

## DB2 10 for z/OS – Im Einsatz, wo andere längst aufgeben...



© 2010 IBM Corporation

### Disclaimer and Trademarks

Information contained in this material has not been submitted to any formal IBM review and is distributed on "**as is**" basis without any warranty either expressed or implied. The use of this information is a customer responsibility.

IBM MAY HAVE PATENTS OR PENDING PATENT APPLICATIONS COVERING SUBJECT MATTER IN THIS DOCUMENT. THE FURNISHING OF THIS DOCUMENT DOES NOT IMPLY GIVING LICENSE TO THESE PATENTS.

TRADEMARKS: THE FOLLOWING TERMS ARE TRADEMARKS OR ® REGISTERED TRADEMARKS OF THE IBM CORPORATION IN THE UNITED STATES AND/OR OTHER COUNTRIES:  
AIX, CICS, DataPower, DataStage, DB2, DB2 Connect, DB2 Extenders, developerWorks, Distributed Relational Database Architecture, DRDA, Enterprise Storage Server, ESCON, FICON, FlashCopy, GDPS, HyperSwap, IBM, IMS, Information Agenda, iSeries, Language Environment, MQSeries, OMEGAMON, OmniFind, Optim, Passport Advantage, Parallel Sysplex, POWER7, ProductPac, PR/SM, pSeries, pureXML, QMF, QualityStage, Query Management Facility, QuickPlace, Quickr, RACF, Rational, Redbooks, RMF, ServicePac, solidDB, Sysplex Timer, System i, System p, SystemPac, System Storage, System x, System z, System z9, System z10, Tivoli, VTAM, xSeries, WebSphere, z9, z10, z/Architecture, zEnterprise, z/OS, z/VM, zSeries.

"Other company, product or service names may be trademarks or service marks of others"  
For additional information see [ibm.com/legal/copytrade.phtml](http://ibm.com/legal/copytrade.phtml)

## Client Technical Professionals DB2 for z/OS



Christian Daser  
[Christian.Daser@de.ibm.com](mailto:Christian.Daser@de.ibm.com)



Peter Hartmann  
[PETERHAR@de.ibm.com](mailto:PETERHAR@de.ibm.com)



Georg Kistenberger  
[kistenberger@de.ibm.com](mailto:kistenberger@de.ibm.com)

## Agenda

- ▶ 09:00 Einführung
- 09:15 Migration & Vorteile nach der Migration
- 10:15 Neue Funktionen zur Qualitätssicherung
- 10:45 Pause
- 11:00 Anwendungsoptimierung
- 11:35 Neue Anforderungen an IT-Sicherheit
- 12:15 Mittagspause
- 13:00 DB2 auf Zeitreise
- 14:00 Pause
- 14:15 Erweiterungen von OLTP bis Warehousing
- 15:45 Abschluss
- 16:00 Veranstaltungsende

## Performance & CPU Verbrauch

### Savings ... right out of the box

IBM DB2 10 for z/OS delivers faster queries and reduced cost with optimized technology

Simple

Cuts CPU

Innovative

Proven

Secure



The most dramatic achievement of DB2 10 comes from **reduced CPU usage**. Compared to previous versions, most customers can achieve out-of-the-box CPU savings of **up to ten percent** for traditional workloads, and **up to 20 percent** for **some specific<sup>1</sup> workloads**.

<sup>1</sup> ... Actual CPU reduction will, of course, vary depending on the specific customer workload mix...

Current expectation is between 0% and 10% CPU savings

## DB2 V10: Für jeden was dabei!

Systemprogrammierer



Datenbankadministrator

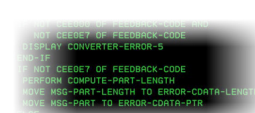


Data Warehousing

ISV Anwendung



OLTP Anwendung



Endbenutzer



Web Anwendung



Anwendungsentwickler



DB2 for z/OS



## DB2 V10: Migration

Systemprogrammierer

Datenbankadministrator



Data Warehousing

- Skip Level Migration
- Catalog Restructure

ISV Anwendung

OLTP Anwendung



Endbenutzer

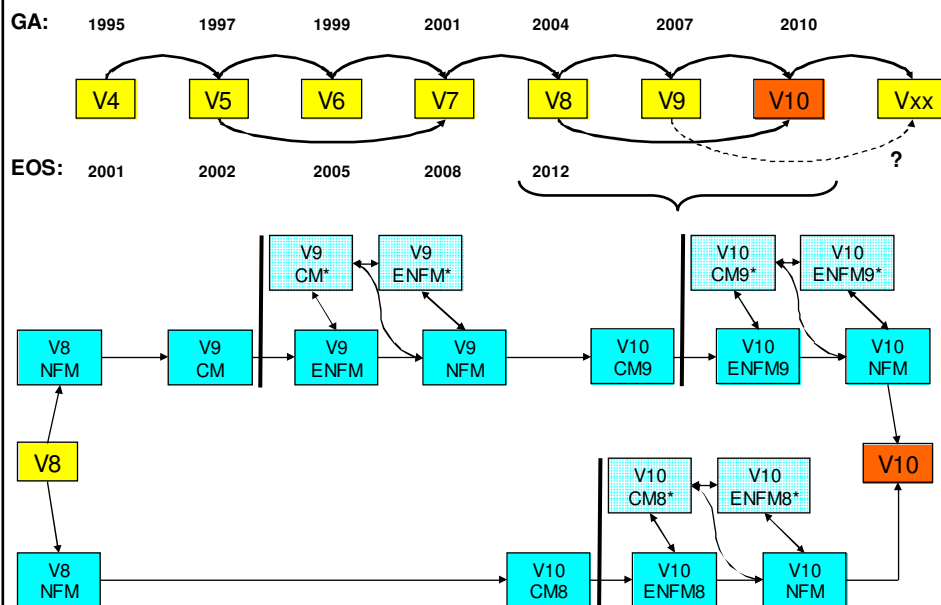


Anwendungsentwickler

Web Anwendung

8

© 2010 IBM Corporation



9

© 2010 IBM Corporation

## Einige Anmerkungen

- Ein V8 System, das nach V10 CM migriert wurde, kann nur einen Fallback nach V8 machen.
- Ein V8 System, das nach V10 CM migriert wurde, kann nach einem Fallback nicht nach V9 migriert werden.
- Ein V8 System, das nach V10 migriert wird, kann neue V9 Funktionen erst in V10 NFM nutzen.
- Coexistence im Data Sharing geht wieder, aber
  - Migration von V8 nach V10: Kein V9 Member in der Gruppe erlaubt
  - Migration von V9 nach V10: Kein V8 Member in der Gruppe erlaubt

## Einige Anforderungen für Migration

- Ab z890, z990 und aufwärts (z9, z10, z196)
  - Aber nicht z800 oder z900
- z/OS V1R10
  - SMS, RACF, LE/370, Unicode
- IRLM V2R3
  - Wird mit DB2 V10 ausgeliefert
- DB2 Connect 9 Fixpack 1 ist notwendig
  - 9.7 Fixpack 3a ist empfohlen wegen neuer Funktionen
- IMS V10, CICS V3.1, Websphere V6
- Optim (Data Studio, pureQuery, ...) ab 2.2
- ...

## Zu beachten: Vor der Migration

- **Installation and Migration Guide:**
  - <http://www-01.ibm.com/support/docview.wss?uid=swg27019288#manuals>
- **Ausgewählte Themen:**
  - Kein Private Protokoll mehr.
  - Explain Tables müssen in V10 Format in Unicode sein.
  - Kein DBRM mehr im Plan.
  - DSNHDECP Newfun=V10/V9/V8.
  - Reorg None für LOBs wird nicht mehr unterstützt.
  - Release Bind Option wird bei DRDA Zugriff unterstützt.
  - zPARMs
    - Sehr viele Neuerungen (FlashCopy, Security, ...)
    - Sehr viele Änderungen (PARTKEYU, CHKTYPE, ...)

## PTF Stand für Migration

- **Toleration Apar als Code Basis vom Startpunkt**
  - PK56922
- **Early Code**
  - Entweder sofort V10 Early Code
  - Oder V8/V9 PK87280 (superseeds PK61766)
- **Information Apars**
  - II14474: V8 nach V10
  - II14477: V9 nach V10
  - Bitte auch auf die entsprechenden Hinweise bezüglich z/OS achten, z.B.
    - 1MB pagesize (LFAREA Ausnutzung im DB2)

## Schritte der Migration

- Vorherige Prüfung auf Migrationsprobleme
  - Job DSNTIJPM
- Migration nach CM (=Conversion Mode)
  - Job DSNTIJTC (Catmaint)
- Migration nach NFM (=New Function Mode)
  - Job DSNTIJEN (Catenfm)
- Set-up der DB2 ausgelieferten Stored Procedures
  - Job DSNTIJRW

## Migration nach Conversion Mode: Catmaint

```

DSNU000I  279 11:27:20.69 DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = RELODCAT
DSNU1044I  279 11:27:20.76 DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNU050I  279 11:27:20.96 DSNUGUTC - CATMAINT UPDATE
DSNU750I  279 11:27:22.66 DSNUECMO - CATMAINT UPDATE PHASE 1 STARTED
DSNU777I  279 11:27:22.66 DSNUECMO - CATMAINT UPDATE STATUS -
        VERIFYING CATALOG IS AT CORRECT LEVEL FOR MIGRATION.
DSNU777I  279 11:27:22.73 DSNUECMO - CATMAINT UPDATE STATUS -
        BEGINNING MIGRATION SQL PROCESSING PHASE.
DSNU777I  279 11:31:39.36 DSNUEXUP - CATMAINT UPDATE STATUS -
        BEGINNING ADDITIONAL CATALOG UPDATES PROCESSING.
DSNU777I  279 11:31:39.40 DSNUEXUP - CATMAINT UPDATE STATUS -
        BEGINNING SYSCOPY TABLE SPACE MIGRATION PROCESSING.
DSNU777I  279 11:31:39.40 DSNUECMO - CATMAINT UPDATE STATUS -
        UPDATING DIRECTORY WITH NEW RELEASE MARKER.
DSNU752I  279 11:31:39.44 DSNUECMO - CATMAINT UPDATE PHASE 1 COMPLETED
DSNU010I  279 11:31:39.64 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
  
```

## Migration nach New Function Mode: Catenfm

- Umbau sehr vieler Catalog/Directory Tablespaces
- Warum ?
  - Contention auf Catalog/Directory durch parallel DDL/Bind usw.
  - A002 muß selber angelegt werden
  - 64GB Grenze

## Migration nach New Function Mode: Catenfm

```

DSNU050I   280 10:22:39.37 DSNUGUTC - REORG TABLESPACE DSND01.DBD01 SHRLEVEL REFERENCE LOG NO COPYDDN(SYSCOPY)
CONVERTV10 RETRY 255 TIMEOUT TERM RETRY_DELAY 1 DRAIN_WAIT 1 SORTDATA SORTDEVT SYSDA SORTNUM 20
DSNU351I   :DBC2 280 10:22:39.37 DSNURFIT - SORTDATA OPTION IS NOT APPLICABLE FOR DB2 CATALOG OR DIRECTORY OBJECT. THE
OPTION IS IGNORED.
...
DSNU1039I  280 10:22:40.87 DSNUGDYN - DATASET ALLOCATED. TEMPLATE=SYSCOPY
          DDNAME=SYS00007
          DSN=CDB2IC.DSND01.DBD01.D10280.T0822.F1P000
DSNU252I   280 10:23:38.77 DSNUECM0 - UNLOAD PHASE STATISTICS - NUMBER OF RECORDS UNLOADED=24951 FOR TABLESPACE
DSND01.DBD01
DSNU304I   280 10:23:48.61 DSNUECML - (RE)LOAD PHASE STATISTICS - NUMBER OF RECORDS=4084 FOR TABLE DBDR
DSNU304I   280 10:23:48.61 DSNUECML - (RE)LOAD PHASE STATISTICS - NUMBER OF RECORDS=0 FOR TABLE DBDS
DSNU304I   280 10:23:48.61 DSNUECML - (RE)LOAD PHASE STATISTICS - NUMBER OF RECORDS=20867 FOR TABLE CONSOLIDATION
DSNU302I   280 10:23:48.63 DSNURCAT - (RE)LOAD PHASE STATISTICS - NUMBER OF INPUT RECORDS PROCESSED=24951
DSNU300I   280 10:23:48.63 DSNURCAT - (RE)LOAD PHASE COMPLETE, ELAPSED TIME=00:00:09
DSNU387I   280 10:23:49.68 DSNURSWT - SWITCH PHASE COMPLETE, ELAPSED TIME = 00:00:01
DSNU428I   280 10:23:49.69 DSNURSWT - DB2 IMAGE COPY SUCCESSFUL FOR TABLESPACE DSND01.DBD01
DSNU1166I  :DBC2 280 10:23:49.84 DSNURSWD - SOME PARTITION STATISTICS MAY HAVE BECOME OBSOLETE ON DSND01.DBD01
DSNU300I   280 10:23:49.89 DSNURSID - COPY PROCESSED FOR TABLESPACE DSND01.DBD01
          NUMBER OF PAGES=0
          AVERAGE PERCENT FREE SPACE PER PAGE = 1.00
          PERCENT OF CHANGED PAGES = 0.00
          ELAPSED TIME=00:00:03
DSNU010I   280 10:23:49.81 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=4

```

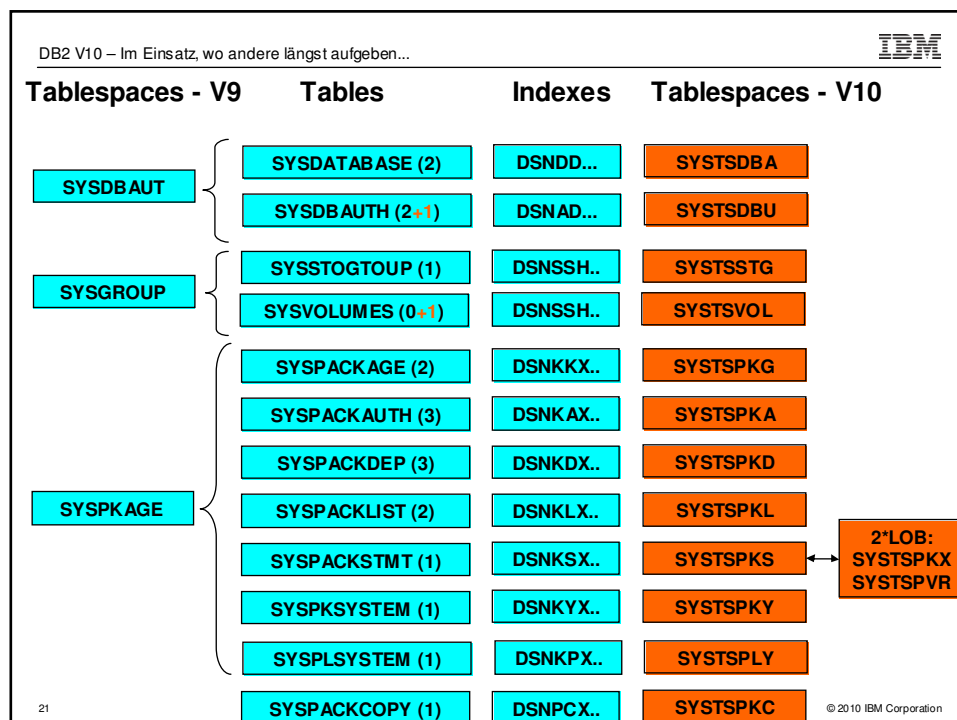
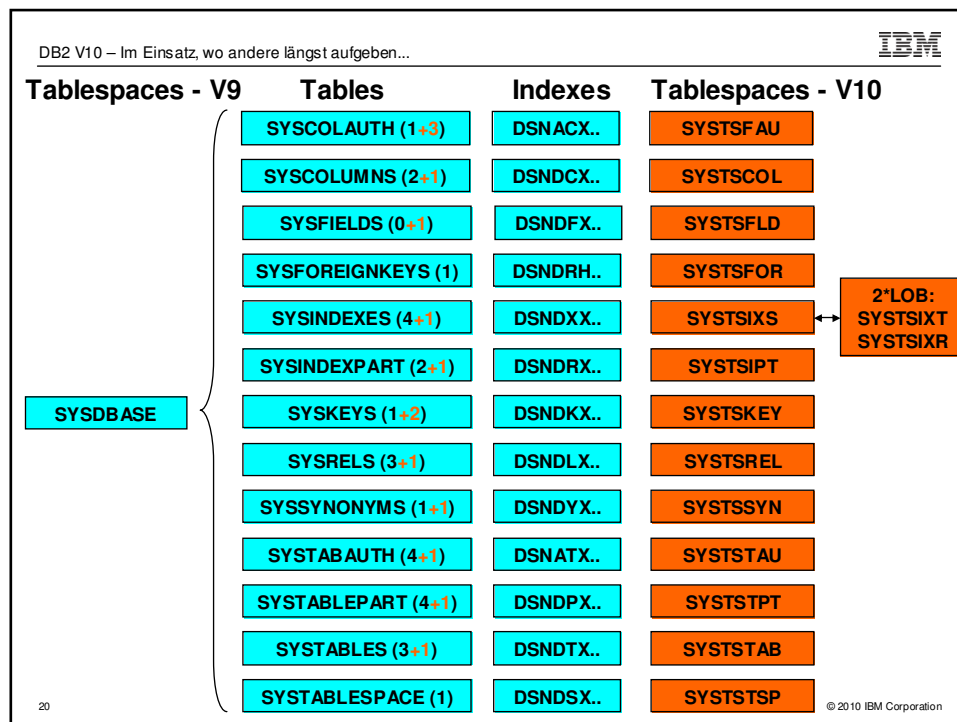


## Catalog Änderungen in V10

- Unveränderte Tablespaces / unveränderte Tables
  - SYSALTER
  - SYSCONTX
  - SYSCOPY: Weiterhin als einziger Tablespace in EBCDIC
  - SYSDDF
  - SYSEBCDC
  - SYSGPAUT
  - SYSGRTNS
  - SYSHIST
  - SYSJAVA
  - SYSROLES
  - SYSRTSTS
  - SYSSEQ
  - SYSSEQ2
  - SYSSTATS
  - SYSSTR
  - SYSTARG
  - SYSUSER
  - SYSXML

## Catalog Änderungen in V10: Konvertierung

- Bisherige Tablespaces
  - SYSDBASE
  - SYSDBAUT
  - SYSGROUP
  - SYSOBJ
  - SYSPKAGE
  - SYSPLAN
  - SYSVIEWS
- Neue Tablespaces / unveränderte Tables (~44)
  - SYSTS...

