITE : APPLIED RULES: DistinctPullUp	3200 98 79

The join transition from table *zztele* to table *zzstadtteil* is specified via column *PLZ*. Table *zzstadtteil* has a single key on column *PLZ*.

The key of table *zzstadtteil* is qualified in the join predicate. So a *JOIN VIA KEY* strategy can be used. Because table *zzstadtteil* only has a single key column on *plz* the join transition can be done with the strategy *JOIN VIA KEY COLUMN*.

oin via mu	ltiple key co	olumns	S	
ztele.name is t ztele.vorname ztele.str is the	he <b>first</b> primary ke is the <b>second</b> prin <b>last/third</b> primary	y column nary key column key column		
SELECT * FROM AND zztele.vorn WHERE zztele.str	zztele <b>JOIN</b> zzmaste name = zzmaster.vorr = 'Alt Moabit'	er <b>ON</b> zztele.name = zzmaster.name name		
<b>AND</b> zzmaster.Ye	ear = '2000'			
AND zzmaster.Ye	ear = '2000' COLUMN_OR_INDEX	STRATEGY	PAGECOUNT	
AND zzmaster.Ye TABLENAME ZZMASTER	ear = '2000' COLUMN_OR_INDEX	STRATEGY RANGE CONDITION FOR KEY	PAGECOUNT 1	
AND zzmaster.Ye TABLENAME ZZMASTER	ear = '2000' COLUMN_OR_INDEX	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN)	PAGECOUNT 1	
AND zzmaster.Ye TABLENAME ZZMASTER ZZTELE	ear = '2000' COLUMN_OR_INDEX YEAR	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA MULTIPLE KEY COLUMNS	PAGECOUNT 1 3200	
AND zzmaster.Ye TABLENAME ZZMASTER ZZTELE	ear = '2000' COLUMN_OR_INDEX YEAR NAME	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA MULTIPLE KEY COLUMNS (USED KEY COLUMN)	PAGECOUNT 1 3200	
AND zzmaster.Ye TABLENAME ZZMASTER ZZTELE	ear = '2000' COLUMN_OR_INDEX YEAR NAME VORNAME	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA MULTIPLE KEY COLUMNS (USED KEY COLUMN) (USED KEY COLUMN)	PAGECOUNT 1 3200	
AND zzmaster.Ye TABLENAME ZZMASTER ZZTELE	ear = '2000' COLUMN_OR_INDEX YEAR NAME VORNAME STR	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA MULTIPLE KEY COLUMNS (USED KEY COLUMN) (USED KEY COLUMN) (USED KEY COLUMN)	PAGECOUNT 1 3200	
AND zzmaster.Ye TABLENAME ZZMASTER ZZTELE	ear = '2000' COLUMN_OR_INDEX YEAR NAME VORNAME STR	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA MULTIPLE KEY COLUMNS (USED KEY COLUMN) (USED KEY COLUMN) (USED KEY COLUMN) NO TEMPORARY RESULTS CREATED	PAGECOUNT 1 3200	

Remember: zztele key: Name, Vorname, Str

If the key of a joined table exists of more than one column and the complete key is qualified the join strategy is the same as *JOIN VIA KEY COLUMN*. Only the name (*JOIN VIA KEY COLUMN / JOIN VIA MULTIPLE KEY COLUMNS*) differs if the joined table has one or several key columns. This is because of historical reasons.

If the complete multiple key is qualified in the join predicates the strategy is called JOIN VIA MULTIPLE KEY COLUMNS.

<b>zztele.name</b> is t zztele.ort is a s	the <b>first</b> primary key tandard column	column	
SELECT * FROM AND zztele.ort = WHERE zzmaste	<b>/</b> zztele <b>JOIN</b> zzmaste = zzmaster.ort er.Year = '2000'	r ON zztele.name = zzmaster.name	
TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
TABLENAME ZZMASTER	COLUMN_OR_INDEX	STRATEGY RANGE CONDITION FOR KEY	PAGECOUNT
TABLENAME ZZMASTER	COLUMN_OR_INDEX	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN)	PAGECOUNT 1
TABLENAME ZZMASTER ZZTELE	COLUMN_OR_INDEX YEAR NAME	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA KEY RANGE	PAGECOUNT 1 3200
TABLENAME ZZMASTER ZZTELE	COLUMN_OR_INDEX YEAR NAME	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA KEY RANGE NO TEMPORARY RESULTS CREATED	PAGECOUNT 1 3200

If the key of a joined table exists of more than one column and only the first column of the multiple key is qualified the join transition is done via a *KEY RANGE*.

