

Session 5

SAP MaxDB: Data Integrity I

Automated and Manual Checks

Thorsten Zielke, SAP AG

Roland Mallmann, SAP AG

26.01.2010

1. Data integrity with SAP MaxDB

- B*Tree structure and Data Cache
- Automated checks
- Checks during Backup, Recovery and 'Check Data'
- 'Check Data' variants
- Best procedures for ensuring data integrity

30 minutes

2. Live Demo

- Perform a Backup
- Recreate an Index
- Run Check Data
- Discover errors

15 minutes

3. Questions & Answers

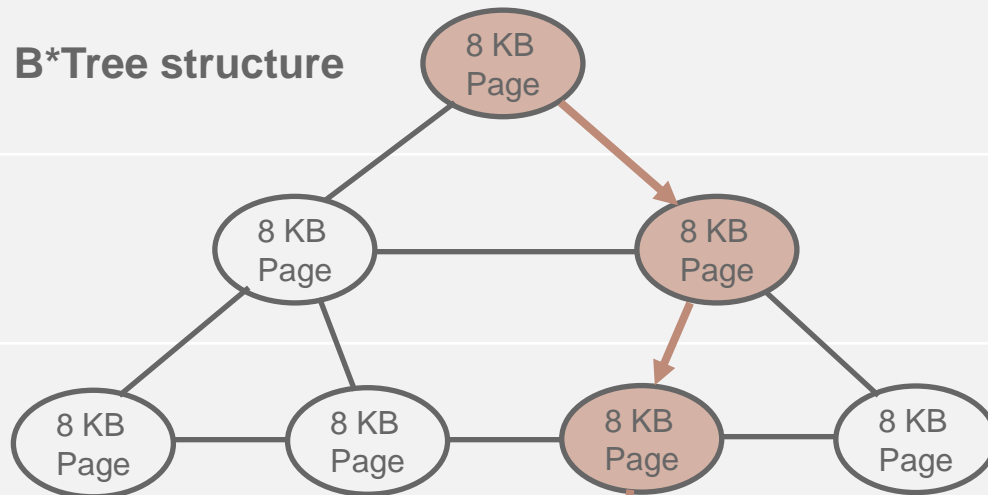
15 minutes

Preface: About SAP MaxDBs B*Tree structure



All SAP MaxDB Database objects are stored in Pages of 8 KB size.

All relational data (tables, indices...) is kept in 'B*Tree'-structures consisting of many pages linked to each other via references (pointers).

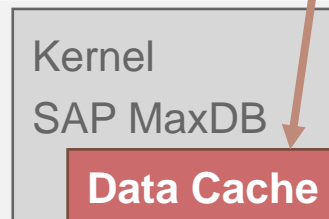


- **Root Page Level:** This is the entry point for each data search (descending via index level to reach the leaf level).

- **Index Page Level:** Collection of pointers to Leaf Page level below (do not confuse with database indices)

- **Leaf Page Level:** This layer stores the application data; Upper levels only supply pointers to reach the matching data.

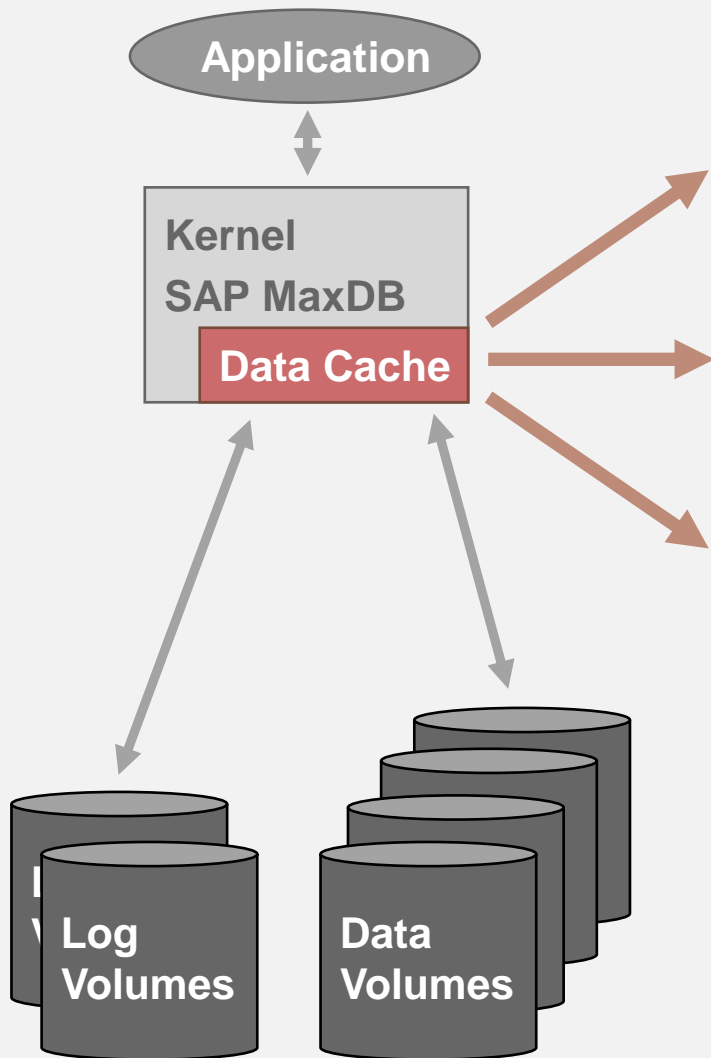
Database Cache



- On data access the relevant pages are requested in the Data Cache. If a page is not found in cache, it first has to be read from the proper volume.