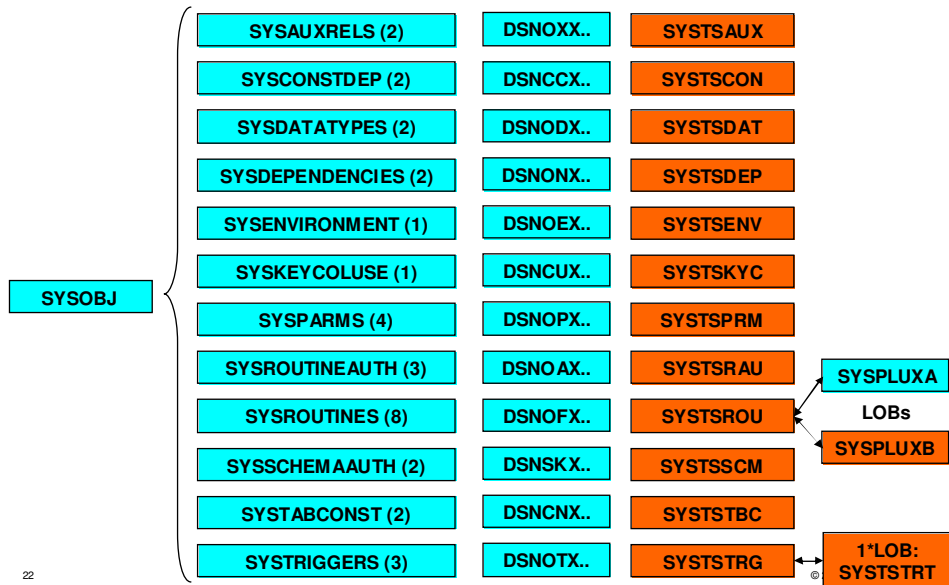


## Tablespaces - V9

## Tables

## Indexes

## Tablespaces - V10

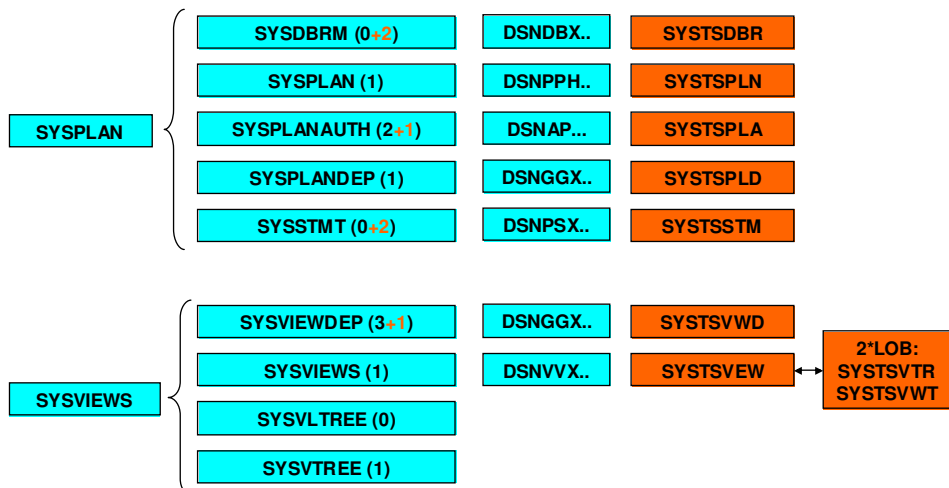


## Tablespaces - V9

## Tables

## Indexes

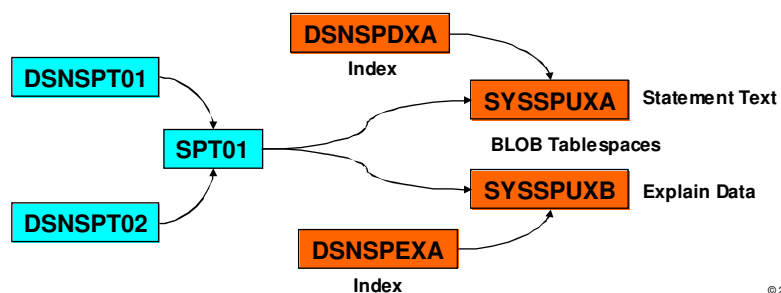
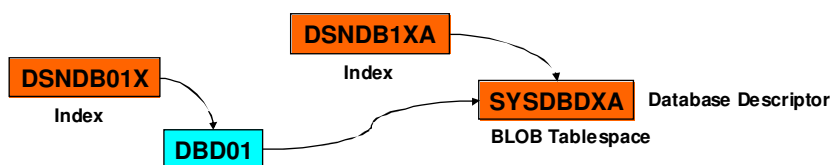
## Tablespaces - V10



## Catalog Änderungen in V10

- Neue Tablespaces / neue Tables (~15)
  - Autostats (e.g. SYSTSATS, SYSTSATW)
  - Runstats Profiles (e.g. SYSTSTPF)
  - Pending (=deferred) Alter (e.g. SYSTSPDO, SYSTSPEN)
  - Access Path Stability (e.g. SYSTSQRY, SYSTSQRP)
  - XML (e.g. SYSTSXTM, SYSTSXTS)
  - Unicode (e.g. SYSTSUNI, SYSTSASC)

## Directory (DSNDB01) Änderungen – V10



## Allgemeine Änderungen im Catalog/Directory

- Keine Links oder Hashes
- SMS managed (schon im CM)
  - Extended addressability in Dataclas
- Row Level Locking
  - Ausnahme: SYSSEQ, SYSEBCDC
  - Locksize kann auf Catalog mit Alter geändert werden
- Partition by Growth
  - Neue Tablespaces sind PBG
  - Ausnahme: Neue XML spaces

## DB2 V10: Vorteile nach der Migration

Systemprogrammierer



Datenbankadministrator

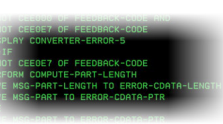


Data Warehousing

ISV Anwendung



OLTP Anwendung



Endbenutzer



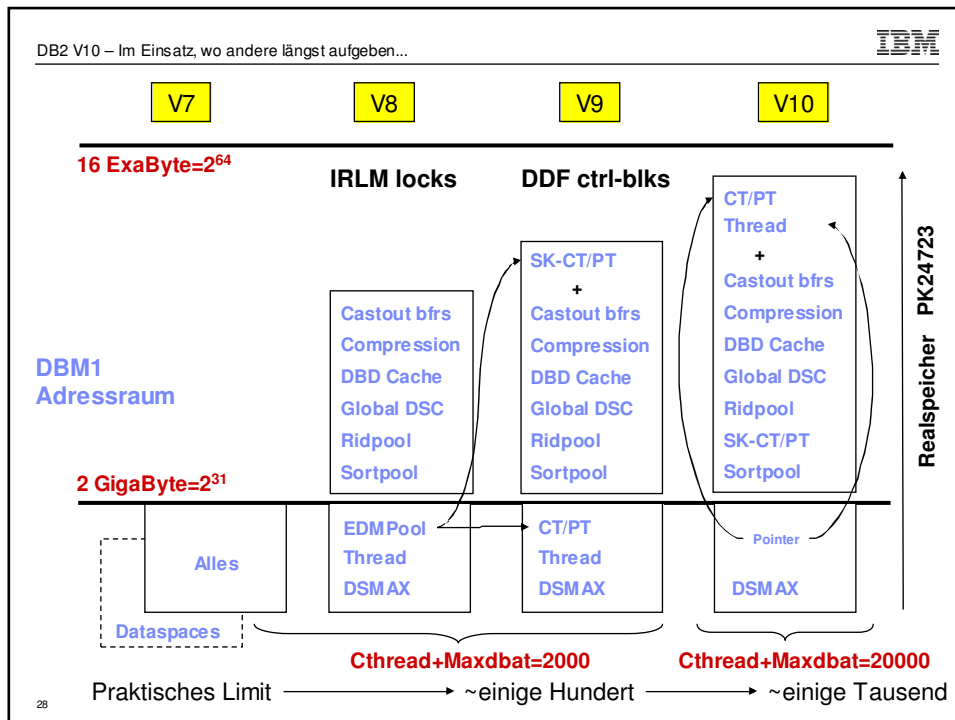
Anwendungsentwickler



Web Anwendung



- Mehr 64bit Nutzung
- Konsolidierung
- Mehr Benutzer
- Potential für reduzierten CPU Verbrauch



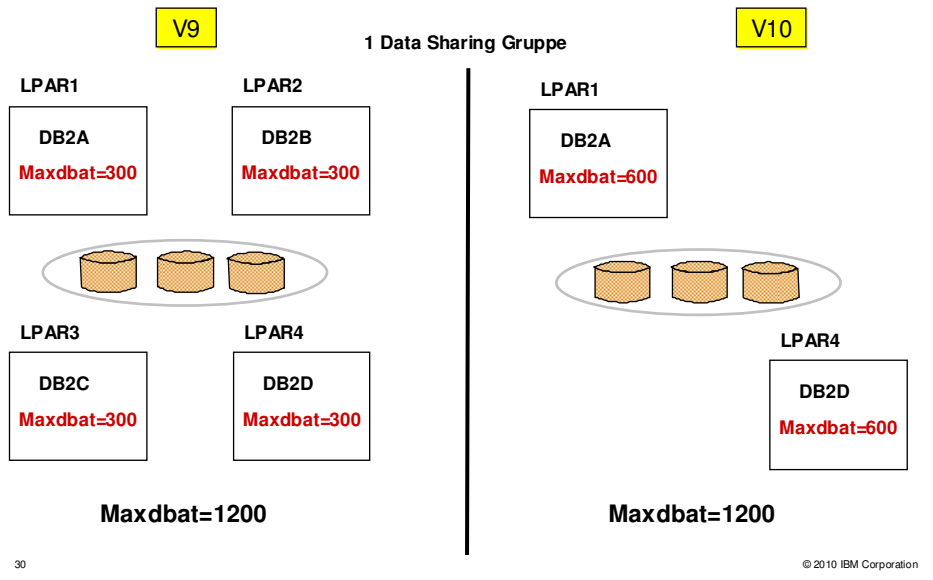
DB2 V10 – Im Einsatz, wo andere längst aufgeben...

## Storage Defintionen

- **HVSHARE** (seit V9)
  - 132GB pro DB2 system
  - DISPLAY VIRTSTOR,HVSHARE
- **HVCOMMON** (seit V10)
  - 6GB pro DB2 system
  - DISPLAY VIRTSTOR,HVCOMMON
- **LFAREA** (seit V10) für Bufferpools mit PGFIX
  - IRA120E (80%) oder IRA121E (95%)
  - DISPLAY VIRTSTOR,LFAREA (OA34024)

© 2010 IBM Corporation

## DBM1 Storage Relief: Konsolidierungsbeispiel



30

© 2010 IBM Corporation

## DBM1 Storage Relief: Performance Beispiel

- Bis V9 „notwendiges Übel“: Cmtstat=Inactive, Release=Commit, Maxdbat~300
- Ab **V10 Performance**: Cmtstat=Inactive, **Release=Deallocate**, Maxdbat~3000
  - Problem: Wie switched man auf Deallocate ?
  - Lösung:
    - Packages mit Deallocate rebinden.
    - Mit neuem Kommando Modify DDF, Pkgrel(Bndopt/Commit) switchen zwischen Online (Deallocate) und Batch (Commit) Zeiten.

31

© 2010 IBM Corporation

## Weitere DDF Erweiterungen (PM26480)

### ■ Modify DDF, Alias

- Hinzufügen, Verändern und Löschen von bis zu 40 “dynamischen” Aliasen
- Unterschied zu den “static” Aliasen definiert mit DSNJU003 und angezeigt mit DSNJU004
  - “Dynamic” Aliase werden von DSNJU004 nicht angezeigt

### ■ Online CDB

- Änderungen in CDB werden sofort wirksam für neue rausgehende Connections (kein DDF Stop und Start mehr)
- Display Location Output reflektiert diese Änderung

## DB2 V10: Qualitätssicherung

Systemprogrammierer



Datenbankadministrator



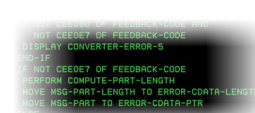
Data Warehousing

- Zugriffspfad
- Explain
- Optimizer

ISV Anwendung



OLTP Anwendung



Endbenutzer



Web Anwendung



Anwendungsentwickler





## Plan/Package Management

- Neue Column Lastused
  - in SYSPLAN
  - In SYSPACKAGE/SYSPACKCOPY
  - zPARM DISABLE\_EDMRTS (PM37672)
- Plan Stability
  - Neue Catalog Tabelle SYSIBM.SYSPACKCOPY
  - Aufbau wie SYSIBM.SYSPACKAGE
  - Neue Column COPYID, um die „Copy Version“ anzuzeigen
    - Original = 2
    - Previous = 1
    - Basic = 0 (wird in SYSPACKAGE geführt)
- Rebind Parameter APRETAINDUP, um gleiche Zugriffspfade von Kopien zu behalten
  - Default ist Yes wie in V9

## Package Management

- Directory tablespace SPT01 wird auf Inline Lob umgestellt werden wegen Performance-und Space-Optimierung
  - PM27073, PM27811
  - Neuer zPARM SPT01\_INLINE\_LENGTH
  - zPARM Compress\_SPT01 macht damit weiterhin Sinn
- Weitere Erweiterungen mit PM25679, z.B.
  - Explain(Only) für Rebind Package (bei Bind geht es schon)

## CURRENT EXPLAIN MODE

- Bietet die Möglichkeit dynamisches SQL automatisch zu erklären
- Optionen
  - NO
  - YES  
Nach Prepare/Execute der SQLs wird die Explain Information abgelegt
  - EXPLAIN  
Nach dem Prepare wird Explain Information abgelegt, SQLs werden nicht ausgeführt (SQLcode +217)
- Special Register
  - SET CURRENT EXPLAIN MODE =
  - JDBC Property *currentExplainMode*=
    - Ab JCC Version 3.61 oder 4.11
- PLAN\_TABLE und DSN\_STATEMENT\_CACHE\_TABLE pro SQLID
  - Ansonsten SQLcode-219

## Explain Tables

- Unicode Requirement für Explain Tables ab CM
  - Falls pre-V8 Format
    - SQLcode -20008, Reason 2
  - Falls in V8 oder V9 Format (unabhängig vom Encoding Schema)
    - SQLcode +20520, Reason 2
  - Falls in V10 Format, aber noch in EBCDIC
    - SQLcode -878
- Neue Column SECTNOI, um eindeutigen Match zwischen Explain Tables und z.B. SYSPACKSTMT herzustellen.

## Explain Tables: Beispiele Encoding Scheme

- V10 Format und kein Unicode
- V9 Format und EBCDIC oder Unicode

```
DSNX200I  $ BIND SQL ERROR
          USING SYSADM AUTHORITY
          PLAN=(NOT APPLICABLE)
          DBRM=JAVAPGM1
          STATEMENT=68
          SQLCODE=-878
          SQLSTATE=53094
          TOKENS=PLAN_TABLE
          CSECT NAME=DSNXOD0Z
          RDS CODE=-150
```

**RC=8**

```
DSNX105I  $ BIND SQL WARNING
          USING SYSADM AUTHORITY
          PLAN=(NOT APPLICABLE)
          DBRM=JAVAPGM1
          STATEMENT=68
          SQLCODE=20520
          SQLSTATE=01694
          TOKENS=2 SYSADM.PLAN_TABLE
```

**RC=4**

## Explain Tables: Beispiel SECTNOI

```
SELECT * FROM SYSIBM.SYSPACKSTMT A,
          SYSIBM.SYSPACKAGE C,
          <owner>.PLAN_TABLE
WHERE A.COLLID = C.COLLID
      AND A.LOCATION = C.LOCATION
      AND A.NAME = C.NAME
      AND A.VERSION = C.VERSION
      AND A.COLLID = PLAN_TABLE.COLLID
      AND A.NAME = PLAN_TABLE.PROGNAME
      AND A.VERSION = PLAN_TABLE.VERSION
      AND A.QUERYNO = PLAN_TABLE.QUERYNO
      AND C.BINDTIME = PLAN_TABLE.BIND_TIME
      AND A.SECTNOI = PLAN_TABLE.SECTNOI
      AND ((A.COLLID = <collectionID> AND
            A.NAME = <packageName> AND
            A.VERSION = <versionID> AND A.LOCATION = ''))
      AND A.EXPLAINABLE = 'Y'
      AND A.QUERYNO > -1
      AND PLAN_TABLE.SECTNOI > -1
ORDER BY A.COLLID , A.NAME , A.VERSION , A.EXPLAINABLE, A.STMTNO , A.SEQNO
```

## Safe Query Optimization

- **RID Pool Overflow**
  - Bisher: Tablespace Scan
  - Mit V10: Workfile kann als Overflow Bereich benutzt werden
    - Einstellung mittels zPARM MAXTEMPS\_RID
    - Muß aber auf MAXTEMPS abgestimmt sein, MAXTEMPS ist oberstes Limit
- **Optimizer betrachtet jetzt nicht nur die Kosten, sondern auch das Risiko eines Zugriffspfads**
  - Zugriffspfade mit ähnlichen Kosten werden nach ihrem Risiko gelistet: Niedrigstes Risiko gewinnt !