If only the first column of the primary key is qualified via a join predicate the join strategy is called JOIN VIA KEY RANGE.

Join via k	ey range of r	nultiple key columns	SAP
ztele.name is ti ztele.vorname SELECT * FROW AND zztele.vorr	he first primary key is the second prim I zztele JOIN zzmaste name = zzmaster.vorr	y column nary key column er ON zztele.name = zzmaster.name name	
WHERE zzmaste	r. Year = '2000'		
TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
TABLENAME ZZMASTER	COLUMN_OR_INDEX	STRATEGY RANGE CONDITION FOR KEY	PAGECOUNT 1
TABLENAME ZZMASTER	COLUMN_OR_INDEX	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN)	PAGECOUNT 1
TABLENAME ZZMASTER ZZTELE	COLUMN_OR_INDEX	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA RANGE OF MULTIPLE KEY COLUMNS	PAGECOUNT 1 3200
TABLENAME ZZMASTER ZZTELE	COLUMN_OR_INDEX YEAR NAME	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA RANGE OF MULTIPLE KEY COLUMNS (USED KEY COLUMN)	PAGECOUNT 1 3200
TABLENAME ZZMASTER ZZTELE	COLUMN_OR_INDEX YEAR NAME VORNAME	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA RANGE OF MULTIPLE KEY COLUMNS (USED KEY COLUMN) (USED KEY COLUMN)	PAGECOUNT 1 3200
TABLENAME ZZMASTER ZZTELE	COLUMN_OR_INDEX YEAR NAME VORNAME	STRATEGY RANGE CONDITION FOR KEY (USED KEY COLUMN) JOIN VIA RANGE OF MULTIPLE KEY COLUMNS (USED KEY COLUMN) (USED KEY COLUMN) NO TEMPORARY RESULTS CREATED	PAGECOUNT 1 3200

If the key of a joined table exists of more than one column and only a part of the multiple key is qualified the join transition is done via a key range.

If there is more than one key column part of the join predicates but not all primary key columns are qualified then we are talking about the join strategy

JOIN VIA RANGE OF MULTIPLE KEY COLUMNS.

The strategy JOIN VIA RANGE OF MULTIPLE KEY COLUMNS is nearly the same as the strategy JOIN VIA KEY RANGE. The difference is the number of key columnes of the joined table and has historical reasons too.

OWNER	TABLENAME	INDEXNAME	COLUMNNAME
SAPWB5	ZZTELE	ZZTELE~2	STR
SAPWB5	ZZTELE	ZZTELE~2	NR
SAPWB5	ZZTELE	CODE	CODE
SAPWB5	ZZTELE	ZZTELE~1	ORT
SAPWB5	ZZTELE	ZZTELE~1	STR
SAPWB5	ZZTELE	ZZTELE~3	PLZ
SAPWB5	ZZTELE	ZZTELE~4	VORNAME

For the next examples about JOIN VIA INDEX accesses the tables ZZTELE, ZZCODE, ZZMASTER and ZZSTADTTEIL are used.

The slide lists the indexes which exist on these tables.

ZZTELE~3, ZZTELE~4 and CODE are single indexes (secondary keys).

ZZTELE~2 and ZZTELE~1 are multiple indexes (secondary keys).

ZZMASTER and ZZCODE do not have any indexes

Join Index Strategies	SAP			
SELECT * FROM scantab JOIN jointab ON scantab.A = jointab.Coll AND scantab.B = jointab.Col2				
Join Strategy	Meaning			
JOIN VIA INDEXED COLUMN	Join table has a single index column Single Index column is part of the join			
JOIN VIA MULTIPLE INDEXED COLUMNS	Join table has multiple index columns all index columns are part of the join			
© SAP 2010 /MaxDB 7.8 Internals – Optimizer Introduction/Page 71	col1 is the first column of a multiple index col2 is the second column of a multiple index			

During join performance analysis an additional focus should be to check if the best join transition is used and if we can optimize the join transition by creating a new index.