There are many automated checks performed while a page is being accessed in Data Cache.

What is checked ,in Data Cache' during normal database operation



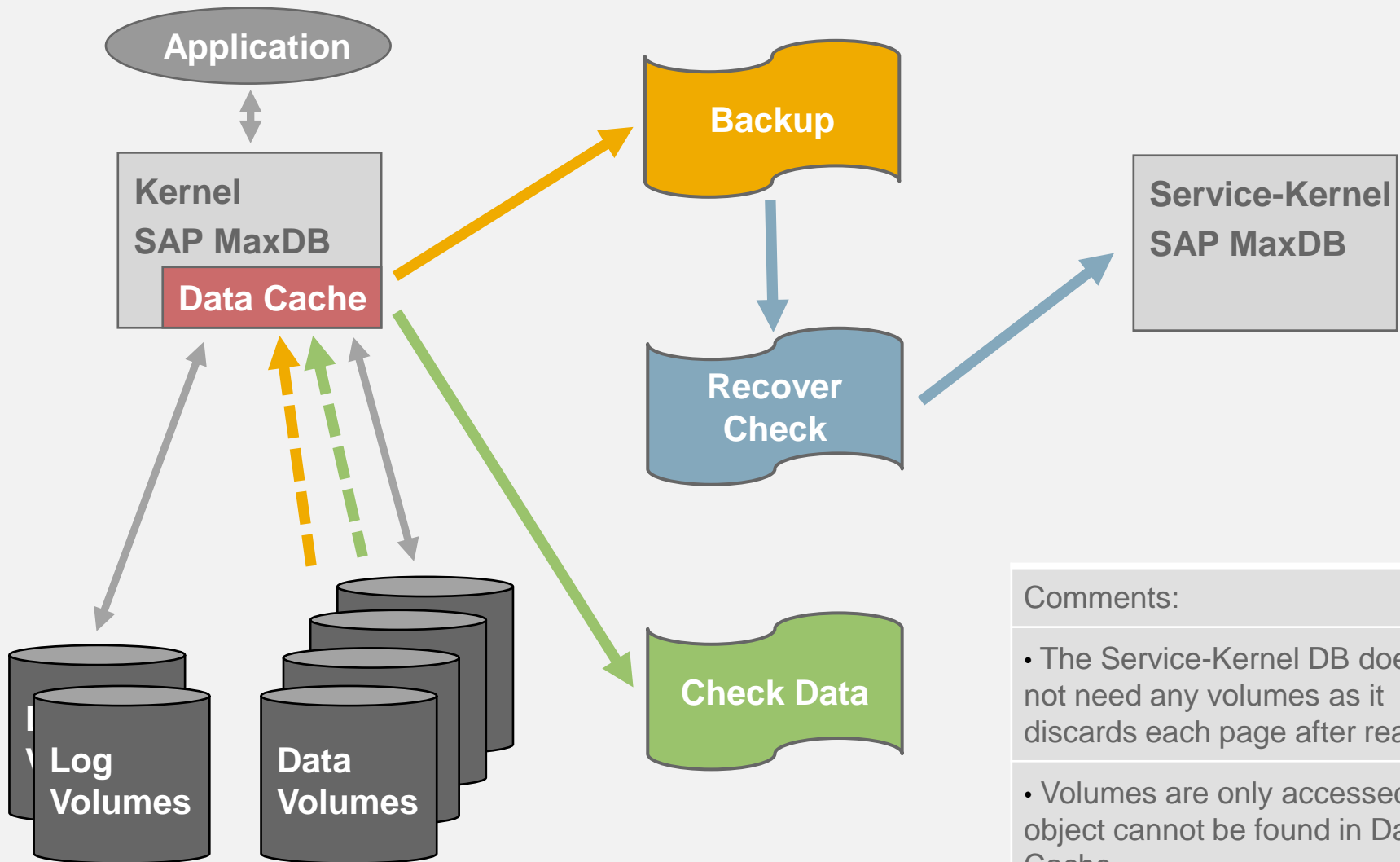
Automated Checks during page access:

- 'Move-error'-Check: Is the data on a page being copied in a valid way?
- Several plausibility checks on page access: 'Page-Number'-Check, 'Page-Type'-Check, 'Header-Trailer'-Check, 'Checksum'-Check, 'Bottom Value'-Check.
- Converter Check: Is the requested page address on data or log volume in a valid area or out of bounds?

- We have a lot of useful automated checks in place, but here we can only check pages that are actually requested during normal business operation and...
 - ...only perform single page level checks, but not verify complete B*Tree objects.

- we need to check all data including B*Tree integrity

Further actions that lead to page level checks



Comments:

- The Service-Kernel DB does not need any volumes as it discards each page after read
- Volumes are only accessed if object cannot be found in Data Cache