## In List Optimization

- **Where col1 IN (?, ?, ?), col1 ist in non-unique Index**
  - In V9: Index wird 3\*mal durchsucht
    - z.B. Accesstype=N und kein Prefetch
  - In V10: Index wird nur 1\*mal durchsucht. Ermöglicht wird dies durch eine In-Memory Table DSNINnnn.
    - z.B. Accesstype=IN und List Prefetch
- **Where col1 IN (?, ?, ?) And col2 IN (?, ?, ?), col1 und col2 sind die beiden Spalten in non-unique Index**
  - In V9: Non matching index access
  - In V10: Matching index access mit Matchcols 2

## Query Transformation

- Predicate Transitive Closure (PTC)
  - In V9: Sehr unwahrscheinlich, daß T2 als erste Tabelle genommen wurde, aber in V10:

```
SELECT * FROM T1, T2 WHERE T1.C1 = T2.C1
AND T1.C1 IN (?, ?, ?)
```

Optimizer generiert dieses PTC

## SQL Pagination

- Seitenweises Blättern
  - Bedingungen für jedes OR Predikat
    - mind. 1 matching predicate
    - mapped auf den selben IX
  - Auswirkungen
    - In V9 multiple IX Access
    - In V10 Accesstype=NR

```
SELECT * FROM JAVATB01 WHERE
(Nachname = 'Daser' AND Vorname > 'Christian')
OR (Nachname > 'Daser')
ORDER BY Nachname, Vorname
FETCH FIRST 20 ROWS ONLY
```

Query für die 2. Seite

Nachname	Vorname
-----	-----
Abel	Elvira
Bromberger	Heiko
...	...
...	...
...	...
Daser	Christian

1. Seite


Nachname	Vorname
-----	-----
Daser	Daniela
Hartmann	Peter
...	...
...	...
...	...
Kistenberger	Georg

2. Seite

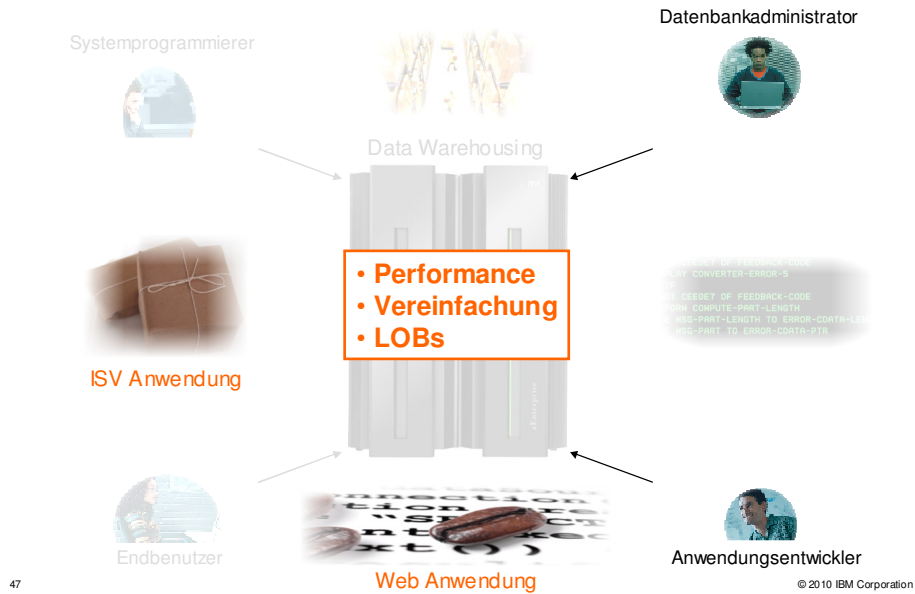
## Parallelism Erweiterungen (Degree=Any)

- Multi-Row-Fetch ist erlaubt
  - Nur für Read Only Cursors
- Workfile kann gemeinsam benutzt werden
  - Nur für CPU Parallelism (kein Full Join !)
- Effektivität bei Parallelism:
  - Aufteilung ist nicht mehr abhängig z.B. von den Partitionsgrenzen:
    - Ziel ist gleichmäßige Aufteilung auf Basis Rekordanzahl
    - „Dynamic Record Range Partitioning“
  - Neues Modell für Parallelism
    - Anzahl der parallelen Tasks weiterhin unverändert, aber kleinere Happen für die Tasks, dafür arbeitet eine Task mehrere Happen ab
    - „Straw Model“

## Agenda

- 
- 09:00 Einführung
  - 09:15 Migration & Vorteile nach der Migration
  - 10:15 Neue Funktionen zur Qualitätssicherung
  - 10:45 Pause
  - 11:00 Anwendungsoptimierung
  - 11:35 Neue Anforderungen an IT-Sicherheit
  - 12:15 Mittagspause
  - 13:00 DB2 auf Zeitreise
  - 14:00 Pause
  - 14:15 Erweiterungen von OLTP bis Warehousing
  - 15:45 Abschluss
  - 16:00 Veranstaltungsende

## DB2 V10: Anwendungsoptimierung



## Optimierung dynamischer & flexibler ? Anwendungen



- Nicht selten **fehlender Zugriff auf Source Code** für Anpassungsmaßnahmen
- Trend zu möglichst datenbank- und plattformneutralen, **generischen Anwendungen**
- Wenn **Optimierung**, dann nur für **ausgewählte Datenbankmanagementsysteme**

## Statements mit Literalen überfüllen den DB2 Cache

### Kunde

☒ **kundennummer**

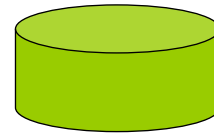
☐ **vorname**

☒ **nachname**
 Maier

☒ **ort**
 Stuttgart

```
SELECT + KUNDENNUMMER +
FROM ISV.KUNDE WHERE + ORT
+ = + 'Stuttgart' + AND
+ NACHNAME + = + 'Maier'
```

Stmt J  
Stmt I  
Stmt G  
Stmt H  
Stmt E  
Stmt C  
Stmt F  
Stmt D  
Stmt B  
Stmt A



49

© 2010 IBM Corporation

## Änderungsmöglichkeit ist jedoch oft nicht gegeben

### ▪ Bisherige Ausgangslage:

- ✗ Schlechte Coding Practices oder Notwendigkeit für hochdynamisches SQL mit String Concatenation
- ✗ Nutzung externer Frameworks oder Anwendungen

```
SELECT ORT FROM ISV.KUNDE WHERE KUNDENNUMMER = 2337168
```

...

### ▪ Lösung mit DB2 10:

- **Literal Replacement** liefert generisches SQL
- Dabei werden ohne Eingriff in die Anwendung Literale durch **&** ersetzt (ähnlich Parameter-Markern)

```
SELECT ORT FROM ISV.KUNDE WHERE KUNDENNUMMER = &
```

50

© 2010 IBM Corporation



## Aktivieren von Literal Replacement in Anwendungen

Auf Connection-Ebene in Java (oder mit JCC-Property **statementConcentrator**)

```
( (DB2Connection) con ).setDBStatementConcentrator (2) ;  
pstmt = conn.prepareStatement ("SELECT NACHNAME FROM  
ISV.KUNDE WHERE KUNDENNUMMER = '47582'");
```

Im ODBC Initialization File

```
MVSDEFAULTSSID=V10A      LITERALREPLACEMENT=1  
AUTOCOMMIT=0              ...
```

Beim PREPARE als Attribut: – **CONCENTRATE STATEMENTS  
WITH LITERALS**

## Dynamische Anwendungen erhöhen die Komplexität

### Kunde

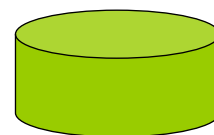
- ☒ kundennummer
- ☐ vorname
- ☐ nachname
- ☒ anrede
- ☐ titel
- ☒ strasse
- ☒ hausnummer
- ☒ postleitzahl
- ☒ ort
- ☐ festnetznummer
- ☐ mobilfunknummer
- ☒ emailadresse
- ...

UPDATE ISV.KUNDE SET ...

$$2^x - 1$$

unterschiedliche  
Kombinationen

$$\text{z.B. } 2^{12} - 1 = 4.095$$



## Entwicklung & Betrieb leiden unter dieser Komplexität

### ■ Bisherige Lösungen:

- Sehr viele einfache SQL Statements
  - ✗ Hoher Programmieraufwand & Sicherheitsrisiken
  - ✗ Ineffiziente Nutzung Dynamic Statement Cache
- Wenige komplexe SQL Statements
  - ✗ Hoher Programmieraufwand mit unnötigen Zugriffen
  - ✗ Schwierigkeiten mit Default-Werten bei INSERTs

### ■ Lösung mit DB2 10:

- **Special Null Indicators** für INSERT/UPDATE/MERGE
- Null Indicator Variable mit Werten von -5 (Default), -7 (kein UPDATE/INSERT bzw. Default) oder 0

## Nutzung von Special Null Indicators in Anwendungen

Mit Hostvariablen, z.B. in COBOL (mit Bindparameter EXTENDEDINDICATOR)

```
HV1 = 47582; IND1 = 0;
HV2 = "Max"; IND3 = -5;
EXEC SQL INSERT INTO ISV.KUNDE VALUES
(:HV1 :IND1, :HV2, :HV3 :IND3);
```

Mit Parameter-Markern, z.B. in Java (mit JCC-Property enableExtendedIndicators)

```
pstmt = con.prepareStatement("UPDATE ISV.KUNDE
SET NACHNAME = ?, VORNAME = ? WHERE KUNDENNR = ?");
pstmt.setString(1, "Mustermann");
((DB2PreparedStatement)pstmt).setDBUnassigned(2);
pstmt.setInt(3, 47582);
```

## Timeout-Risiko bei Lesezugriffen auf aktuelle Daten

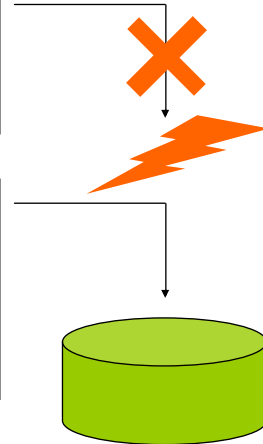
### Anwendung A



```
SELECT ... FROM ISV.KUNDE
SELECT ... FROM ISV.KUNDE
SELECT ... FROM ISV.KUNDE ...
```

```
INSERT INTO ISV.KUNDE
INSERT INTO ISV.KUNDE ...
```

```
DELETE FROM ISV.KUNDE
DELETE FROM ISV.KUNDE ...
```



### Anwendung B

56

© 2010 IBM Corporation

## Bisher keine zufriedenstellende Lösung vorhanden

### ▪ Bisherige Alternativen:

- Isolation Level Uncommitted Read
  - ✗ Unter Umständen Verarbeitung von inkonsistenten Daten
- Isolation Level Cursor Stability oder höher
  - ✗ SQLCODE -913 ... CAUSED BY DEADLOCK OR TIMEOUT

### ▪ Lösung mit DB2 10:

- **Currently Committed** liefert für die Isolation Level CS oder RS den Datenstand vom letzten Commit-Zeitpunkt
- Kein Warten auf die Freigabe inkompatibler Locks von INSERT/DELETE Operationen (**nicht für UPDATE**)

57

© 2010 IBM Corporation

## Einsatz von Currently Committed in Anwendungen

### Als Bind-Parameter für statisches SQL

```

▶▶—BIND PACKAGE—(—location-name.—collection-id—)
▶—CONCURRENTACCESSRESOLUTION(—USECURRENTLYCOMMITTED—)
  —WAITFOROUTCOME—
  
```

Auf Connection-Ebene in Java (oder mit JCC-Property **concurrentAccessResolution**)

```
( (DB2Connection) con ).setDBConcurrentAccessResolution (1) ;
```

Beim PREPARE als Attribut: — **USE CURRENTLY COMMITTED**  
— **WAIT FOR OUTCOME**

## Currently Committed Beispielszenario I – DELETE

### Transaktion 1

**DELETE**

⋮



**COMMIT**

ISV.KUNDE

**X**

47582	Maier
50153	Schmidt
93661	Müller

**S**

### Transaktion 2

Isolation Level CS  
CURRENTDATA(NO)

**SELECT**

47582 Maier

3. Row wird trotzdem zurückgeliefert unter Verwendung von **Currently Committed**

## Currently Committed Beispielszenario II – INSERT

### Transaktion 1



INSERT

X

ISV.KUNDE

47582 Maier

50153 Schmidt

93661 Müller

...



COMMIT

### Transaktion 2

Isolation Level CS  
oder RS

S



SELECT

3. Row wird über-  
gangen unter  
Verwendung  
von **Currently  
Committed**

## Inkompatible Datentypen bedeuten Effizienzverluste

- Bisherige Ausgangslage:
  - X Aufwändige Portierung von anderen Plattformen
  - X Ggf. explizite Casts in Anwendungen notwendig
- Lösung mit DB2 10:
  - **Kompatibilität** zwischen **numerischen** und **String Datentypen** sowie Ausweitung des **Implicit Casting**

CHAR  
VARCHAR  
GRAPHIC  
VARGRAPHIC



DECIMAL

(SMALL/BIG) INT  
NUMERIC  
DECIMAL  
REAL  
DOUBLE  
(DEC) FLOAT



VARCHAR

## DB2-interne Ablage für kleine LOBs unvorteilhaft

- Bisheriges Ablageverfahren:
  - ✗ LOB-Daten werden unabhängig von ihrer Größe in einer Auxiliary Table im LOB Table Space abgelegt
- Lösung mit DB2 10:
  - Mit **Inline LOB** Columns verbleibt ein vorgegebener Teil der LOB-Daten direkt im Base Table Space (**muss im Reordered Row Format sein**)
  - Dadurch lassen sich Performance-Vorteile realisieren sowie Plattenplatz einsparen
  - Inline LOBs erlauben die Definition von Default-Werten sowie von Indexes (on Expression)

## Definition und Funktionsweise von Inline LOBs

```
CREATE TABLE ISV.VERTRAG
  (VERTRAGSNR INTEGER NOT NULL,
   VERTRAG CLOB(500K) INLINE LENGTH 1000);
```

```
ALTER TABLE ISV.VERTRAG ALTER COLUMN
  VERTRAG SET INLINE LENGTH 1500);
```

- Inline Length zwischen 0 und 32680 Bytes
  - LOB < Inline Length: Gesamtes LOB im Base Table Space
  - LOB > Inline Length: Über Inline-Teil hinausgehend wird ausgelagert
- **zPARM LOB\_INLINE\_LENGTH** legt subsystemweit den Defaultwert fest (0 bedeutet per Default kein Inline-Teil)
- Table geht in **REORG Pending** Status bei **Verkleinerung**

## Komplexität durch generierte LOB und XML Objekte

### ▪ Bisherige Ausgangslage:

- ✗ Vor allem Kaufsoftware erstellt oft eine Vielzahl an DB2 Objekten (Tabellen, Indexes...), von denen später nur wenige tatsächlich genutzt werden

### ▪ Lösung mit DB2 10:

- **DEFINE NO** greift nun auch für **LOB** und **XML Objekte** (Table Spaces und Indexes) und verzögert die Erstellung der entsprechenden VSAM Data Sets
- Der Verzicht auf die direkte, physische Allokation mindert auch die Auswirkungen auf den Speicherbedarf und bestehende Backup-/Recovery-Prozesse

## Verwendung von DEFINE NO für LOB und XML

```
CREATE TABLESPACE ISVTS002
  IN DATABASE ISVDB001
  ...
  DEFINE NO;
```

Alternativ:

**zPARM IMPDSDEF** in V10  
ausgeweitet auf LOB und  
XML Objekte, Default = YES

```
CREATE TABLE ISV.VERTRAG
  (VERTRAGSNR INTEGER NOT NULL,
  VERTRAG CLOB(500K) INLINE LENGTH 250)
  IN ISVDB001.ISVTS002;
```



No data set names found

## Auswirkungen von DEFINE NO für LOB und XML

```
INSERT INTO ISV.VERTRAG (7289113, ... CLOB < 250 BYTES ...);
```



```
DSNC910.DSNDBC.ISVDB001.ISVTS002.I0001.A001
DSNC910.DSNDBD.ISVDB001.ISVTS002.I0001.A001
```

```
INSERT INTO ISV.VERTRAG (7289114, ... CLOB > 250 BYTES ...);
```



```
DSNC910.DSNDBC.ISVDB001.ISVTS002.I0001.A001
DSNC910.DSNDBD.ISVDB001.ISVTS002.I0001.A001
```



```
DSNC000.DSNDBC.ISVDB001.IVERTRVE.I0001.A001
DSNC000.DSNDBC.ISVDB001.LDVEP1X9.I0001.A001
DSNC000.DSNDBD.ISVDB001.IVERTRVE.I0001.A001
DSNC000.DSNDBD.ISVDB001.LDVEP1X9.I0001.A001
```

## Zusätzliche Erweiterungen im Anwendungsumfeld

- Erweiterte Online-Fähigkeit von Utilities für LOBs:
  - **REORG** mit **SHRLEVEL CHANGE**
- Funktionalitätserweiterungen im JCC T2 Treiber:
  - Echtes **Progressive Streaming** für LOBs und XML
  - Unterstützung des **Limited Block Fetch** Protokolls
  - Query Buffer liegen im 64-Bit adressierbaren Storage-Bereich für Anwendungen, die mit 64-Bit JVMs laufen



## DB2 V10: IT-Sicherheit

Sicherheitsadministrator

Datenbankadministrator



Data Warehousing

- Admin-Rechte
- Auditprofile
- Granulare Sicht auf Daten

ISV Anwendung

OLTP Anwendung



Endbenutzer



Anwendungsentwickler

Web Anwendung

70

© 2010 IBM Corporation

## Sicherheitsbetrachtungen

- Sicherheitsbedrohungen von extern & intern
- Granulare Sicht von speziellen Benutzergruppen auf sensitive Daten
- Vermeidung von impliziten Berechtigungen beim Datenzugriff durch privilegierte Benutzer
- Trennung von Verantwortlichkeiten
  - Datenmanagement & Berechtigungsvergabe
- Überwachung & Protokollierung von Datenzugriffen und Berechtigungsvergaben

71

© 2010 IBM Corporation

## Sicherheit & Compliance

- Mehr granulare administrative Berechtigungen
- Schutz von sensiblen Daten
  - Privilegierte Benutzer ohne Datenzugriff
- Security-Administrator SECADM zur Durchführung von DB2 Berechtigungen
- Anwendungsentwickler benötigen keine Tabellenberechtigungen zur Zugriffspfadanalyse
- Integrierte Audit-Funktion

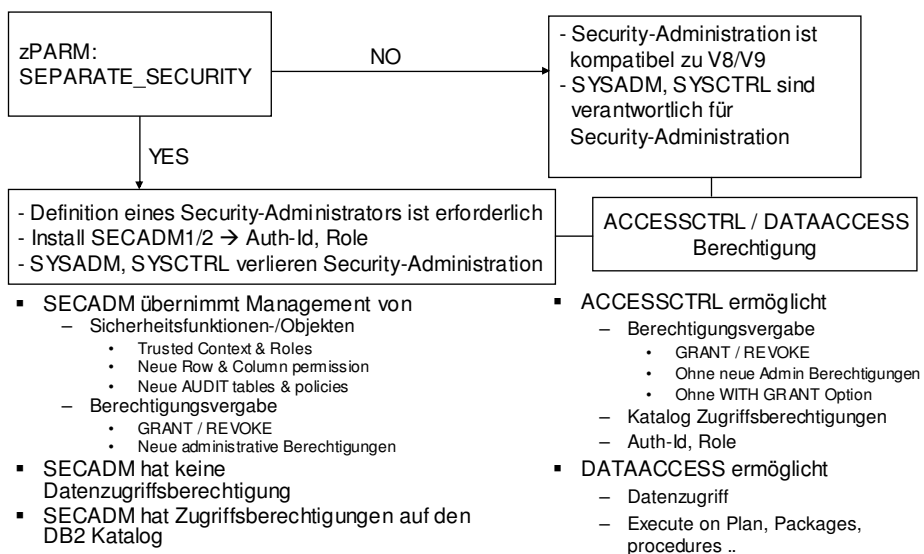


- **Row & column level access control**
  - Ermöglicht Maskierung von Daten
  - Begrenzt den Datenzugriff auf einzelnen Datenzellen
- **Temporale Daten**

72

© 2010 IBM Corporation

## Security-Administration vs. System-Administration



73

© 2010 IBM Corporation

## Neue Administrative Berechtigungen

- Definition zur System Administration
- GRANT DBADM .... ON SYSTEM TO Auth-id, Role

- System DBADM
- Management von DB Objekten
- ohne Sicherheitsobjekte
- ohne WITH GRANT Option

### WITH ACCESSCTRL

- Berechtigungsvergabe
- ohne Sicherheitsobjekte
- ohne WITH GRANT Option
- default

### WITH DATAACCESS

- Datenzugriff
- Execute on Plan, Packages, procedures ..
- default

### WITHOUT ACCESSCTRL

### WITHOUT DATAACCESS

## Neue Administrative Berechtigungen

- Definition zur dedizierten Performance Analyse
- GRANT SQLADM .... ON SYSTEM TO Auth-id, Role

- Management von Performance Monitoring und Zugriffspfadanalyse
- Ausführung von
  - EXPLAIN
    - dynamic sql statements
    - BIND option
  - START, STOP und DISPLAY PROFILE command
  - PREPARE, DESCRIBE TABLE statement
  - Privileges
    - Explain, STATS, MONITOR2
    - DB2 supplied stored procedures und routines
  - Zugriffen auf Katalog
  - RUNSTATS und MODIFY STATISTICS utility
- Keine Berechtigung zum
  - Datenzugriff
  - Management von DB Objekten
  - Execute von Plan, Packages, Stored Procedures ...