The following slides explain the join strategies via index access.

Join via ind	exed column		SAP
<b>ctele.plz</b> is a <b>sing</b> ctele.ort is a stand	<b>jle</b> index column of dard column	zztele~3	
ELECT * ROM zztele JOIN z ND zzstadtteil.ort =	zzstadtteil <b>ON zzstadt</b> - zztele.ort	teil.plz = zztele.plz	
WHERE zzstadtteil.	stadtteil = 'Kreuzberg	STRATECY	DAGECOUNT
WHERE zzstadtteil.	stadtteil = 'Kreuzberg COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
WHERE zzstadtteil. TABLENAME ZZSTADTTEIL	Stadtteil = 'Kreuzberg'	STRATEGY TABLE SCAN	PAGECOUNT 98
WHERE zzstadtteil. TABLENAME ZZSTADTTEIL ZZTELE	Stadtteil = 'Kreuzberg COLUMN_OR_INDEX ZZTELE~3	STRATEGY TABLE SCAN JOIN VIA INDEXED COLUMN	PAGECOUNT 98 3200
WHERE zzstadtteil. TABLENAME ZZSTADTTEIL ZZTELE	stadtteil = 'Kreuzberg COLUMN_OR_INDEX ZZTELE~3 PLZ	STRATEGY TABLE SCAN JOIN VIA INDEXED COLUMN (USED INDEX COLUMN) NO TEMPORARY RESULTS CREATED	PAGECOUNT 98 3200

In this SQL statement a local predicate is specified (stadtteil) on table zzstadtteil.

The join transition between ZZSTADTTEIL and ZZTELE is specified via column PLZ and column ORT.

Table *ZZTELE* has a single index on column *PLZ*. Column *ORT* is neither part of an index nor part of the primary key.

The index ZZTELE~3 of table ZZTELE is qualified in the join predicate. So a JOIN VIA INDEX strategy can be used. Because index zztele~3 is a single index on column PLZ the join transition can be done with the strategy JOIN VIA INDEXED COLUMN.
zztele.nr is the	second index column	of index ZZTELE~1 nn of index ZZTELE~1	
SELECT * FROM zztele JC AND zztele.ort WHERE zzcode	DIN zzcode ON zztele = zzcode.ort .code = 'Cable TV'	.str = zzcode.str	
TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
		TABLECOM	2776
ZZCODE		TABLE SCAN	2//0
ZZCODE ZZTELE	ZZTELE~1	JOIN VIA MULTIPLE INDEXED COLUMNS	3200
ZZCODE	ZZTELE~1 ORT	JOIN VIA MULTIPLE INDEXED COLUMNS (USED INDEX COLUMN)	3200
ZZCODE	ZZTELE~1 ORT STR	JOIN VIA MULTIPLE INDEXED COLUMNS (USED INDEX COLUMN) (USED INDEX COLUMN)	3200
ZZCODE	ZZTELE~1 ORT STR	JOIN VIA MULTIPLE INDEXED COLUMNS (USED INDEX COLUMN) (USED INDEX COLUMN) NO TEMPORARY RESULTS CREATED	3200
ZZCODE ZZTELE IDBC_CURSOR_14	ZZTELE~1 ORT STR	JOIN VIA MULTIPLE INDEXED COLUMNS (USED INDEX COLUMN) (USED INDEX COLUMN) NO TEMPORARY RESULTS CREATED RESULT IS COPIED , COSTVALUE IS	93483620
ZZCODE ZZTELE JDBC_CURSOR_14 JDBC_CURSOR_14	ZZTELE~1 ORT STR	JOIN VIA MULTIPLE INDEXED COLUMNS (USED INDEX COLUMN) (USED INDEX COLUMN) NO TEMPORARY RESULTS CREATED RESULT IS COPIED, COSTVALUE IS QUERYREWRITE : APPLIED RULES:	3200 93483620

On table ZZTELE there exists a multiple index zztele~2 on columns STR,NR. The join transition qualifies the complete index ZZTELE~2.

For the join transition a strategy called JOIN VIA MULTIPLE INDEXED COLUMNS can be used.