Backup vs. ‚Recover Check‘ vs. ‚Check Data‘-Variants



Operation	Performed Integrity Checks	Pros	Cons
Backup	<ul style="list-style-type: none"> • Accesses all pages that are ‘marked for backup’ and writes them to backup medium. • Performs the following checks on page access: Page-Number, Header-Trailer, Page-Type, Checksum 	<ul style="list-style-type: none"> • ‚free‘ check on top of usual backup cycle. • no additional performance impact 	<ul style="list-style-type: none"> • We have to rely on the I/O system to judge if the write call was ok. • Only page level checks, but no B*Tree analysis like following pointers to neighbour pages.
Recover Check	<ul style="list-style-type: none"> • Uses own service database reading each page to /dev/null • Performs the following checks on page access: same as above plus ‘total page count’. 	<ul style="list-style-type: none"> • Verifies backups without needing a full DB instance. • no data cache used 	<ul style="list-style-type: none"> • Only page level checks, but no B*Tree analysis like following pointers to neighbour pages. • Possible I/O impact
Check Data Variants	<ul style="list-style-type: none"> • Offers various checks ranging from complete database structure to single tables/indexes. • Thorough B*Tree checks 	<ul style="list-style-type: none"> • Complete B*Tree consistency check (neighbours, root page) • Extended page level check also verifies ‚key-order‘ on page. 	<ul style="list-style-type: none"> • possible performance impact (depending on check variant: I/O and/or partial table locks).

What makes Check Data so important



Operation	Do not rely on only Backups and/or ,Recover Checks'!	
Backup	<ul style="list-style-type: none"> • A successful backup cannot substitute a complete data check! 	<ul style="list-style-type: none"> • regular check data operations are a requirement to ensure data integrity.
Recover Check	<ul style="list-style-type: none"> • A successful 'recover check' cannot substitute a complete data check! 	<ul style="list-style-type: none"> • regular check data operations are a requirement to ensure data integrity.

What you need Check Data for:

- Not all page corruptions can be detected by backup or recoveries.
- Imagine your database is corrupt and you have never performed a check data – which backup could you trust for recovery? In a worst case scenario, all of your available backups would include the page defect!
- If all available backups include the page defect, you will likely have lost some of your data.

,'Check Data' Variants



Database State	Available Checks
DB_Online	Check Data
	Check Table
	Check Index (since SAP MaxDB 7.8)
	Check Data on Snapshot (work in progress: may or may not come as part of a future SAP MaxDB version e.g. 7.9; also deletes unreferenced page entries)
	Check Catalog (for internal use by SAP MaxDB support)
DB_Admin	Check Data (also deletes unreferenced page entries)

'Check ... [extended]' -> keyword currently obsolete

This option was implemented to test a new 'Ascending-Key-Order Check' which has become part of any default check since SAP MaxDB 7.6.01.00 and 7.7.01.04.

Database State	Check Type		
DB_Online	Check Data	<p>Checks all data objects</p> <ul style="list-style-type: none"> ▪ Tables ▪ Indexes (can be excluded to save runtime) ▪ LOBs (Binary Large Object Files e.g. a ‚picture‘-type file) ▪ History Pages ▪ Internal structures like e.g. Catalog entries, Filedirectory Pages 	
		Usage	...db_execute check data [except index]
		Pros	<ul style="list-style-type: none"> • Verifies all database objects • Faster than ‚db_admin‘-check as data is possibly read from Data Cache (benefit depends on ratio Data Cache size versus total data size)
		Cons	<ul style="list-style-type: none"> • The I/O load will restrict the systems usability and the check sets partial table locks • SAP recommends to limit this to check to off duty hours or when there is scarce user activity • Possibly long runtime, because all data is checked. • Reads from Cache if possible, but may be corrupt on disk.

Database State	Check Type		
DB_Online	Check Table	<p>Checks a single table</p> <ul style="list-style-type: none"> ▪ Tables only ▪ A table's LOB-entries (Binary Large Object Files e.g. a ,picture'-type file) are only verified if additional ,long'-option is set 	
		Usage	...db_execute check table <tablename>[with long check]
		Pros	<ul style="list-style-type: none"> • Fast because affects only a single table (especially if run without 'long'-check as then only the base table is read). • Less interference with normal database operation (partial table locks are set on a single table only). • Less impact on Data Cache content, because fewer pages are to be read and less existing pages flushed out.
		Cons	<ul style="list-style-type: none"> • a partial table lock can still lead to a significant performance impact, if this table gets many updates during normal business operation. • only the 'long'-check option checks the integrity of the whole object; a table with defective longs is still unusable

Database State	Check Type	
DB_Online	Check Index	Checks an index (since SAP MaxDB 7.8) <ul style="list-style-type: none">▪ indexes only
		Usage ...db_execute check index <indexname>
		Pros <ul style="list-style-type: none">• Fast because only affects one single index.• Almost no interference with normal database operation (apart from I/O usage).• Less impact on Data Cache content, because fewer pages are to be read and less existing pages flushed out.
		Cons <ul style="list-style-type: none">• Only checks the index structure but not if it is consistent to the base table content• Can only check one index and not all indexes belonging to the same table

Database State	Check Type		
DB_Online	Check Data on Snapshot	<p>Checks all data using Snapshot technology (work in progress: may or may not come as part of a future SAP MaxDB version e.g. 7.9)</p> <ul style="list-style-type: none"> ▪ Tables & Indexes ▪ LOBs (Binary Large Object Files e.g. a ,picture'-type file) ▪ History Data ▪ Internal structures like e.g. Catalog entries, Filedirectory Pages 	
		Usage	...db_execute check data on snapshot
		Pros	<ul style="list-style-type: none"> • Verifies all objects and removes unreferenced pages • No table locking issues, because working on Snapshot. • Runs with low I/O priority in background to minimize performance impact.
		Cons	<ul style="list-style-type: none"> • Long runtime • Might need considerable free disk space, if existing data is frequently changed while Snapshot is active. • If e.g. a data recovery is to be checked before applying all logs, an 'admin' check data is required, because after DB state 'online' the logs would not fit any more.