## Neue Administrative Berechtigungen

- Definition zur Zugriffspfadanalyse
- GRANT EXPLAIN .... ON SYSTEM TO Auth-id, Role

- Ausführung von
  - EXPLAIN
    - dynamic sql statements
    - BIND option
  - PREPARE, DESCRIBE TABLE statement
  - Explain privilege
- Keine Berechtigung zum
  - Datenzugriff ...

## Wegnahme von Berechtigungen

- Optionale Kaskadierung beim Berechtigungsentzug
- zPARM: REVOKE DEP\_PRIVILEGES
- REVOKE ... FROM Auth-id, Role ...  
(NOT) INCLUDING DEPENDENT PRIVILEGES

### zPARM: NO

- REVOKE bewahrt kaskadierende Berechtigungen
- Fehlernachricht
  - INCLUDING DEPENDENT PRIVILEGES clause
- NOT INCLUDING ... default clause
  - DATAACCESS
  - ACCESSCTRL
  - System DBADM

### zPARM: YES

- REVOKE entzieht kaskadierende Berechtigungen
- Fehlernachricht
  - NOT INCLUDING DEPENDENT PRIVILEGES clause
- Ausnahme
  - DATAACCESS
  - ACCESSCTRL
  - System DBADM privileges

### zPARM: SQLSTMT default

- REVOKE berücksichtigt
  - NOT INCLUDING
  - INCLUDING
- DEPENDENT PRIVILEGES clause
- Default:
  - INCLUDING DEPENDENT PRIVILEGES

## Neue AUDIT Möglichkeiten

- Neue Audit Policy Tabelle
  - Katalogtabelle SYSAUDITPOLICIES
  - SECADM Security-Administrator verwaltet Audit policies
  - ohne AUDIT table clause
  - Wildcarding von Tabellennamen
  - Dynamische (De)Aktivierung
- Auditing von
  - Privileged Users
    - Zugriff auf Daten
  - SQL Aktivitäten gegen Tabellen
    - Read & update Zugriffe
    - Protokolliert SQL statement optional
  - Utility Aktivitäten
  - Grant, Revoke und Trusted Context Aktivitäten
  - Autorisierungs-/Authentifizierungsfehler

## Erstellen von AUDIT POLICIES

- INSERT in SYSAUDITPOLICIES
- (De)Aktivieren von audit policies
  - STOP/START TRACE mit AUDTPLY option
    - Generiert AUDIT traces
  - Autostart von bis zu 8 audit policies
- DISPLAY TRACE mit AUDTPLY option

Column Name	Col No	Col Type	Length
---	---	---	---
AUDITPOLICYNAME	1	VARCHAR	128
OBJECTSCHEMA	2	VARCHAR	128
OBJECTNAME	3	VARCHAR	128
OBJECTTYPE	4	CHAR	1
CREATEDTS	5	TIMESTAMP	10
ALTEREDTS	6	TIMESTAMP	10
CHECKING	7	CHAR	1
VALIDATE	8	CHAR	1
OBJMAINT	9	CHAR	1
EXECUTE	10	CHAR	1
CONTEXT	11	CHAR	1
SECMAINT	12	CHAR	1
SYSADMIN	13	VARCHAR	128
DBADMIN	14	VARCHAR	128
DBNAME	15	VARCHAR	24
COLLID	16	VARCHAR	128
DB2START	17	CHAR	1
IBMREQD	18	CHAR	1

## AUDIT Beispiel

- Erstellen einer AUDITADMIN1 policy zum Auditing von SYSADM und SYSOPR berechtigte Benutzer

```
INSERT INTO SYSIBM.SYSAUDITPOLICIES
(AUDITPOLICYNAME, SYSADMIN)
VALUES('AUDITADMIN1', 'OS');
```

- Erstellen einer policy zum Auditing von Insert, Update, Delete sql statements gegen Tabellen die mit Namen E\_P und Schema Namen TSCHEMA beginnen.

```
INSERT INTO SYSIBM.SYSAUDITPOLICIES
(AUDITPOLICYNAME, OBJECTSCHEMA, OBJECTNAME, OBJECTTYPE, EXECUTE)
VALUES('TEST2', 'TSCHEMA', 'E_P%', 'T', 'C');
```

## Zugriffsbeschränkungen für Tabellen

- View – Definitionen
  - SQL Nutzung
    - INSTEAD of TRIGGER für update Fähigkeit
  - View privileges
  - Row & Column level access
- Multi-level Security / MLS
  - Table privileges
  - Security label information wird an Tabellen hinzugefügt
    - SECLABEL von RACF
    - Abb. von Hierarchien → Multi-level
  - SQL und Utility Nutzung
  - Row level access
  - Zugriff, Result set abhängig vom SECLABEL
- Row & Column level access
  - Neue DB Objekte
    - Eingeschränkte Sicht auf Tabellen für bestimmte User (SQL ID) und Gruppen
    - Maskierung von Informationen
    - Optimizer SQL Rewrite
  - SQL Nutzung
    - SELECT, UPDATE, INSERT, DELETE und MERGE

## Row & Column level access

- Row level security

```
CREATE PERMISSION policy-name ON table-name  
FOR ROWS WHERE search-condition  
ENFORCED FOR ALL ACCESS ENABLE
```

- Column level security

- Maskierung von Spaltenwerten

```
CREATE MASK mask-name ON table-name FOR COLUMN  
column-name RETURN CASE expression ENABLE;
```

## Wer darf die sensiven Daten verarbeiten?

- SESSION-USER

- Primary authorization Id

- CURRENT SQLID

- SQL authorization Id

- VERIFY\_GROUP\_FOR\_USER

- Neue BIF
- Primary und Secondary authorization Ids
- Return ,1', wenn authorization Id vorhanden!

```
WHERE  
VERIFY_GROUP_FOR_USER(SESSION_USER,'MGR','PAYROLL') = 1
```

- VERIFY\_ROLE\_FOR\_USER

## Aktivierung der Row & Column Security

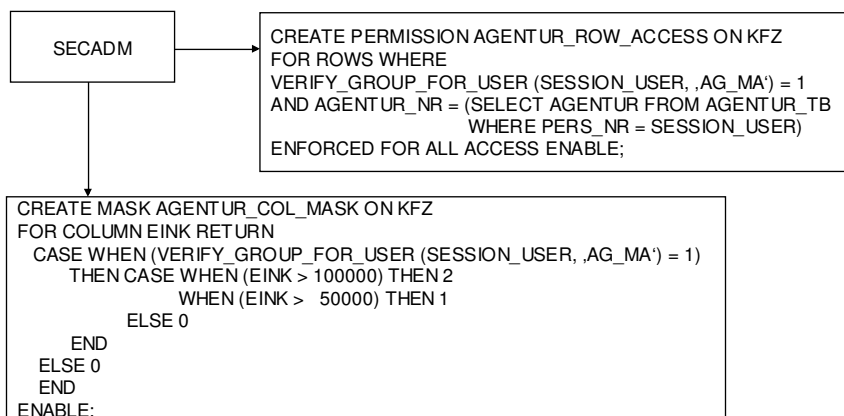
- Deaktivierung bedeutet keine Einschränkung der SQL Verarbeitung durch PERMISSION und MASK Objekte

```
ALTER TABLE table-name
  DEACTIVATE ROW LEVEL ACCESS CONTROL
  DEACTIVATE COLUMN LEVEL ACCESS CONTROL;
```

- Aktivierung bedeutet Einschränkung der SQL Verarbeitung gemäß PERMISSION und MASK Definitionen
  - Optimizer SQL rewrite

```
ALTER TABLE table-name
  ACTIVATE ROW LEVEL ACCESS CONTROL
  ACTIVATE COLUMN LEVEL ACCESS CONTROL;
```

## Row & Column level access - Beispiel



- Mitarbeiter einer Vers.- Agentur (RACF group AG\_MA) können auf die KFZ Verträge der zugeordneten Agentur zugreifen.
- Maskierung der Einkommenswerte, um eine Transparenz zu vermeiden.

```
ALTER TABLE KFZ
  ACTIVATE ROW LEVEL ACCESS CONTROL
  ACTIVATE COLUMN LEVEL ACCESS CONTROL;
```



## Workshop – Sicherheit im Zentrum

### ▪ Motivation

- Aktuelle Prägnanz des Themas Datenschutz und IT Sicherheit in der Öffentlichkeit
- Medieninteresse
- Gesetzliche Bestimmungen und Normen
- Vermehrte Kundenanfragen

### ▪ Zielsetzung

- Betrachtung des Themas über Produktsicht hinaus
- Festlegung auf möglichst konkrete Szenarien
- Focus auf moderne Webarchitekturen
- Viel Diskussion und Interaktion
- Aufmerksamkeit erwecken
- Handlungsempfehlungen geben

## Sicherheit im Zentrum: Agenda

Thema	Start	Ende	Zeit	Sprecher
Begrüßung und Vorstellung	09:00	09:20	00:20	Alle
Aktuelle Bedrohungen in Webarchitekturen	09:20	10:00	00:40	Christian Daser Patrick Hempeler
<b>Pause</b>	<b>10:00</b>	<b>10:15</b>	<b>00:15</b>	<b>Alle</b>
Blickwinkel Authentifizierung	10:15	11:00	00:45	Holger Wunderlich
Blickwinkel Autorisierung	11:00	11:35	00:35	G. Kistenberger
Blickwinkel Verschlüsselung	11:35	12:10	00:35	G. Kistenberger
<b>Mittagessen</b>	<b>12:10</b>	<b>13:00</b>	<b>00:50</b>	<b>Alle</b>
Blickwinkel Anwendungsentwicklung	13:00	13:35	00:35	Christian Daser
Auditing und Compliance	13:35	14:10	00:35	Carsten Hinz
<b>Pause</b>	<b>14:10</b>	<b>14:25</b>	<b>00:15</b>	<b>Alle</b>
Demo auf System z	14:25	14:45	00:20	Christian Daser
<b>Zusammenfassung und Diskussion</b>	<b>14:45</b>	<b>15:15</b>	<b>00:30</b>	<b>Alle</b>

## Agenda

- 09:00 Einführung
- 09:15 Migration & Vorteile nach der Migration
- 10:15 Neue Funktionen zur Qualitätssicherung
- 10:45 Pause
- 11:00 Anwendungsoptimierung
- 11:35 Neue Anforderungen an IT-Sicherheit
- 12:15 Mittagspause
- 13:00 DB2 auf Zeitreise
- 14:00 Pause
- 14:15 Erweiterungen von OLTP bis Warehousing
- 15:45 Abschluss
- 16:00 Veranstaltungsende

## DB2 V10: DB2 auf Zeitreise

Sicherheitsadministrator



Datenbankadministrator



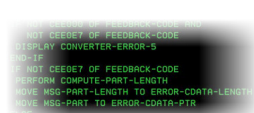
Data Warehousing

- Automatische Historisierung
- Abbildung von Geschäftsvorfällen
- Zeitzonen

ISV Anwendung



OLTP Anwendung



Endbenutzer



Web Anwendung



Anwendungsentwickler

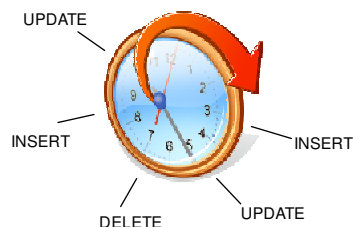


## Temporal Tables

- Inhalte von Tabellen waren bislang ohne zeitlichen Zusammenhang
- In V10 gibt es zwei Erweiterungen
  - Pflege der Historie von Daten (SYSTEM\_TIME)  
DB2 legt eine Historisierung von Datensätzen an
  - Zeitliche Spezifikation von Datensätzen (BUSINESS\_TIME)  
Anwendung gibt die Gültigkeit von Daten zu einer bestimmten Zeit an  
Geschäftsvorfälle können zeitlich korrekt abgebildet werden
- SYSTEM\_TIME und BUSINESS\_TIME können kombiniert (Bi-Temporal Tables) verwendet werden

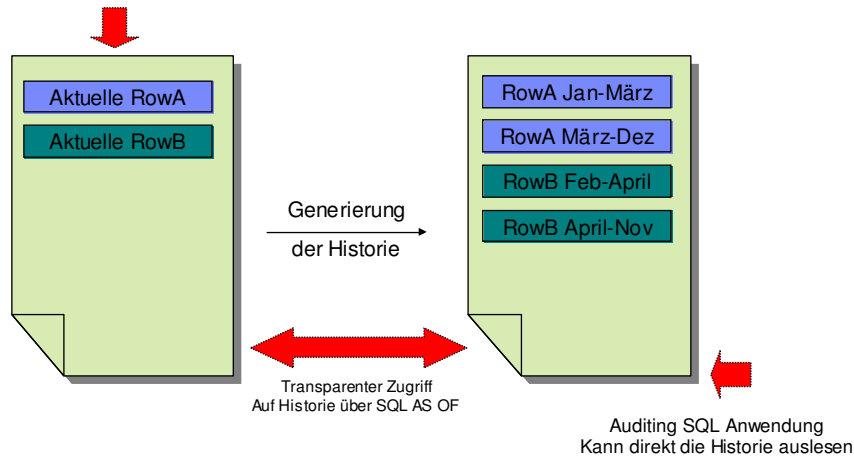
## SYSTEM\_TIME

- Audit und Complianceregeln der Unternehmen erfordern ein protokollieren aller Änderungen in der Datenbank
- Auditing für ISV Anwendungen ist sehr aufwändig
- Unternehmen müssen nachvollziehen, wie sich Daten zeitlich geändert haben



## Architektur für SYSTEM\_TIME

Online SQL Anwendung



93

© 2010 IBM Corporation

## Vereinfachte DDL für SYSTEM\_TIME

```
CREATE TABLE KFZ (
    VERTRAGSNUMMER CHAR(4),
    KUNDENNUMMER_ID CHAR(4),
    KASKO_ID CHAR(6),
    JAEHRL_FAHRLLEISTUNG INTEGER,
    SYS_START TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN,
    SYS_END TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END,
    CREATE_ID TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID,
    PERIOD SYSTEM_TIME(SYS_START, SYS_END));

CREATE TABLE KFZ_SYS_HIST (
    VERTRAGSNUMMER CHAR(4),
    KUNDENNUMMER_ID CHAR(4),
    KASKO_ID CHAR(6),
    JAEHRL_FAHRLLEISTUNG INTEGER,
    SYS_START TIMESTAMP(12) NOT NULL,
    SYS_END TIMESTAMP(12) NOT NULL,
    CREATE_ID TIMESTAMP(12));

ALTER TABLE KFZ ADD VERSIONING USE HISTORY TABLE KFZ_SYS_HIST;
```

94

© 2010 IBM Corporation

## Beispiele für SYSTEM\_TIME

```
INSERT INTO KFZ (VERTRAGSNUMMER, KUNDENNUMMER_ID, KASKO_ID,
JAEHRL_FAHRLLEISTUNG) VALUES ('A123', '123', 'T150', 18000); --am 01. Dezember 2010
```

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	T150	18000	2010-12-01	9999-12-31

```
UPDATE KFZ SET KASKO_ID='V300' WHERE VERTRAGSNUMMER='A123'; --am 01. März 2011
```

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	V300	18000	2011-03-01	9999-12-31

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	T150	18000	2010-12-01	2011-03-01

```
UPDATE KFZ SET JAEHRL_FAHRLLEISTUNG=25000 WHERE VERTRAGSNUMMER='A123'; --am 01. Juni 2011
```

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	V300	25000	2011-06-01	9999-12-31

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	T150	18000	2010-12-01	2011-03-01
A123	123	V300	18000	2011-03-01	2011-06-01

## Abfragemöglichkeiten

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	V300	25000	2011-06-01	9999-12-31

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	T150	18000	2010-12-01	2011-03-01
A123	123	V300	18000	2011-03-01	2011-06-01

```
SELECT KASKO FROM KFZ WHERE VERTRAGSNUMMER='A123';
```

V300

```
SELECT KASKO FROM KFZ FOR SYSTEM_TIME AS OF '2010-12-25' WHERE VERTRAGSNUMMER='A123';
```

T150

```
DELETE FROM KFZ WHERE VERTRAGSNUMMER='A123'; --am 15. September 2011
```

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
---------	-------	-------	--------	-----------	---------

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	T150	18000	2010-12-01	2011-03-01
A123	123	V300	18000	2011-03-01	2011-06-01
A123	123	V300	25000	2011-06-01	2011-09-15

```
SELECT KASKO FROM KFZ FOR SYSTEM_TIME AS OF '2010-12-25' WHERE VERTRAGSNUMMER='A123';
```

T150

## Wissenswertes

- SYS\_START inklusive, SYS\_END exklusive
- Zugriff auf History Tabelle möglich durch eigene GRANTS
  - SELECT
  - INSERT, UPDATE: Vorsicht, da keine Period Check Constraints
  - DELETE
- DDL
  - Gleichheit von Columns und Datentypen von Basis und History Tabelle
  - Systemspalten können IMPLICITLY HIDDEN angelegt werden
  - Kein ALTER, ausgenommen ALTER ADD COLUMN (PM31313)
  - DROP Versioning ist möglich, vorher kein DROP der Version-Tabelle
  - History Tabelle wird im Katalog als Type=H ausgewiesen
  - Restriktionen bei GENERATED ALWAYS, CLONES, Security Labels, Multitable Tablespace
- Utilities
  - Point in Time Recovery nur im Set
  - Keine Utilities, die Daten löschen (LOAD Replace, Reorg Discard,...)
- Auswirkungen
  - Speicherbedarf
  - Performance

## BUSINESS\_TIME

- Geschäftsvorfälle zeitlich korrekt darstellen
- Datenmodellierung in der Vergangenheit, Gegenwart und Zukunft
- Bereits bei vielen Kunden manuell implementiert
- DB2 bietet nun eine SQL Schnittstelle

## Vereinfachte DDL für BUSINESS\_TIME

```
CREATE TABLE KFZ
(
  VERTRAGSNUMMER          CHAR(4)          NOT NULL,
  KUNDENNUMMER_ID         CHAR(4)          NOT NULL,
  KASKO_ID                 CHAR(6)          NOT NULL,
  JAEHRL_FAHRLLEISTUNG     INT             NOT NULL,
  BUS_START                DATE            NOT NULL,
  BUS_END                  DATE            NOT NULL,
  PERIOD BUSINESS_TIME(BUS_START, BUS_END)
)

CREATE UNIQUE INDEX IX_KFZ
ON KFZ (VERTRAGSNUMMER, BUSINESS_TIME WITHOUT OVERLAPS);
```