This is same strategy as *JOIN VIA INDEX COLUMN*. The only difference is that we have a multiple index instead of a single index.

the second inde	e first index columi ex column (NR) of i	n of index ZZTELE~2 ndex ZZTELE~2  is not qualified	
SELECT * FROM zztele JOI WHERE zzcode.c	N zzcode <b>ON zztele</b> . code = 'Cable TV'	str = zzcode.str	
TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT
TABLENAME	COLUMN_OR_INDEX	STRATEGY TABLE SCAN	PAGECOUNT
TABLENAME ZZCODE ZZTELE	COLUMN_OR_INDEX	STRATEGY TABLE SCAN JOIN VIA RANGE OF MULTIPLE INDEXED COL.	PAGECOUNT 1 3200
TABLENAME ZZCODE ZZTELE	COLUMN_OR_INDEX ZZTELE~2 STR	STRATEGY TABLE SCAN JOIN VIA RANGE OF MULTIPLE INDEXED COL. (USED INDEX COLUMN)	PAGECOUNT 1 3200
TABLENAME ZZCODE ZZTELE	COLUMN_OR_INDEX ZZTELE~2 STR	STRATEGY TABLE SCAN JOIN VIA RANGE OF MULTIPLE INDEXED COL. (USED INDEX COLUMN) NO TEMPORARY RESULTS CREATED	PAGECOUNT 1 3200

If the index of a joined table exists of more than one column and only a part of the multiple secondary key is qualified the join transition is done via an index range.

If there is more than one index column part of the join predicates but not all secondary key columns are qualified then we are talking about the Join strategy

JOIN VIA RANGE OF MULTIPLE INDEXED COLUMNS.



## The hash join strategy is employed when a join transition to a small table is done and it is probable that a large number of records needs to be read from the small table several times.

In this case it would be faster to import the small table once and generate a temporary hash table. Searching for the keys in a hash table is faster than searching via the B\* tree of the table. The accesses on the hash table need not to be synchronized.

The strategy "TABLE HASHED" identifies the join via a hash table.

### JoinHashMinimalRatio - default 1

The minimal ratio between size of tables joined so far to the size of the next table to be joined which has to be equal or exceeded to use hashing for this next table

### HashJoinSingleTableMemorySize (MAX\_SINGLE\_HASHTABLE\_SIZE)

The maximum table size in KB for which hash joins will be executed. If HashJoinSingleTableMemorySize = 0 then no hash tables will be created during join execution.

### HashJoinTotalMemorySize (MAX\_HASHTABLE\_MEMORY)

As there can be multiple hash joins running at the same time, the amount of memory used for all hashes might become excessive if it is unlimited. This parameter sets the upper limit for the memory provided for all hash joins that are running in parallel. If during join execution a join transition qualifies for a hash join but the overall memory used for all hash joins would be more than HashJoinTotalMemorySize a regular join will be executed instead.

If HashJoinTotalMemorySize = 0 then no hash joins will be executed.

SELECT zzstadtteil.* FROM zztele JOIN zzstadtteil ON zztele.plz = zzstadtteil.plz					
TABLENAME	COLUMN_OR_INDEX	STRATEGY	PAGECOUNT		
ZZTELE	ZZTELE~3	INDEX SCAN	3200		
		ONLY INDEX ACCESSED			
ZZSTADTTEIL	PLZ	JOIN VIA KEY COLUMN	98		
		TABLE HASHED	]		
		NO TEMPORARY RESULTS CREATED			
JDBC_CURSOR_248		RESULT IS COPIED , COSTVALUE IS	3801		

```
Hints

Ints provide the Optimizer with rules that it can use if necessary.

Example:

SELECT *+ORDERED*/ zztele.plz, zzstadtteil.stadtteil

FROM zzstadtteil, zztele

WHERE zztele.plz = zzstadtteil.plz

AND zzstadtteil.stadtteil = 'Moabit'

Hints are supported as of:

• MaxDB Version 7.5

• WebAS ABAP Version 6.20
```

MaxDB supports the several hints, see SAP note 832544 FAQ SAP MaxDB Hints for detailed information.

During join performance analysis the ORDERED Hint can be used to force a special order of table processing.

# Thank you!