Database State	Check Type		
DB_Online	Check Catalog	<p>Checks the database catalog</p> <ul style="list-style-type: none"> ▪ plausibility check for all catalog entries ▪ for SAP internal use, no need to run regularly 	
		Usage	...db_execute check catalog [with update]
		Pros	<ul style="list-style-type: none"> • unless the ‘page level’ catalog check executed as part of a ‘check data’, this check specifically evaluates the catalog content for consistency.
		Cons	<ul style="list-style-type: none"> • Developer knowledge needed for interpreting error output.

Database State	Check Type		
DB_Admin	Check Data	<p>Checks all data objects</p> <ul style="list-style-type: none"> ▪ Tables ▪ Indexes (can be excluded to save runtime) ▪ LOBs (Binary Large Object Files e.g. a ‚picture‘-type file) ▪ History Pages ▪ Internal structures like e.g. Catalog entries, Filedirectory Pages 	
		Usage	...db_execute check data with update
		Pros	<ul style="list-style-type: none"> • Reads all database objects from disk • No interference with other task/users activity • Only this check (and check data on Snapshots) will remove unreferenced pages (e.g. DB is shut down with asynchronous ‚drop table‘ still in progress) freeing up disk space.
		Cons	<ul style="list-style-type: none"> • Requires downtime • Even this check cannot remove ‚long‘ filedirectory entries that are unreferenced to the normal filedirectory

Best Procedures to Ensure Data Integrity



Small Databases	
Best Practice	Check Data (on Weekends)
	Run usual Backup cycle
	Verify Backup with 'Recover Check' via Service DB
Pros	<ul style="list-style-type: none"> • All checks can be run on production. • No dependency between Check Data and Backup
Cons	<ul style="list-style-type: none"> • Backup/Recovery not verified with Check Data, therefore Backup may contain undetected errors. • Possible performance impact on production.

Large Databases	
Best Practice	Run usual backup cycle
	Recover to Test-/Devsystem
	Run Check Data on Test-/Devsystem
Pros	<ul style="list-style-type: none"> • Check Data verifies the whole chain from Production to Backup to Recovery in one single run. • Highest Data safety
Cons	<ul style="list-style-type: none"> • Requires a certain infrastructure size (target system big enough to handle the backup from production) • Target system gets refreshed on a regular basis (might interfere with other tasks that system is used for).

Do's and Dont's:

✓ Do keep your backups in a different location than the source system, at least do not store it on the same server.

✓ Do (at least occasionally) check if you can rebuild a valid system with your backups.

✓ Do mistrust your backup medium, even if it was proven to be ok - it may still have become corrupt later due to external factors.

✓ Do not rely on one single backup set - always have more than one backup generation of data backups, so that you can chose the next good backup, if your latest is faulty.

✓ Do pay attention to keep your log backups going back in time as far as the data backup you want to be able to revert back to in case of disaster. You may need it.

- **If you have SAP OSS access, do not miss these FAQ notes:**
 - SAP note 940420 'FAQ: Database structure check (verify)
 - SAP note 846890 'FAQ: SAP MaxDB Administration'

Addendum: Live Demo



The screenshot shows the SAP MaxDB Database Studio Administration interface for the instance 10.29.14.132:EXPERTDB. The interface includes a menu bar (File, Edit, Navigate, Search, Project, Run, Window, Help), a toolbar, and a left-hand navigation pane with 'Servers' and 'DBM' sections. The main area displays the database's operational state as 'ONLINE' with performance metrics: Data at 64.58%, Log at 0.04%, and Sessions at 2.00%. Below this, there are tabs for Overview, Data Area, Log Area, DBA History, Analyzer, Task Manager, Activities, Caches, Parameters, and Backup. The 'General' tab is active, showing details such as Name, Operational State, Version, Operating System, Instance Type, Installation Path, Independent Program Path, Independent Data Path, and Run Directory Path. A 'Warnings' section lists two messages: '1 bad indexes found' and 'No initial complete data backup found'. Red arrows point from these messages to a text box on the right. The 'Settings' and 'Data Cache' sections are also visible at the bottom of the main area.

Warnings

- 1 bad indexes found. ([Indexes...](#))
- No initial complete data backup found. ([Backup...](#))

Starting DBStudio reveals two warning messages:

- „1 bad index found“
- „No initial complete data backup found“

Addendum: Live Demo

Backup of Database 10.29.14.132:EXPERTDB

Complete Data Backup

Backup successfully completed.

10.29.14.132:EXPERTDB Data: Log: Sessions:

Summary Results

Label	Date	Result	Transferred	Left	Medium	Device/File
✓ <input type="checkbox"/> DAT_000000018	20.01.2010 15:39:39	OK	151,06 MB	0,00 MB	Backup Thorsten Test	c:\sdb\ba...

151,06 MB transferred.

The backup was successfully completed. ←

Close

The ,complete backup' returns ok

Addendum: Live Demo



Refresh Screen: The backup warning has disappeared

The screenshot shows the SAP MaxDB Database Studio Administration window for instance 10.29.14.132:EXPERTDB. The interface includes a menu bar (File, Edit, Navigate, Search, Project, Run, Window, Help), a toolbar, and a left-hand navigation pane with 'Servers' expanded to show the instance. The main area displays the 'Administration' tab with a status bar showing '10.29.14.132:EXPERTDB ONLINE' and resource usage (Data: 64,59%, Log: 0,04%, Sessions: 6,00%). Below the status bar are tabs for Overview, Data Area, Log Area, DBA History, Analyzer, Task Manager, Activities, Caches, Parameters, and Backup. The 'Overview' tab is active, showing a 'General' section with instance details and a 'Warnings' section with one warning: '1 bad indexes found. (Indexes...)'. A 'Settings' section is also visible with 'Automatic Log Backup' and 'Monitoring' both set to 'OFF'. The 'Data Cache' section shows 'Data Cache Size: 7,14 MB' and 'Data Cache Hit Rate: 100,00 %'. A red arrow points from the text box above to the 'Warnings' section.