## BUSINESS\_TIME Beispiel

- Kunde schliesst einen neuen KFZ Vertrag mit Teilkasko ab
- Für die Sommermonate wandelt er seinen Vertrag in Vollkasko
- Ein Schadensfall wird korrekt prozessiert
- Aufgrund Zeitversatz bei der Prozessierung von Änderungen wird ein Schadensfall fehlerhaft behandelt
- Kundenbeschwerde muss rückwirkend analysiert werden

DB2 V10 – Im Einsatz, wo andere längst aufgeben...
IBM

### KFZ - Vertrag

Max Mustermann  
Musterstraße 16  
93777 Musterort  
Kundennummer: 123

Versicherung GmbH  
Firmenstraße 11  
33777 Firmenort

01. Januar

Vertragsnummer:	<b>A123</b>
Wagen:	Mittelklassewagen
Typ:	hsn 0588
Fahrleistung:	20.000 km
Kasko:	<b>Teilkasko</b>
Selbstbeteiligung:	<b>150 €</b>
Zeitraum:	ab <b>März</b>
Mtl. Beitr.:	200 Euro

Kunde schliesst  
neuen KFZ  
Vertrag ab

Neuertrag

Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez	...
T150												

102
© 2010 IBM Corporation

DB2 V10 – Im Einsatz, wo andere längst aufgeben...
IBM

```

INSERT INTO KFZ (VERTRAGSNUMMER, KUNDENNUMMER_ID, KASKO_ID, JAEHRL, BUS_START, BUS_END)
VALUES ('A123', '123', 'T150', 20000, '2011-03-01', '9999-12-31');
INSERT INTO KFZ (VERTRAGSNUMMER, KUNDENNUMMER_ID, KASKO_ID, JAEHRL, BUS_START, BUS_END)
VALUES ('A123', '123', 'V300', 18000, '2011-04-01', '2012-06-01');
SQLCode -803: Zeitlich nicht eindeutig!
```

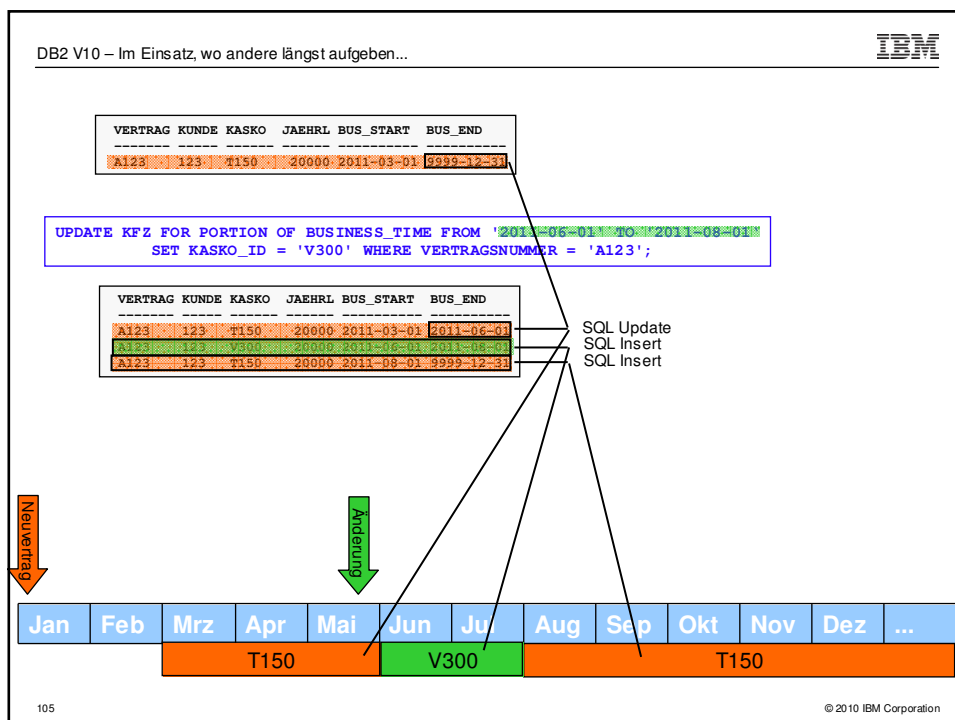
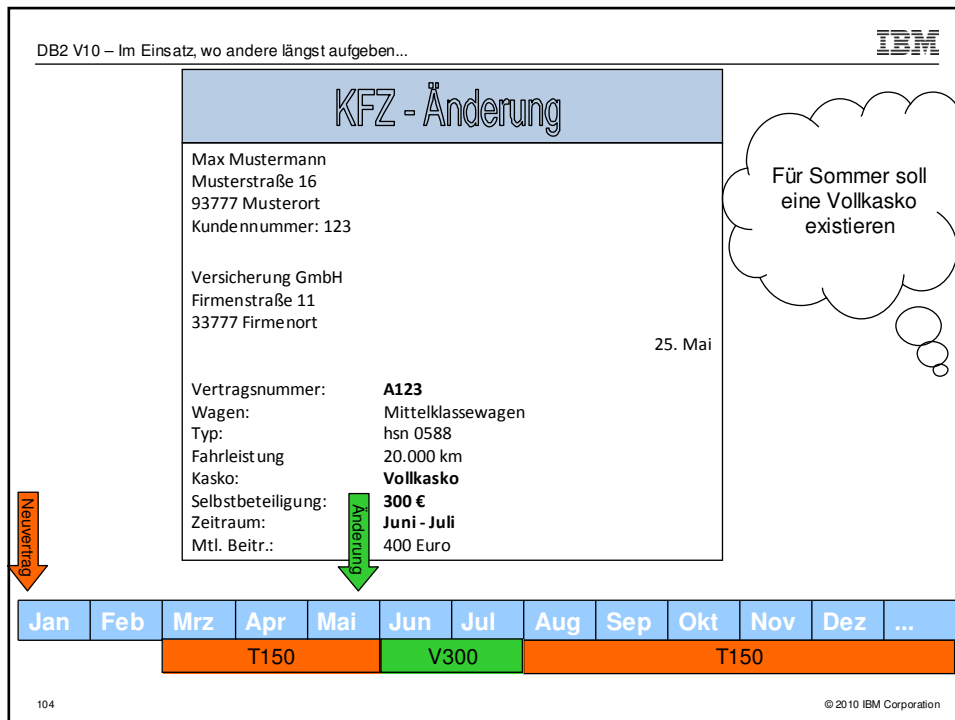
VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END
A123	123	T150	20000	2011-03-01	9999-12-31

Neuertrag

Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez	...
T150												


103
© 2010 IBM Corporation





DB2 V10 – Im Einsatz, wo andere längst aufgeben...

IBM



Quelle: [http://commons.wikimedia.org/wiki/File:Rolling\\_Unter.jpg](http://commons.wikimedia.org/wiki/File:Rolling_Unter.jpg)

Schaden durch  
Unaufmerk-  
samkeit am  
7.Juni

Neuvertrag

Änderung

Schaden

Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez	...
		T150			V300		T150					

106

© 2010 IBM Corporation

DB2 V10 – Im Einsatz, wo andere längst aufgeben...

IBM

KFZ - Schadensmeldung

Max Mustermann  
Musterstraße 16  
93777 Musterort  
Kundennummer: 123

Versicherung GmbH  
Firmenstraße 11  
33777 Firmenort

10. Juli

Vertragsnummer:     **A123**

**Bitte um Kostenerstattung von 10000EUR  
zwecks Unfall vom 07.Juni**

Existierte für den  
7.Juni eine  
Vollkasko ?

Neuvertrag

Änderung

Schaden

Meldung

Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez	...
		T150			V300		T150					

107

© 2010 IBM Corporation

VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END
A123	123	T150	20000	2011-03-01	2011-06-01
A123	123	V300	20000	2011-06-01	2011-08-01
A123	123	T150	20000	2011-08-01	9999-12-31

Existierte für den  
7.Juni eine  
Vollkasko ?

```
SELECT KASKO FROM KFZ FOR BUSINESS_TIME AS OF '2011-06-07' WHERE VERTRAGSNUMMER='A123';
```

V300



Schaden wird  
übernommen



Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez	...
				T150		V300				T150		

108

© 2010 IBM Corporation



Schaden durch  
Unaufmerk-  
samkeit am  
15.August



Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez	...
				T150		V300				T150		

110

© 2010 IBM Corporation

## KFZ - Schadensmeldung

Max Mustermann  
Musterstraße 16  
93777 Musterort  
Kundennummer: 123

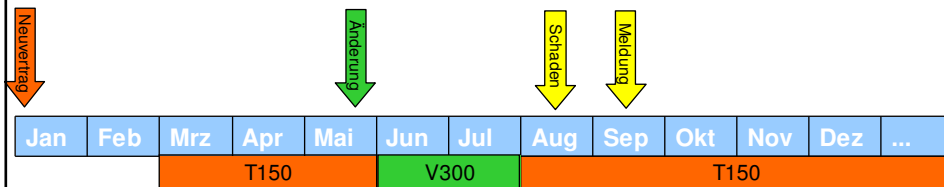
Versicherung GmbH  
Firmenstraße 11  
33777 Firmenort

15. Sept

Vertragsnummer: **A123**

Bitte um Kostenerstattung  
zwecks Schaden vom 15. August

Existierte für den  
15. August eine  
Vollkasko ?



111

© 2010 IBM Corporation

VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END
A123	123	T150	20000	2011-03-01	2011-06-01
A123	123	V300	20000	2011-06-01	2011-08-01
A123	123	T150	20000	2011-08-01	9999-12-31

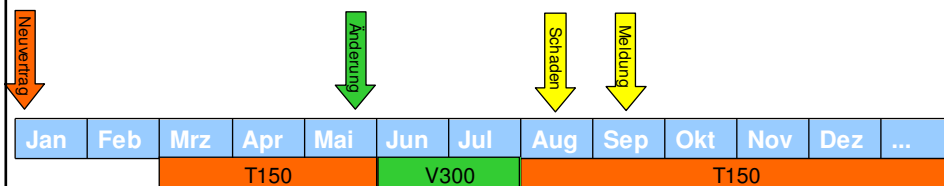
Existierte für den  
15. August eine  
Vollkasko ?

```
SELECT KASKO FROM KFZ FOR BUSINESS_TIME AS OF '2011-08-15' WHERE VERTRAGSNUMMER='A123';
```

T150



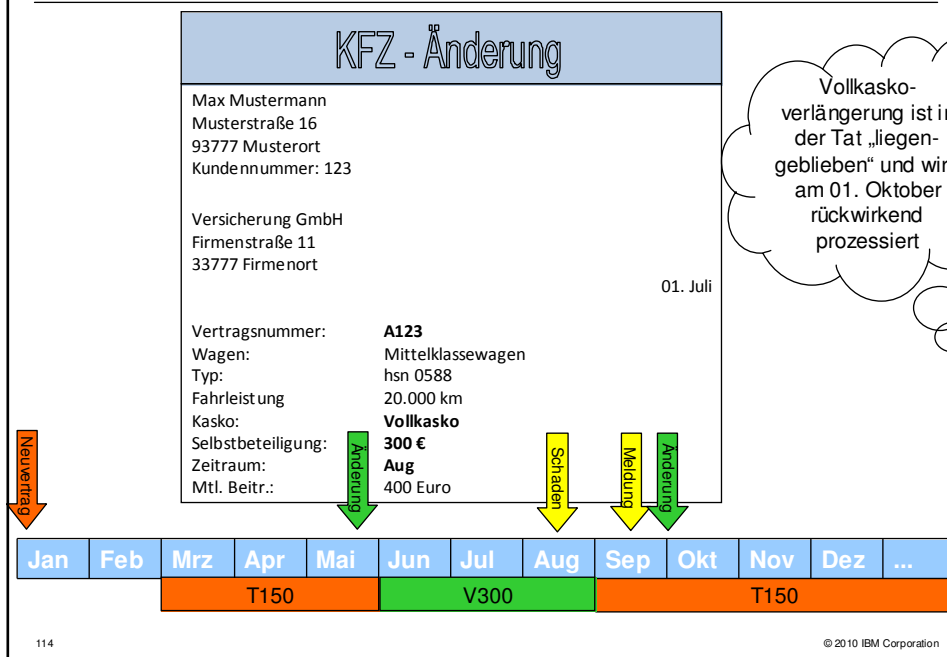
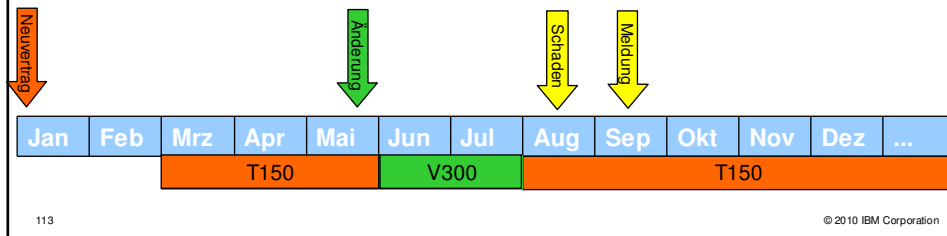
Schaden wird  
abgelehnt

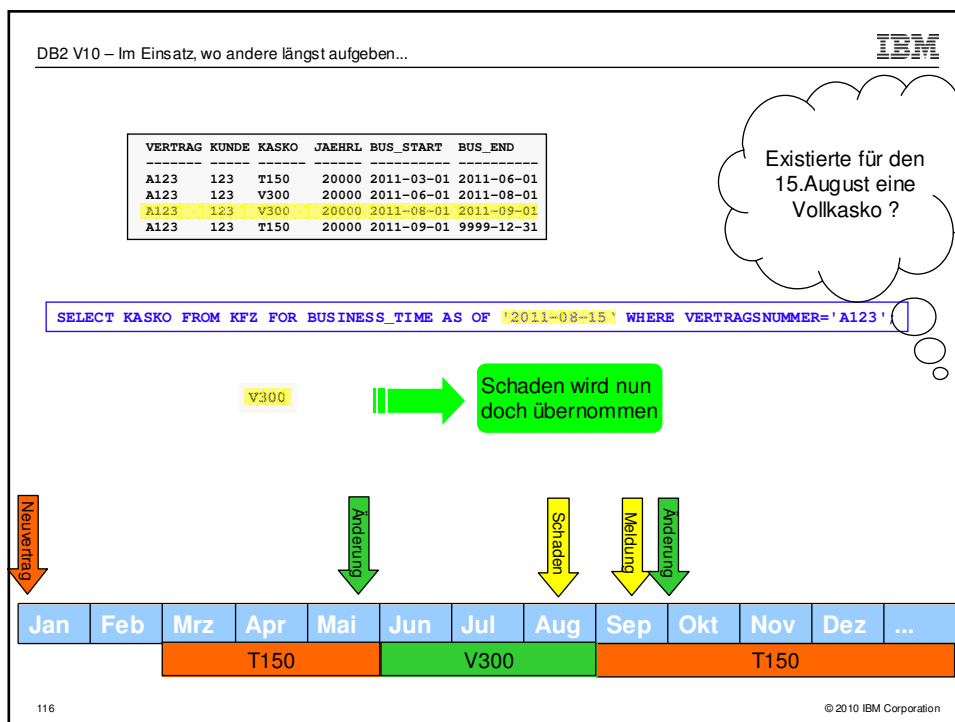
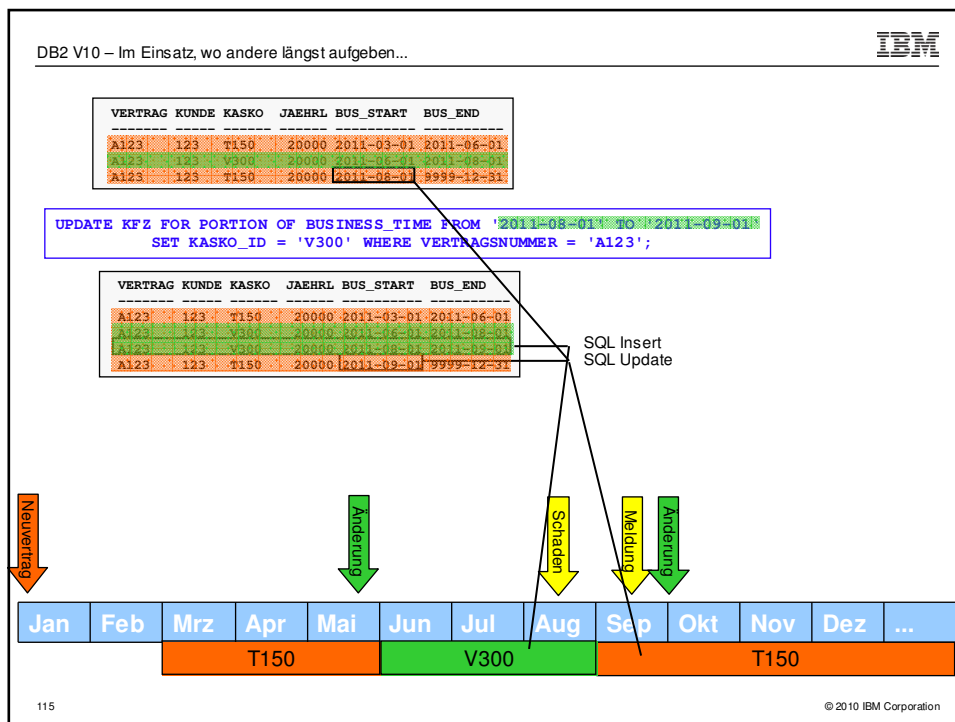