Section	Property	Value
General	Name:	10.29.14.132:EXPERTDB
	Operational State:	ONLINE since 20.01.2010 15:31:22
	Version:	7.7.06.16
	Operating System:	Windows XP (WIN32)
	Instance Type:	OLTP
Installation Paths	Installation Path:	C:\sdb\expertdb\db
	Independent Program Path:	C:\sdb\programs
	Independent Data Path:	C:\sdb\data
Run Directory Path:	C:\sdb\data\wrk\EXPERTDE	
Warnings	Warning:	1 bad indexes found. (Indexes...)
	Automatic Log Backup:	OFF
Settings	Monitoring:	OFF
	Data Cache Size:	7,14 MB
Data Cache	Data Cache Hit Rate:	100,00 %

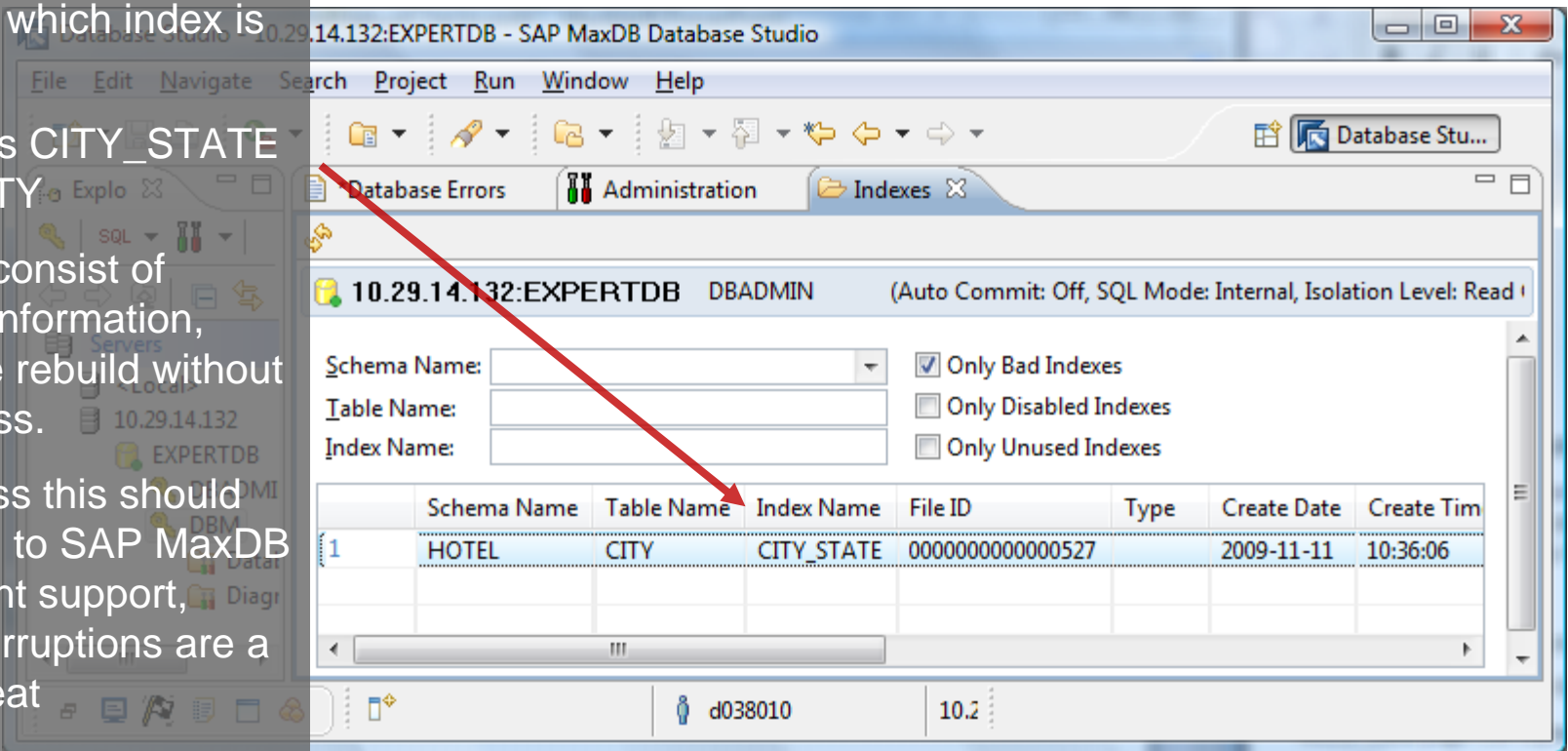
Click on indexes determines which index is affected:

The name is CITY_STATE on table CITY

As indices consist of redundant information, they can be rebuilt without any data loss.

Nevertheless this should be reported to SAP MaxDB development support, because corruptions are a serious threat

At least look for dumped '<pr>.bad' files located in MaxDBs rundirectory and send them to DevSupport before recreating the index



The screenshot shows the SAP MaxDB Database Studio interface. The 'Indexes' tab is active, displaying a table of indexes. A red arrow points to the 'Index Name' column, which contains the value 'CITY_STATE' for the first row. The table also shows the Schema Name 'HOTEL' and Table Name 'CITY'.

	Schema Name	Table Name	Index Name	File ID	Type	Create Date	Create Time
1	HOTEL	CITY	CITY_STATE	00000000000000527		2009-11-11	10:36:06

10.29.14.132:EXPERTDB DBADMIN (Auto Commit: Off, SQL Mode: Internal, Isolation Level: Read)

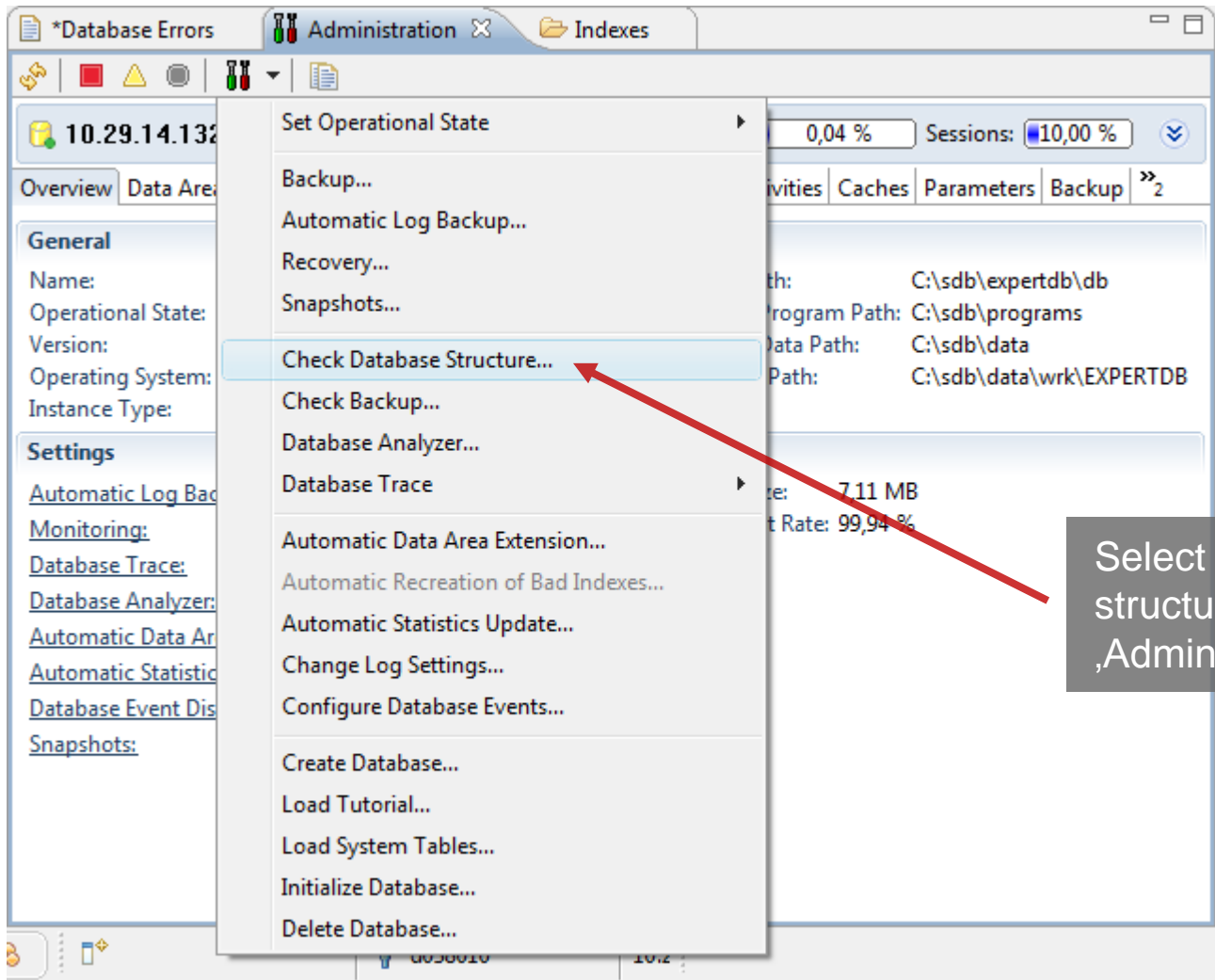
Schema Name:
Table Name:
Index Name:

	Schema Name	Table Name	Index Name	File ID	Type	Create Date	Create Time	Ir
1	HOTEL	CITY	CITY_STATE	0000000000000527		2009-11-11	10:36:06	