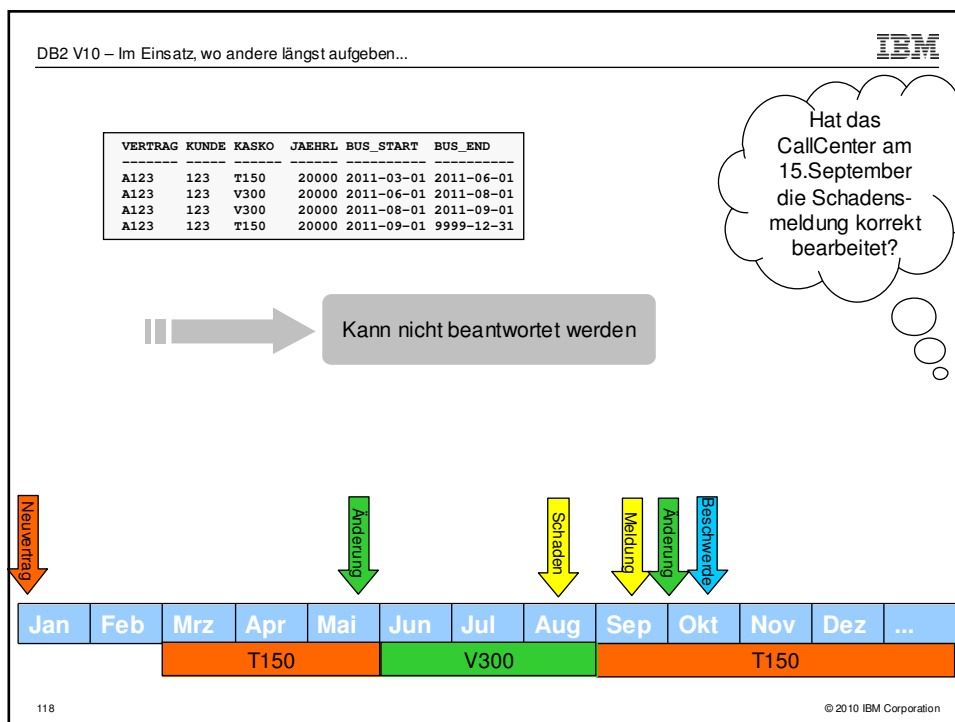
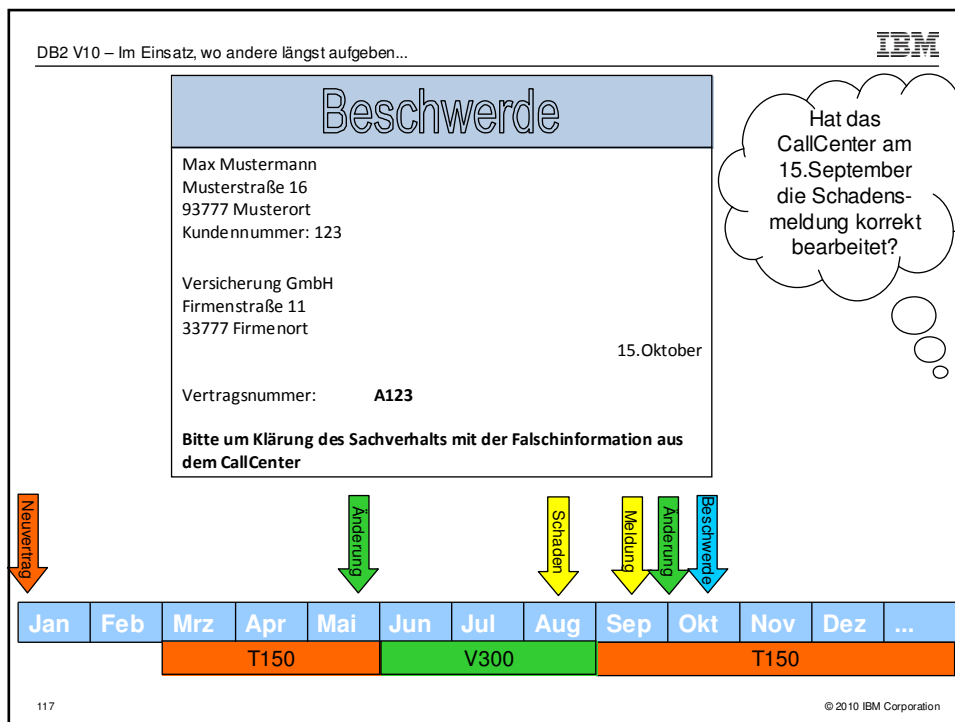
112

© 2010 IBM Corporation

- Kunde behauptet, er hätte die Vollkasko per Einschreiben am 01. Juli verlängert
- Kunde beschwert sich über das CallCenter







DB2 V10 – Im Einsatz, wo andere längst aufgeben...
IBM

## KFZ - Kündigung

Max Mustermann  
Musterstraße 16  
93777 Musterort  
Kundennummer: 123

Versicherung GmbH  
Firmenstraße 11  
33777 Firmenort

Vertragsnummer: **A123**  
Hiermit kündige ich die Versicherung zum Jahresende

01. November

Neuvertrag

Änderung

Schaden

Meldung

Änderung

Beschwerde

Kündigung

Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez	...
		T150			V300			T150				

119
© 2010 IBM Corporation

DB2 V10 – Im Einsatz, wo andere längst aufgeben...
IBM

VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END
A123	123	T150	20000	2011-03-01	2011-06-01
A123	123	V300	20000	2011-06-01	2011-08-01
A123	123	V300	20000	2011-08-01	2011-09-01
A123	123	T150	20000	2011-09-01	9999-12-31

```
DELETE FROM KFZ FOR PORTION OF BUSINESS_TIME FROM '2012-01-01' TO '9999-12-31'
WHERE VERTRAGSNUMMER = 'A123';
```

VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END
A123	123	T150	20000	2011-03-01	2011-06-01
A123	123	V300	20000	2011-06-01	2011-08-01
A123	123	V300	20000	2011-08-01	2011-09-01
A123	123	T150	20000	2011-09-01	2012-01-01

Neuvertrag

Änderung

Schaden

Meldung

Änderung

Kündigung

Jan	Feb	Mrz	Apr	Mai	Jun	Jul	Aug	Sep	Okt	Nov	Dez	...
		T150			V300			T150				

120
© 2010 IBM Corporation



## Bi-temporal Tables

- Kombiniert SYSTEM\_TIME und BUSINESS\_TIME
- Ermöglicht auch alle Änderungen der BUSINESS\_TIME systemtechnisch festzuhalten

```
CREATE TABLE KFZ (
    VERTRAGSNUMMER CHAR(4),
    KUNDENNUMMER_ID CHAR(4),
    KASKO_ID CHAR(6),
    JAEHRL_FAHRLLEISTUNG INTEGER,
    BUS_START DATE NOT NULL,
    BUS_END DATE NOT NULL,
    SYS_START TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN,
    SYS_END TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END,
    CREATE_ID TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID,
    PERIOD BUSINESS_TIME(BUS_START, BUS_END),
    PERIOD SYSTEM_TIME(SYS_START, SYS_END));

CREATE TABLE KFZ_SYS_HIST (
    VERTRAGSNUMMER CHAR(4),
    KUNDENNUMMER_ID CHAR(4),
    KASKO_ID CHAR(6),
    JAEHRL_FAHRLLEISTUNG INTEGER,
    BUS_START DATE NOT NULL,
    BUS_END DATE NOT NULL,
    SYS_START TIMESTAMP(12) NOT NULL,
    SYS_END TIMESTAMP(12) NOT NULL,
    CREATE_ID TIMESTAMP(12));

ALTER TABLE KFZ ADD VERSIONING USE HISTORY TABLE KFZ_SYS_HIST;

CREATE UNIQUE INDEX IX_KFZ ON KFZ (VERTRAGSNUMMER, BUSINESS_TIME WITHOUT OVERLAPS);
```

121

oration

VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END	SYS_START	SYS_END
A123	123	T150	20000	2011-03-01	2011-06-01	2011-05-25	9999-12-31
A123	123	V300	20000	2011-06-01	2011-08-01	2011-05-26	9999-12-31
A123	123	V300	20000	2011-08-01	2011-09-01	2011-10-01	9999-12-31
A123	123	T150	20000	2011-09-01	9999-12-31	2011-10-01	9999-12-31

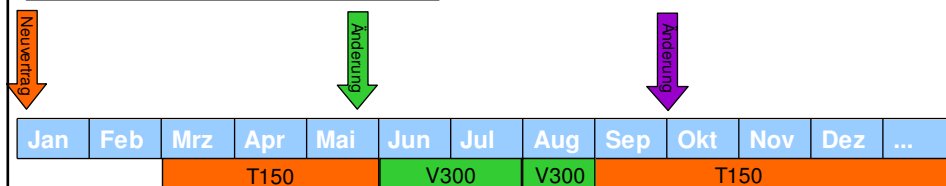
Automatisch gepflegte Tabelle KFZ\_SYS\_HIST

VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END	SYS_START	SYS_END
A123	123	T150	20000	2011-03-01	9999-12-31	2011-01-01	2011-05-25
A123	123	T150	20000	2011-08-01	9999-12-31	2011-05-26	2011-10-01

A123	123	T150	20000	2011-03-01	2011-06-01
A123	123	V300	20000	2011-06-01	2011-08-01
A123	123	V300	20000	2011-08-01	2011-09-01
A123	123	T150	20000	2011-09-01	9999-12-31

A123	123	T150	20000	2011-03-01	2011-06-01
A123	123	V300	20000	2011-06-01	2011-08-01
A123	123	T150	20000	2011-08-01	9999-12-31

A123	123	T150	20000	2011-03-01	9999-12-31
------	-----	------	-------	------------	------------



122

© 2010 IBM Corporation

DB2 V10 – Im Einsatz, wo andere längst aufgeben...

**IBM**

VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END	SYS_START	SYS_END
A123	123	T150	20000	2011-03-01	2011-06-01	2011-05-25	9999-12-31
A123	123	V300	20000	2011-06-01	2011-08-01	2011-05-25	9999-12-31
A123	123	V300	20000	2011-08-01	2011-09-01	2011-05-25	9999-12-31
A123	123	T150	20000	2011-09-01	9999-12-31	2011-10-01	9999-12-31

Automatisch gepflegte Tabelle KFZ\_SYS\_HIST

VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END	SYS_START	SYS_END
A123	123	T150	20000	2011-03-01	9999-12-31	2011-01-01	2011-05-25
A123	123	T150	20000	2011-08-01	9999-12-31	2011-05-25	2011-10-01

SELECT KASKO FROM KFZ FOR BUSINESS TIME AS OF '2011-08-15'  
FOR SYSTEM TIME AS OF '2011-09-15'  
WHERE VERTRAGSNUMMER='A123';

Callcenter hatte aufgrund damaliger Sachlage richtig entschieden

Neuertrag

Änderung

Schaden

Meldung

Änderung

Beschwerde

Jan Feb Mrz Apr Mai Jun Jul Aug Sep Okt Nov Dez ...

T150 V300 T150

123

© 2010 IBM Corporation

DB2 V10 – Im Einsatz, wo andere längst aufgeben...

**IBM**

## TIMESTAMP(12)

- Die Genauigkeit des Timestamp wird auf 12 Stellen erweitert  
yyyy-mm-dd-hh.mm.ss.nnnnnnnnnnnnn
- Special Register CURRENT TIMESTAMP
- JDBC Treiber arbeitet in der Regel mit TIMESTAMP(9)
  - Cast nach VARCHAR möglich

125

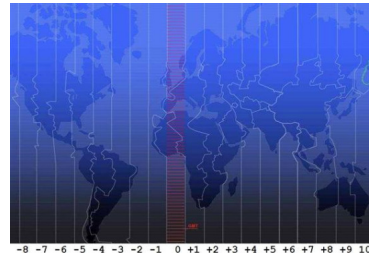
© 2010 IBM Corporation

## DB2 auf Zeitreise

- Eine Timezone ist der Unterschied in HH:MM zwischen der lokalen Zeit und der UTC (oder früher Greenwich Mean Time GMT)

- $\pm th:tm$ , where

- $\pm$  positives oder negatives Offset
- *th* Stundenunterschied
- *tm* Minutenunterschied



- **TIMESTAMP WITH TIMEZONE**

- Year-Month-Day-Hour.Minutes.Seconds time zone
- 2007-11-05-08.00.00-08:00

SQL

```
CREATE TABLE TABLE1
(C1 TIMESTAMP WITH TIME ZONE, C2 INTEGER);
```

```
--SVL
```

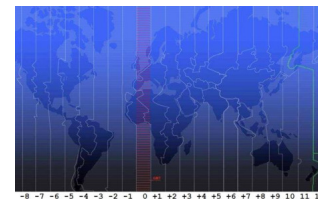
```
INSERT INTO TABLE1 VALUES ( '2010-08-10-08.00.00-08:00', 1);
```

```
--Deutschland
```

```
INSERT INTO TABLE1 VALUES ( '2010-08-10-17.00.00+01:00', 2);
```

```
SELECT * FROM TABLE1 WHERE C1 = '2010-08-10-17.00.00+01:00';
```

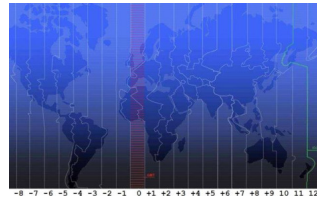
C1	C2
2010-08-10-08.00.00.000000-08:00	1
2010-08-10-17.00.00.000000+01:00	2



SQL

## Special Registers

- CURRENT TIMEZONE
  - Unterschied zwischen UTC und DB2 Server
- SESSION TIME ZONE
  - Zeitzone der Anwendung Format ' $\pm th:tm$ '
  - SET SESSION TIME ZONE = '+01:00';



```
SET SESSION TIME ZONE = '+01:00';
SELECT C1 AT LOCAL, C2 FROM TABLE1;
```

1	C2
2010-08-10-17.00.00.000000+01:00	1
2010-08-10-17.00.00.000000+01:00	2

```
SELECT C1 AT TIMEZONE '-08:00', C2 FROM TABLE1;
```

1	C2
2010-08-10-08.00.00.000000-08:00	1
2010-08-10-08.00.00.000000-08:00	2

## DB2 V10: XML

Systemprogrammierer



Datenbankadministrator



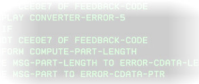
Data Warehousing

- XML Validierung
- XML Update
- XML Performance

ISV Anwendung



OLTP Anwendung



Endbenutzer



Web Anwendung



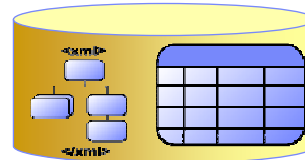
Anwendungsentwickler



## Übersicht pureXML in DB2 V10

### ■ V9

- XML Datentyp mit Parsing und Schema Validierung
- XPATH zur Navigation
- SQLXML Queries mit XPATH
- Umwandlung XML  $\leftrightarrow$  Relational



### ■ V10

- Erweiterungen bzgl. Schema Validierung
- Check Pending
- Modifizieren von XML Dokumenten
- Tablespacestruktur für Performance, Locking und erweiterte Funktionen
- BinaryXML für Datenaustausch und Transport

## DDL für SYSTEM\_TIME mit XML

```
CREATE TABLE KFZ (
    VERTRAGSNUMMER CHAR(4) FOR SBCS DATA NOT NULL,
    KUNDENNUMMER_ID CHAR(4) FOR SBCS DATA NOT NULL,
    KASKO_ID CHAR(6) FOR SBCS DATA NOT NULL,
    JAEHRL_FAHRLLEISTUNG INTEGER,
    VDOCUMENT XML,
    SYS_START TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN,
    SYS_END TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END,
    CREATE_ID TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID,
    PERIOD SYSTEM_TIME(SYS_START, SYS_END));

CREATE TABLE KFZ_SYS_HIST (
    VERTRAGSNUMMER CHAR(4) FOR SBCS DATA NOT NULL,
    KUNDENNUMMER_ID CHAR(4) FOR SBCS DATA NOT NULL,
    KASKO_ID CHAR(6) FOR SBCS DATA NOT NULL,
    JAEHRL_FAHRLLEISTUNG INTEGER,
    VDOCUMENT XML,
    SYS_START TIMESTAMP(12) NOT NULL,
    SYS_END TIMESTAMP(12) NOT NULL,
    CREATE_ID TIMESTAMP(12));

ALTER TABLE KFZ ADD VERSIONING USE HISTORY TABLE KFZ_SYS_HIST;
```

Es entstehen zwei XML Tablespaces (Aktuelle XML Daten und Historie)

## Zusätzlich: XML Multiversioning

- XML Daten sind in einem separaten XML Tablespace abgelegt
- XML Multiversioning in V10
  - Performance / Locking
  - XML Subdocument Update
  - Currently Committed
  - SELECT from DELETE/UPDATE

Basistabelle

DOCID	...	XMLCol
1		
2		
3		

V# update timestamp  
(LRSN/RBA) (14 bytes)

NodeID index

(DOCID, NODEID, ET V, ST V)

B+tree

DOCID	NODEID	XMLDATA
1	02	
2	02	
2	0208	
2	0210	
3	02	
3	02	

XML Tablespace enthält die Daten

132

© 2010 IBM Corporation

## XML Subdocument update

- In DB2 9 for z/OS gab es nur ein Komplettupdate für XML Dokumente
- DB2 10 for z/OS bietet XMLMODIFY
  - INSERT
  - REPLACE
  - DELETE

```

<PurchaseOrder PoNum="11223345">
  <items>
    <item>
      <partid>100-100-01</partid>
      <name>Snow Shovel, Basic 22 inch</name>
      <quantity>4</quantity>
      <price>9.99</price>
    </item>
  </items>
</PurchaseOrder>

```

→

```

<PurchaseOrder PoNum="11223345">
  <items>
    <item>
      <partid>100-100-01</partid>
      <name>Snow Shovel</name>
      <desc>Basic 22 inch</desc>
      <quantity>4</quantity>
      <price>9.99</price>
    </item>
  </items>
</PurchaseOrder>

```

133

© 2010 IBM Corporation

## Replace text node in XML

POID	...	PORDER
11223344	...	<pre>&lt;PurchaseOrder PoNum="11223355" OrderDate="2007-07-18"&gt;   &lt;client&gt;     &lt;companyname&gt;MyComp Ltd&lt;/companyname&gt;     &lt;address&gt;       &lt;line&gt;75, West Street&lt;/line&gt;       &lt;line&gt;1234 Southern City&lt;/line&gt;     &lt;/address&gt;   &lt;/client&gt;   &lt;items&gt;     &lt;item&gt;       &lt;partid&gt;100-100-01&lt;/partid&gt;       &lt;name&gt;Snow Shovel, Basic 22 inch&lt;/name&gt;       &lt;quantity&gt;4&lt;/quantity&gt;       &lt;price&gt;9.99&lt;/price&gt;     &lt;/item&gt;   &lt;/items&gt; &lt;/PurchaseOrder&gt;</pre>

```
UPDATE PURCHASEORDER SET PORDER = xmlmodify(
  'replace value of node /PurchaseOrder/client/address/line[1]
  with "75, West Street"'
WHERE poid=11223344;
```

## Replace node in XML

POID	...	PORDER
11223344	...	<pre>&lt;PurchaseOrder PoNum="11223355" OrderDate="2007-07-18"&gt;   &lt;client&gt;     &lt;companyname&gt;MyComp Ltd&lt;/companyname&gt;     &lt;address&gt;       &lt;street&gt;75, West Street&lt;/street&gt;       &lt;city&gt;1234 Southern City&lt;/city&gt;     &lt;/address&gt;   &lt;/client&gt;   &lt;items&gt;     &lt;item&gt;       &lt;partid&gt;100-100-01&lt;/partid&gt;       &lt;name&gt;Snow Shovel, Basic 22 inch&lt;/name&gt;       &lt;quantity&gt;4&lt;/quantity&gt;       &lt;price&gt;9.99&lt;/price&gt;     &lt;/item&gt;   &lt;/items&gt; &lt;/PurchaseOrder&gt;</pre>

```
UPDATE PURCHASEORDER SET PORDER = xmlmodify('replace node
/PurchaseOrder/client/address with $x', XMLPARSE(document '
  <address>
    <street>75, West Street</street>
    <city>1234 Southern City</city>
  </address>' ) as "x")
WHERE poid=11223344;
```

## Insert node in XML

POID	...	PORDER
11223344	...	<pre>&lt;PurchaseOrder PoNum="11223355" OrderDate="2007-07-18"&gt;   &lt;client&gt;     &lt;companyname&gt;MyComp Ltd&lt;/companyname&gt;     &lt;address&gt;       &lt;line&gt;10, East Main Street&lt;/line&gt;       &lt;line&gt;1234 Southern City&lt;/line&gt;       &lt;country&gt;Germany&lt;/country&gt;     &lt;/address&gt;   &lt;/client&gt;   &lt;items&gt;     &lt;item&gt;       &lt;partid&gt;100-100-01&lt;/partid&gt;       &lt;name&gt;Snow Shovel, Basic 22 inch&lt;/name&gt;       &lt;quantity&gt;4&lt;/quantity&gt;       &lt;price&gt;9.99&lt;/price&gt;     &lt;/item&gt;   &lt;/items&gt; &lt;/PurchaseOrder&gt;</pre>

```
UPDATE PURCHASEORDER SET PORDER = xmlmodify('
  insert node $in as last into /PurchaseOrder/client/address',
  xmlparse(document ' <country>Germany</country>' ) as "in")
WHERE poid=11223344;
```

## Delete node in XML

POID	...	PORDER
11223344	...	<pre>&lt;PurchaseOrder PoNum="11223355" OrderDate="2007-07-18"&gt;   &lt;client&gt;     &lt;companyname&gt;MyComp Ltd&lt;/companyname&gt;     &lt;address&gt;       &lt;line&gt;10, East Main Street&lt;/line&gt;       &lt;line&gt;1234 Southern City&lt;/line&gt;       &lt;country&gt;Germany&lt;/country&gt;     &lt;/address&gt;   &lt;/client&gt;   &lt;items&gt;     &lt;item&gt;       &lt;partid&gt;100-100-01&lt;/partid&gt;       &lt;name&gt;Snow Shovel, Basic 22 inch&lt;/name&gt;       &lt;quantity&gt;4&lt;/quantity&gt;       &lt;price&gt;9.99&lt;/price&gt;     &lt;/item&gt;   &lt;/items&gt; &lt;/PurchaseOrder&gt;</pre>

```
UPDATE PURCHASEORDER SET PORDER = xmlmodify('
  delete node /PurchaseOrder/client/address/country')
WHERE poid=11223344;
```



## XML Schema Validierung - Erweiterungen

- Performanceverbesserungen und Grössenerweiterungen
- In DB2 V9 gab es noch keine systemseitige Abhängigkeit eines Schemas zur DB2 Tabelle/Spalte
  - V9 Anwendungsseitig:
    - SQL und DSN\_XMLVALIDATE Funktion
    - Stored Procedures oder Views und Triggers
  - V10 System- oder Anwendungsseitig:
    - Bisherige Möglichkeiten aus V9
    - Spalte erhält Schema Definitionen anhand „XML type modifier“
      - XSR Name
      - XML Target Namespace und (optional) Schema Location Hint
- Automatische Validierung bei INSERT, UPDATE, LOAD
- Tablespace kann CHECK PENDING werden bei Änderung des Schemas

## Schema Validierung in V9

```
INSERT INTO PURCHASEORDER(poid, status, porder)
  values(11223355, 'Unshipped',
    XMLPARSE(DOCUMENT SYSFUN.DSN_XMLVALIDATE('
    <PurchaseOrder PoNum="11223355" OrderDate="2007-07-18" Status="Shipped">
    <client>
      <companyname>MyComp Ltd</companyname>
      <address>
        <line>10, East Main Street</line>
        <line>1234 Southern City</line>
        <line>Mr. Smith</line>
      </address>
    </client>
    <items>
      <item>
        <partid>100-100-01</partid>
        <name>Snow Shovel, Basic 22 inch</name>
        <quantity>4</quantity>
        <price>9.99</price>
      </item>
      ...
    </PurchaseOrder>', 'SYSXSR.PURCHASEORDER')));
```

## V10: Schema Definitionen als XML type modifier

- Festgelegt über den XSR Name

```
CREATE TABLE PURCHASEORDER ( POID BIGINT NOT NULL,
                             STATUS VARCHAR(10) NOT NULL WITH DEFAULT 'Unshipped',
                             PORDER XML (XMLSCHEMA ID SYSXSR.PURCHASEORDER), ...
```

- Flexible Definition über TargetNamespace und Location Hint  
(empfohlen bei mehreren gültigen XSDs und dynamischer XSD Änderung)

```
CREATE TABLE PURCHASEORDER ( POID BIGINT NOT NULL,
                             STATUS VARCHAR(10) NOT NULL WITH DEFAULT 'Unshipped',
                             PORDER XML (XMLSCHEMA
                                         URI 'http://www.ibm.com/db2/java/xml/ns'
                                         LOCATION 'PurchaseOrderV10.xsd'), ...
```

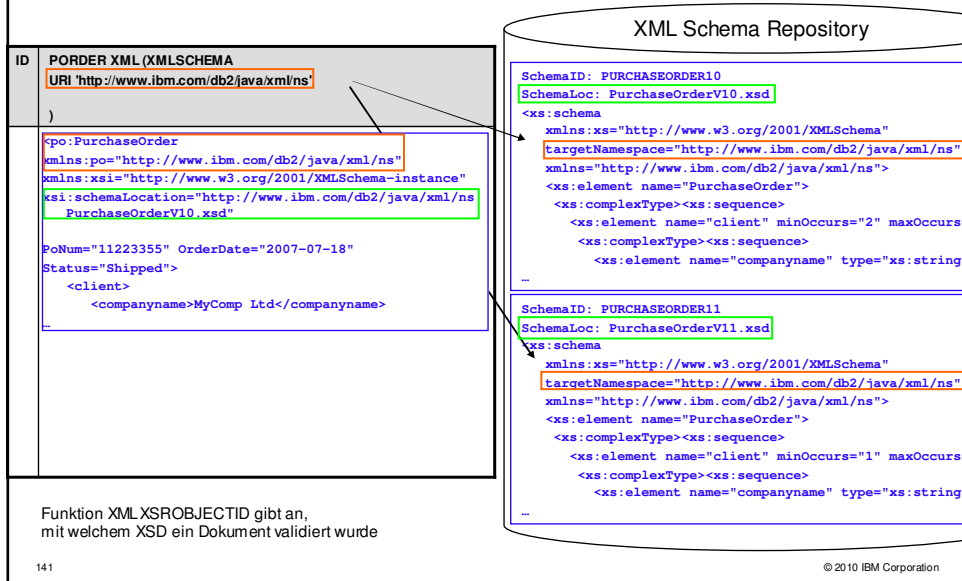
- Beispiel XSD (registriert mit SchemaLocation „PurchaseOrderV10.xsd“)

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.ibm.com/db2/java/xml/ns"
            xmlns="http://www.ibm.com/db2/java/xml/ns">
  <xs:element name="PurchaseOrder">
    <xs:complexType><xs:sequence>
      <xs:element name="client" minOccurs="1" maxOccurs="1">
        <xs:complexType><xs:sequence>
          <xs:element name="companyname" type="xs:string" minOccurs="1"/>
        ...
```

140

© 2010 IBM Corporation

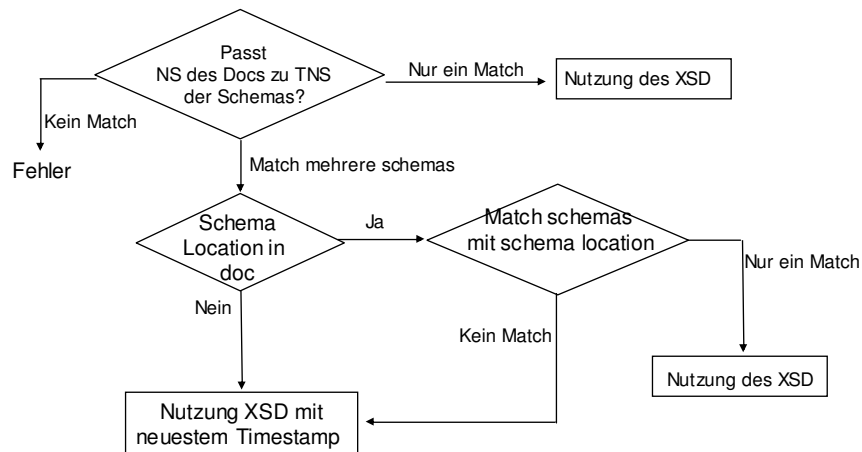
## Schema Validierung per type modifier



141

© 2010 IBM Corporation

## Schema Auswahl

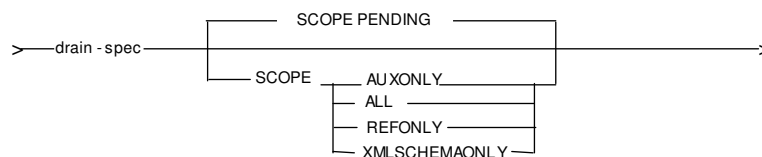


142

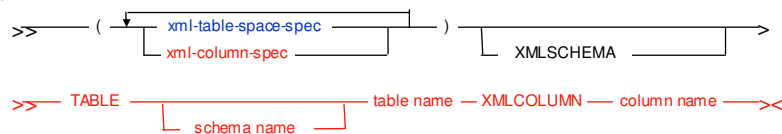
© 2010 IBM Corporation

## DB2 10 CHECK DATA

- Konsistenzprüfung NODEID Index
- Konsistenzprüfung der XML Dokumente
- Schemavalidierung für Spalten mit XML type modifier
- Syntax SCOPE Section



- Syntax INCLUDE XML TABLESPACES



144

© 2010 IBM Corporation

## Binary XML

- In V10 wird ein eigenes Format Binary XML eingeführt
- Genutzt bei
  - UNLOAD / LOAD
  - Zwischen DB2 Servern und Anwendungen
    - DRDA Protokoll
    - CLI/ODBC, JDBC, SQLJ

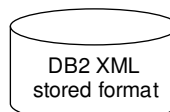
- Vorteile (erste Messungen)
  - 17 – 46 % kleiner
  - 9 – 30 % CPU Einsparung bei Insert
  - 8 and 50 % schneller (elapsed)

### Binary XML

```
....X.....X.person.}..X.firstName.~...U.SmithzT.X.nickName....U.JoeyzzZ
00005C300000050767766870050667774666870005667675050666646668000504667775
03008AB510002860523FE7D008969234E1D57E0053D948A4088E93BE1D5800054AF59AAA
```

### Textual XML

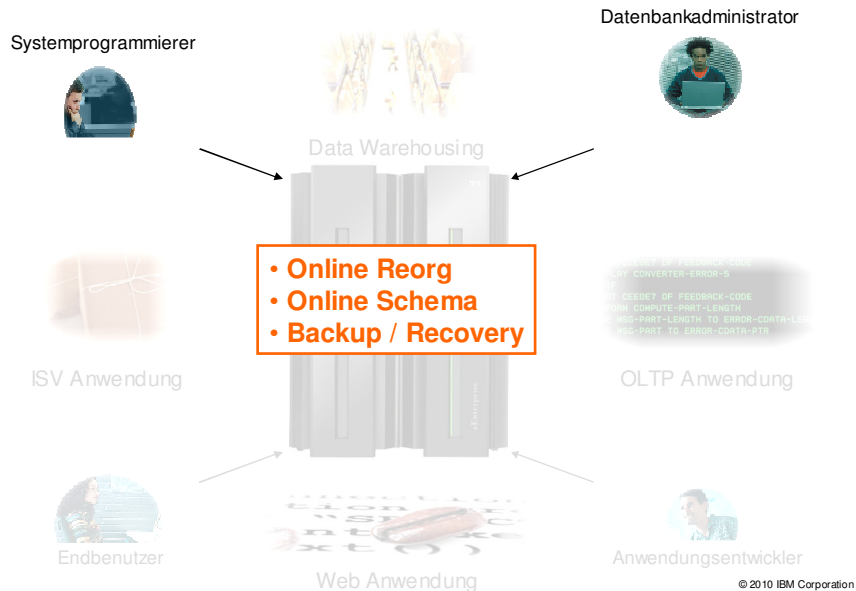
```
<person>
  <firstName>Joe</firstName>
  <lastName>Smith</lastName>
  <nickName>Joey</nickName>
</person>
```



## Agenda

- 09:00 Einführung
- 09:15 Migration & Vorteile nach der Migration
- 10:15 Neue Funktionen zur Qualitätssicherung
- 10:45 Pause
- 11:00 Anwendungsoptimierung
- 11:35 Neue Anforderungen an IT-Sicherheit
- 12:15 Mittagspause
- 13:00 DB2 auf Zeitreise
- 14:00 Pause
- 14:15 Erweiterungen von OLTP bis Warehousing
- 15:45 Abschluss
- 16:00 Veranstaltungsende

## DB2 V10: Utilities



## Utility Neuerungen: Online Reorg

- FORCE option

- Lesende Transaktionen zu canceln (READERS)
- Und auch schreibende Transaktionen zu canceln (ALL)
- Oder den bestehenden Default beizubehalten (NONE)
- Bitte beachten:
  - Schlägt nur zu bei dem letzten Drain Versuch
  - Sperrende Transaktion muß während des Drainprozesses im DB2 aktiv werden (PM31243)
  - Beide Online Reorg und Transaktion können verlieren, also Online Reorg bekommt x'00C200EA' und Transaktion auch 04E

## Utility Neuerungen: Online Reorg

- **AUX option**
  - LOB Objekte werden automatisch einbezogen
  - AUX YES ist Default, falls (LOB Columns vorausgesetzt)
    - ein ganzer PBG reorganisiert wird
    - REBALANCE für einen partitionierten Tablespace spezifiziert wird
    - ein SQL ALTER Statement die Partitionengrenzen verschoben hat
    - DISCARD für einen partitionierten Tablespace spezifiziert ist
- **Mehrfache Partitionsangaben möglich**
  - PART(1, 3:4, 8:10)
  - Auch in V9 mit PK87762

## Utility Neuerungen: Online Reorg

- **Inline LOB Unterstützung nach Alter LOB Column mit Inline Length**
- **Für alle Catalog und Directory Objekte**
  - Ausnahme: SYSUTILX
- **REORPending wird berichtet**
  - Alter Limitkey !
- **Rowformat Option: BRF oder RRF**
  - V9: PK85881
- **Statistik Update raus aus der SWITCH Phase**

## Utility Neuerungen: Runstats

- Use Profile:
  - Profile können spezifiziert werden, sind gespeichert in SYSIBM.SYSTABLES\_PROFILES Tabelle
- Set Profile
  - Setzt das Profil aus dem jetzt laufenden Runstats
  - Oder From Existing Stats
- Update/Delete Profile

## Runstats: Profile

```
DSNU050I      336 04:03:20.15 DSNUGUTC - RUNSTATS TABLESPACE JAVADB01.JAVATS01
TABLE(JAVATB01) USE PROFILE REPORT YES
DSNU1361I $ 336 04:03:20.15 DSNUGPRF - THE STATS PROFILE WITH
STATTIME = 2010-12-02-04.01.15.313562 FOR TABLE JAVATB01 HAS BEEN USED
DSNU1368I      336 04:03:20.15 DSNUGPRB - PARSING STATS PROFILE FOR TABLE JAVATB01
DSNU1369I      336 04:03:20.16 DSNUGPRB - PARSING STATS PROFILE FOR TABLE JAVATB01 COMPLETED
DSNU613I $ 336 04:03:21.69 DSNUSUTP - SYSTABLEPART CATALOG STATISTICS FOR
JAVADB01.JAVATS01 PARTITION 1
```

```
DSNU050I      336 04:01:15.29 DSNUGUTC - RUNSTATS TABLESPACE JAVADB01.JAVATS01 TABLE(JAVATB01)
SET PROFILE FROM EXISTING STATS REPORT YES
DSNU1357I $ 336 04:01:15.48 DSNUGPRF - THE STATS PROFILE FOR TABLE JAVATB01 HAS BEEN
SET IN SYSTABLES_PROFILES.
DSNU610I $ 336 04:01:15.48 DSNUGPRF - SYSTABLES_PROFILES CATALOG UPDATE FOR
SYSADM.JAVATB01 SUCCESSFUL
```

## Online Schema Evolution

- SQL Alter auf Tablespace Attribute wie DSSIZE
- Dies ist ein „Pending Alter“ im Gegensatz zu einem „Immediate Alter“
  - Wird im DB2 Catalog zwischengespeichert:
    - Neue Catalog Tabelle SYSIBM.SYSPENDINGDDL (mit Original ALTER Statement) und SYSIBM.SYSPENDINGOBJECTS
  - Angezeigt wird ein „Pending Alter“ durch den Status AREOR auf den Objekten (AREO\* ist für „Immediate Alter“), SQLcode+610
  - „Immediate Alter“ und dann „Pending Alter“ ist erlaubt
  - „Pending Alter“ und dann „Immediate Alter“ ist nicht erlaubt, z.B. SQLcode-20385, Reason 2

## Online Schema Evolution

- Änderungen können kumuliert werden,
  - z.B. Segmented Tablespace
    - DSSIZE Änderung ist nicht erlaubt, SQLcode-650, Reason 7
    - DSSIZE Änderung ist aber erlaubt, falls vorher Änderung auf UTS gemacht wurde (kann auch noch Pending sein)
- Änderungen können wieder zurückgenommen werden
  - Alter Tablespace Drop Pending Changes
  - Allerdings bleibt AREOR stehen



## Online Schema Evolution

- Online Reorg (Change+Reference) materialisiert diese Alters
  - Bedingungen:
    - Alle Partitionen müssen adressiert werden
    - Fastswitch Yes
- Bitte beachten: Statistik Update
  - Default ist Table All Index All Update All History All

## Online Schema Evolution: Online Reorg