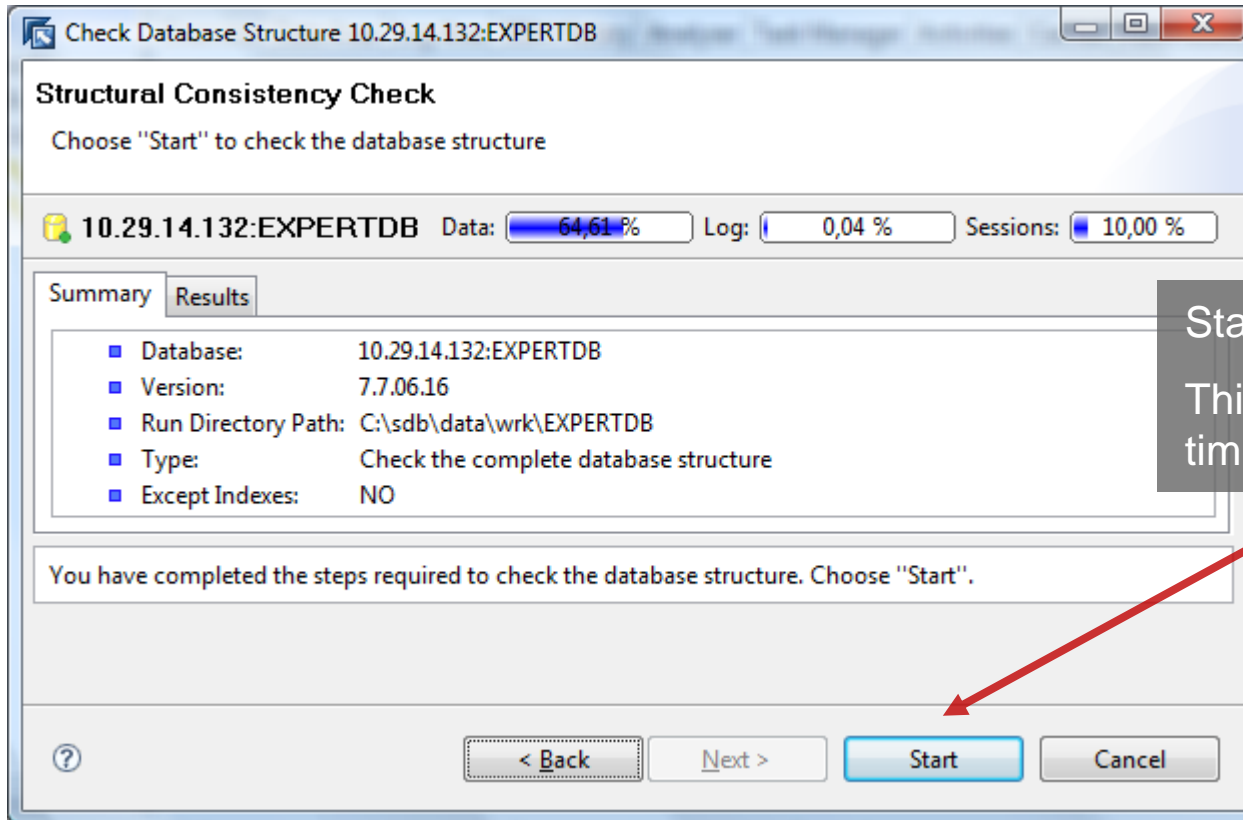
Right Click on index and rebuild via 'recreate index'.

- Disable
- Enable
- Delete
- Recreate Index
- Reset Usage
- Details
- Refresh F5
- Copy Cells
- Copy Rows Ctrl+C
- Columns...

Addendum: Live Demo



Select 'Check Database structure' from the 'Administration' area.



Start the 'check data' job.
This will run for some time

Addendum: Live Demo



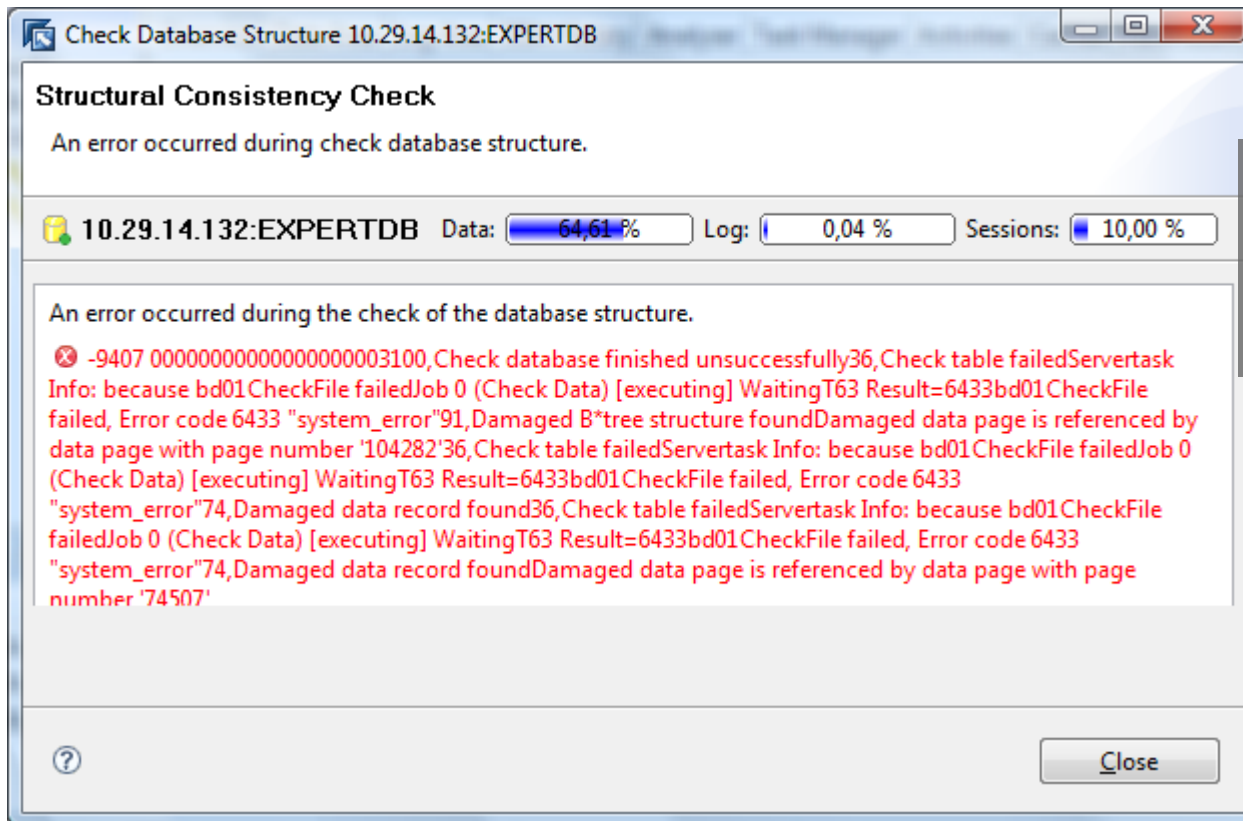
SQL *SQL Editor 2 *Database Messages X

10.29.14.132:EXPERTDB ONLINE Data: 64,69 % Log: 0,04 % Sessions: 10,00 %

Show Line Numbers Search from selection check database Case Sensitive

```
Connect req. (EXPERTDB, T63, connection obj. 0x7F9F6870, Node:'BERN00176467A.dhcp.ber.sap.corp', PID: 3308)
Start check database
Check data on database object failed,KNL_BASE_ERROR=system_error,ROOT=74507,_FILE=vbd38.cpp,_LINE=353
DESCRIPTION:
An error occurred while checking the structure of the database object with Root '74507'.
Check data on database object failed,KNL_BASE_ERROR=system_error,ROOT=104280,_FILE=vbd38.cpp,_LINE=353
DESCRIPTION:
An error occurred while checking the structure of the database object with Root '104280'.
Check database progress report: 1931 of about 19313 pages checked
Connect req. (EXPERTDB, T61, connection obj. 0x7F9F6960, Node:'BERN00176467A.dhcp.ber.sap.corp', PID: 1112)
Connection released (EXPERTDB, T61, connection obj. 7F9F6960)
Connect req. (EXPERTDB, T61, connection obj. 0x7F9F6960, Node:'BERN00176467A.dhcp.ber.sap.corp', PID: 1112)
Connection released (EXPERTDB, T61, connection obj. 7F9F6960)
Check database progress report: 3862 of about 19313 pages checked
Check database progress report: 5793 of about 19313 pages checked
Check database progress report: 7724 of about 19313 pages checked
Check data on database object failed,KNL_BASE_ERROR=system_error,ROOT=104282,_FILE=vbd38.cpp,_LINE=353
DESCRIPTION:
An error occurred while checking the structure of the database object with Root '104282'.
Check database progress report: 9655 of about 19313 pages checked
Start LOB checking
Check database progress report: 11586 of about 19313 pages checked
Check database progress report: 13517 of about 19313 pages checked
```

Have a look into the 'Database Messages' file to determine the 'check data' progress



,check data' has failed although the complete data backup did not return any error!

select * from roots where root = 74507 -> CITY

Determine the affected objects

Error as logged in the 'Database Errors' file (summary version):

An error occurred while checking the table structure with the FileID '00000000000002D8' or Root '104280'.

bd01CheckFile failed, Error code 6433 "system_error"

Damaged data record found,DETECTEDCORRUPTION=invalid record length resp. bottom value,PERSISTENT_TYPE=perm,FILETYPE=table,PAGENO=104280,_FILE=vbd31.cpp,_LINE=2738

DESCRIPTION:

While checking the data page with pagenumber '104280' a serious flaw was detected in at least one data record. The faulty database object has type 'table' and persistence type 'perm'.

This kind of error can be lead back to problems outside of the database software, for example the IO system and must not be ignored! The faulty data page has been written into the run directory of the database for a possible necessary detailed analysis by the development support.

-> Effect on table data: Update/Delete/Insert might corrupt data, because of incorrect record length value

```
select * from roots where root = 74507 -> RESERVATION
```

Determine the affected objects

Error as logged in the 'Database Errors' file (summary version):

An error occurred while checking the table structure with the FileID '0000000000002DF' or Root '74507'.

```
bd01CheckFile failed, Error code 6433 "system_error"
```

```
Damaged data record found, RECORD_POSITION=129,RECORD_INDEX=1,DETECTEDCORRUPTION=invalid key order,PERSISTENT_TYPE=perm,FILETYPE=table,PAGENO=105811,_FILE=vbd31.cpp,_LINE=2705
```

DESCRIPTION:

While checking the data page with pagenumber '105811' a serious flaw was detected in at least one data record. The faulty database object has type 'table' and persistence type 'perm'.

This kind of error can be lead back to problems outside of the database software, for example the IO system and must not be ignored! The faulty data page has been written into the run directory of the database for a possible necessary detailed analysis by the development support.

```
Damaged data page is referenced by data page with page number '74507',RECORD_POSITION=81,RECORD_INDEX=0,PAGENO=105811
```

DESCRIPTION:

The damaged data page with page number '105811' ("child") is referenced by the data page with the page number '74507' ("parent").

For a possibly needed detailed analysis from development support the parent data page was written to the run directory of the database.

-> Effect on table data: the first two table rows are now inaccessible

Conclusion:

The damaged index can be easily recreated.

The damaged tables can be fixed by either deleting parts of the table causing data loss or by performing a full recovery..

Best strategy: Perform a recovery using a backup taken **before** the last successful 'check data' run.