```

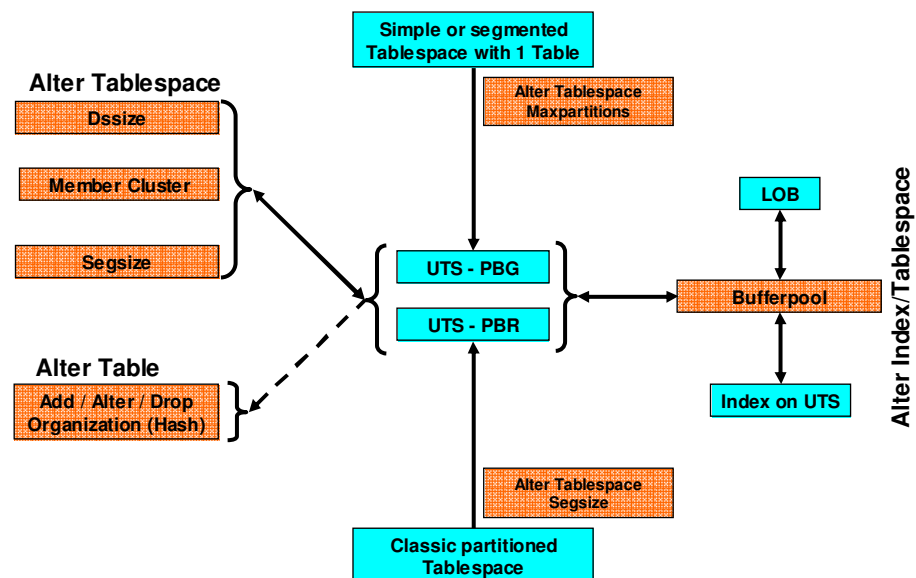
REORG TABLESPACE JAVADB01.JAVATS01 SHRLEVEL REFERENCE COPYDDN(TTTT)
...
DSNU387I      341 03:49:18.24 DSNURSWT - SWITCH PHASE COMPLETE, ELAPSED TIME = 00:00:01
DSNU428I      341 03:49:18.24 DSNURSWT - DB2 IMAGE COPY SUCCESSFUL FOR TABLESPACE
JAVADB01.JAVATS01
DSNU1163I S 341 03:49:17.04 DSNUSFAN - APPLYING PENDING DEFINITION CHANGES COMPLETE FOR
JAVADB01.JAVATS01
DSNU1166I S 341 03:49:17.05 DSNURSWD - SOME PARTITION STATISTICS MAY HAVE BECOME OBSOLETE ON
JAVADB01.JAVATS01
DSNU610I $ 341 03:49:18.69 DSNUSUTP - SYSTABLEPART CATALOG UPDATE FOR JAVADB01.JAVATS01 SUCCESSFUL
DSNU610I $ 341 03:49:19.46 DSNUSUPT - SYSTABSTATS CATALOG UPDATE FOR SYSADM.JAVATB01 SUCCESSFUL
DSNU610I $ 341 03:49:19.47 DSNUSUPC - SYSCOLSTATS CATALOG UPDATE FOR SYSADM.JAVATB01 SUCCESSFUL
DSNU610I $ 341 03:49:19.47 DSNUSUTB - SYSTABLES CATALOG UPDATE FOR SYSADM.JAVATB01 SUCCESSFUL
DSNU610I $ 341 03:49:19.48 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR SYSADM.JAVATB01 SUCCESSFUL
DSNU610I $ 341 03:49:19.48 DSNUSUTS - SYSTABLESPACE CATALOG UPDATE FOR JAVADB01.JAVATS01 SUCCESSFUL
DSNU610I $ 341 03:49:19.61 DSNUSUITP - SYSINDEXPART CATALOG UPDATE FOR SYSADM.JAVAIX SUCCESSFUL
DSNU610I $ 341 03:49:20.03 DSNUSUCO - SYSCOLUMNS CATALOG UPDATE FOR SYSADM.JAVAIX SUCCESSFUL
DSNU610I $ 341 03:49:20.03 DSNUSUIX - SYSINDEXES CATALOG UPDATE FOR SYSADM.JAVAIX SUCCESSFUL
DSNU620I $ 341 03:49:20.79 DSNUSEOF - RUNSTATS CATALOG TIMESTAMP = 2010-12-07-03.49.14.684533
DSNU010I      341 03:49:22.35 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=4
  
```

## Online Schema Evolution

- Nach Materialisierung ist Point in time Recovery nicht erlaubt, da Strukturänderungen nicht in Catalog/Directory mit zurückgenommen werden.
  - Unload von Image Copy funktioniert

```
REPORT RECOVERY TABLESPACE JAVADB01.JAVATS01
...
TIMESTAMP = 2010-12-07-03.48.00.784439, IC TYPE = #F#,
DEV TYPE = 3390 , IC BACK = ,
LOW DSNUM = 0001, HIGH DSNUM = 0001, OLDEST VERSION =
JOBNAME = SYSADM2 , AUTHID = SYSADM , COPYPA
NPAGESF = 2.3E+01 , CPAGES
DSNAME = DSNC910.IC.JAVADB01.JAVATS01.A000.F.G0001V00
```

## Online Schema Evolution



## Utility Neuerungen: FlashCopy

- FlashCopy für Image Copies (auch Inline) auf Dataset Level
  - Alle Utilities werden unterstützt
    - Unload mittels Umweg über CopyToCopy
  - Template bei zPARM oder in den Utilities
    - Da VSAM Dataset, sind keine Parameter nötig außer Name
  - Keine incrementelle Kopie möglich
  - Consistent Parameter um eine konsistente Kopie zu bekommen anstelle einer „Fuzzy“ Kopie

## FlashCopy (1)

```

COPY LIST LLLL SHRLEVEL CHANGE FLASHCOPY CONSISTENT
PROCESSING LIST ITEM: TABLESPACE JAVADB01.JAVATS01
...
ADR030I (SCH)-PRIME(01), DCB VALUES HAVE BEEN MODIFIED FOR SYSPRINT
COPY DATASET(INCLUDE( -
  DSNADB2.DSNDBC.JAVADB01.JAVATS01.I0001.A001 )) -
  RENAMEU( -
    (DSNADB2.DSNDBC.JAVADB01.JAVATS01.I0001.A001 , -
    HLQ.IC.JAVADB01.JAVATS01.N00001 )) -
  REFUNC ALLDATA(*) ALLEXCP CANCELERROR SHARE -
  WRITECHECK TOLERATE(ENQF)
...
ADR454I (001)-DDDS (01), THE FOLLOWING DATA SETS WERE SUCCESSFULLY PROCESSED
      DSNADB2.DSNDBC.JAVADB01.JAVATS01.I0001.A001

```

## FlashCopy (2)

```

DSNU1540I  336 03:23:17.83 DSNUGFCD - CONSISTENCY PROCESSING FOR FLASHCOPY IS STARTED
DSNU578I  $ 336 03:23:17.87 DSNUCALZ - SYSLGRNX INFORMATION FOR MEMBER VA1A
DSNU513I  $ 336 03:23:17.87 DSNUCALZ - FLASHCOPY WITH CONSISTENCY LOG APPLY RANGE IS
RBA 00001D393B17 LRSN C6F7BEDCAD62 TO RBA 000000000000 LRSN C6F7BEEC1478
DSNU1511I $ 336 03:23:17.88 DSNUCALZ - FAST LOG APPLY WAS NOT USED FOR FLASHCOPY WITH CONSISTENCY
DSNU1510I  336 03:23:17.88 DSNUCBLA - LOG APPLY PHASE COMPLETE, ELAPSED TIME = 00:00:00
DSNU1550I  $ 336 03:23:17.88 DSNUCALC - LOGCSR IS STARTED FOR MEMBER VA1A,
PRIOR CHECKPOINT RBA = X'00001D387CEB'
DSNU1551I  $ 336 03:23:17.88 DSNUCALC - LOGCSR IS FINISHED FOR MEMBER VA1A, ELAPSED TIME = 00:00:00
DSNU1552I  $ 336 03:23:17.88 DSNUCALC - LOGCSR PHASE COMPLETE, ELAPSED TIME = 00:00:00
DSNU1553I  $ 336 03:23:17.88 DSNUCALC - FLASHCOPY WITH CONSISTENCY DETECTS THE FOLLOWING ACTIVE URS:
INFLIGHT = 1, INABORT = 0, INDOUBT = 0, POSTPONED ABORT = 0 COMMITTED = 0, ABORTED = 0

      MEM  T  CONNID  CORRID  AUTHID  PLAN  S  URID  DATE  TIME
      VA1A  B  TSO    SYSADM  SYSADM  DSNESPCS F 00001D39B241 2010-12-02 11.23.09
      DBNAME SPACENAME DBID/PSID PART  RBA
      JAVADB01 JAVATS01 011A/0002 0001 C6F7BEE46667

DSNU1554I  $ 336 03:23:17.89 DSNUCALU - LOGUNDO IS STARTED FOR MEMBER VA1A
DSNU1556I  $ 336 03:23:17.89 DSNUCALU - LOGUNDO IS FINISHED FOR MEMBER VA1A, ELAPSED TIME = 00:00:00
DSNU1557I  $ 336 03:23:17.89 DSNUCALU - LOGUNDO PHASE COMPLETE, ELAPSED TIME = 00:00:00
DSNU1541I  336 03:23:17.89 DSNUGFCD - CONSISTENCY PROCESSING FOR FLASHCOPY IS FINISHED,
ELAPSED TIME = 00:00:00
DSNU010I  336 03:23:17.91 DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0

```

## Utility Neuerungen: Point in Time Recovery

- BACKOUT Option im Gegensatz zu Recovery mit Image Copy und den Logsätzen
- VERIFYSET Option
- ENFORCE Option

## Utility Neuerungen: Verifyset Yes

```

-DIS DB(JAVADB01) SP(*) LIMIT(*)
DSNT360I $ *****
...
NAME      TYPE PART  STATUS
-----
JAVATS01 TS      0001 RW
      -THRU      0003
JAVATS01 TS              RW
JAVATSA1 LS              RW ** not defined **
JAVATSA2 LS              RW
JAVATSA3 LS              RW ** not defined **
JAVAIX01 IX      L0001 RW
      -THRU      0003
JAVAIX01 IX      L*    RW
JAVAIXA1 IX              RW
JAVAIXA2 IX              RW
JAVAIXA3 IX              RW
***** DISPLAY OF DATABASE JAVADB01

```

RECOVER TABLESPACE JAVADB01.JAVATS01  
TOLOGPOINT X'C6E774FAB4D4' VERIFYSET YES  
...  
DSNU1316I \$ 319 06:12:20.18 DSNUCAIN - THE FOLLOWING  
TABLESPACES ARE MISSING FROM THE RECOVERY LIST  
JAVADB01.JAVATSA2

RECOVER TABLESPACE JAVADB01.JAVATS01  
TABLESPACE JAVADB01.JAVATSA2  
TOLOGPOINT X'C6E774FAB4D4' VERIFYSET YES  
...  
DSNU535I \$ 319 06:17:02.52 DSNUCATM - FOLLOWING  
TABLESPACES RECOVERED TO A CONSISTENT POINT  
JAVADB01.JAVATS01  
JAVADB01.JAVATSA1  
JAVADB01.JAVATSA2  
JAVADB01.JAVATSA3

163

## Utility Neuerungen: Verifyset No und Enforce Yes

```

RECOVER TABLESPACE JAVADB01.JAVATS01
TOLOGPOINT X'C6E774FAB4D4'
VERIFYSET NO ENFORCE YES
...
DSNU815I $ 319 06:23:04.40 DSNUGSRX - TABLE SPACE
JAVADB01.JAVATS01 IS IN AUX CHECK PENDING STATE

```

```

-DIS DB(JAVADB01) SP(*) LIMIT(*)
DSNT360I $ *****
...
NAME      TYPE PART  STATUS
-----
JAVATS01 TS      0001 RW,ACHKP
      -THRU      0003
JAVATS01 TS              RW,ACHKP
JAVATSA1 LS              RW
JAVATSA2 LS              RW
JAVATSA3 LS              RW

```

```

CHECK DATA TABLESPACE JAVADB01.JAVATS01 SCOPE ALL
...
DSNU817I 323 05:02:08.43 DSNUKERK - TABLE=SYSADM.JAVATB01
COLUMN=COL3 IS COMPLETELY INLINE BUT ENTRY IN INDEX
SYSADM.JAVAIXA2 FOUND
ROWID=X'DC89697DEE763DF081C2017DD98801000000100000402'
VERSION=X'0001'

```

164

## Utility Neuerungen: Set CHKP flag

```

CHECK DATA TABLESPACE JAVADB01.JAVATS02 SCOPE ALL
...
DSNU733I  327 03:29:36.91 DSNUKERR - ROW (RID=X'0000000201')
HAS NO PARENT FOR SYSADM.JAVATB01C FOREIGN KEY CONSTRAINT JAVATB01
DSNU739I  327 03:29:36.91 DSNUKDAT - CHECK TABLE SYSADM.JAVATB01C COMPLETE,
ELAPSED TIME=00:00:00
DSNU561I  S 327 03:29:36.92 DSNUGSRX - TABLESPACE JAVADB01.JAVATS02 PARTITION 1
IS IN CHECK PENDING
DSNU749I  327 03:29:36.92 DSNUK001 - CHECK DATA COMPLETE,ELAPSED TIME=00:00:00
DSNU010I  327 03:29:36.93 DSNUGBAC - UTILITY EXECUTION COMPLETE,
HIGHEST RETURN CODE=4
  
```

**zPARM CHECK\_SetCHKP=YES/NO**

**zPARM betrifft nur Check Data und Check LOB, aber nicht Check Index**

## DB2 V10: Performance und Automation

Systemprogrammierer

Datenbankadministrator

Data Warehousing

- Index Erweiterungen
- Auto Stats
- Hash Access

ISV Anwendung

OLTP Anwendung

Endbenutzer

Web Anwendung

Anwendungsentwickler

## Mehr Flexibilität bei der Datenkomprimierung

- Tablespace mit COMPRESS YES
- Datenkomprimierung benötigt ein Dictionary
- Heute in V9
  - Dictionary Erstellung
    - LOAD, REORG
    - LOAD COPYDICTIONARY 1 ... PART 3
  - Problem: keine regelmäßige Nutzung von REORG/LOAD
  - Anforderung: Anwendung fügt Daten hinzu!
- V10 Neuerung bei Dictionary Erstellung
  - MERGE
  - INSERT
  - LOAD .... SHRLEVEL CHANGE
  - Internal threshold
  - Asynchroner Dictionary Aufbau
    - Inserts sind unkomprimiert

```
DSNU241I @ DSNUZLCR - DICTIONARY WITH
4096 ENTRIES HAS BEEN SUCCESSFULLY
BUILT FROM 824 ROWS FOR
TABLE SPACE LI864DB.LI864TS, PARTITION 1
```

## Mehr OLAP Funktionen

- Abfragen mit kumulativer Summenbildung und fortlaufender Durchschnittsberechnung
  - In Select-Liste, Expressions, Order by clause
- Umsatz-Tabelle mit folgenden Inhalten:

GEBIET	MONAT	UMSATZ
OST	200910	10.00
WEST	200910	8.00
OST	200911	4.00
WEST	200911	12.00
OST	200912	10.00
WEST	200912	7.00
OST	201001	7.00
WEST	201001	11.00
OST	201002	9.00
WEST	201002	7.00

## Beispiel „fortlaufende Durchschnittsberechnung“

```
SELECT GEBIET, MONAT, UMSATZ, AVG(UMSATZ)
OVER (PARTITION BY GEBIET ORDER BY MONAT) AS MOVING_AVG
FROM UMSATZ_TB
```

GEBIET	MONAT	UMSATZ	MOVING_AVG
OST	200910	10.00	10.00
OST	200911	4.00	7.00
OST	200912	10.00	8.00
OST	201001	7.00	7.75
OST	201002	9.00	8.00
WEST	200910	8.00	8.00
WEST	200911	12.00	10.00
WEST	200912	7.00	9.00
WEST	201001	11.00	9.50
WEST	201002	7.00	9.00

## Beispiel „fortlaufende Durchschnittsberechnung“

```
SELECT GEBIET, MONAT, UMSATZ, AVG(UMSATZ)
OVER (PARTITION BY GEBIET ORDER BY MONAT
ROWS 2 PRECEDING) AS MOVING_AVG
FROM UMSATZ_TB
```

GEBIET	MONAT	UMSATZ	MOVING_AVG
OST	200910	10.00	10.00
OST	200911	4.00	7.00
OST	200912	10.00	8.00
OST	201001	7.00	7.00
OST	201002	9.00	8.66
WEST	200910	8.00	8.00
WEST	200911	12.00	10.00
WEST	200912	7.00	9.00
WEST	201001	11.00	10.00
WEST	201002	7.00	8.33



## Beispiel „kumulative Summenbildung“

```
SELECT GEBIET, MONAT, UMSATZ, SUM(UMSATZ)
OVER (PARTITION BY GEBIET ORDER BY MONAT
      ROWS UNBOUNDED PRECEDING) AS CUMULATIVE_SUM
FROM UMSATZ_TB
```

GEBIET	MONAT	UMSATZ	KUMUL_SUM
OST	200910	10.00	10.00
OST	200911	4.00	14.00
OST	200912	10.00	24.00
OST	201001	7.00	31.00
OST	201002	9.00	40.00
WEST	200910	8.00	8.00
WEST	200911	12.00	20.00
WEST	200912	7.00	27.00
WEST	201001	11.00	38.00
WEST	201002	7.00	45.00

## Parallelität beim Index Update

- Tabellen mit vielen Indexes
- Heute in V9
  - Sequentieller Index update
  - Performance / elapsed time Nachteil
- V10 Parallel Index Update
  - I/O parallelism für non-clustered Indexes
    - > 2 Indexes (ohne clustered Index) pro Tabelle
  - zPARM: INDEX\_IO\_PARALLELISM = YES (default)
  - Reduzierte Index I/O wait time
  - Laufzeitvorteil
    - Abhängig von Bufferpool hit ratio
    - Monitoring über IFCID 358 zu I/O parallelism
    - bis zu 50% elapsed time Verbesserung gemessen (6 Indexes)
  - UTS & classic partitioned tablespace

## Weniger Indexes / weniger Ressourcen

- Unique Indexes beinhalten relevante Tabellenspalten zur Sicherstellung der Eindeutigkeit
- Weitere Indexes werden benötigt
  - Obermenge von Unique Indexes für Index only Zugriffe
- V10 Index Include Columns für Unique Indexes
  - Zusätzliche Spalten werden nicht zur Festlegung der Eindeutigkeit einbezogen
  - Performance
  - Vorteile durch Potential für weniger Indexes
    - Reduzierter Platzbedarf
    - Reduzierte Indexpflege → Performance Verbesserung

## Index Include Column - Beispiel

- Optimierung von Abfragen auf die KFZ Tabelle mit Vertrags-/Kundennr., Kasko
- In V9

```
CREATE UNIQUE INDEX I1 ON KFZ (VERTRAGSNR);  
CREATE INDEX I2 ON KFZ (VERTRAGSNR, KUNDENNR, KASKO);
```

- Optimierung in V10

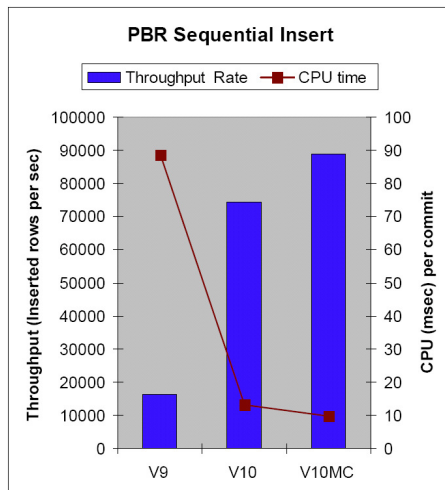
```
CREATE UNIQUE INDEX I1 ON KFZ (VERTRAGSNR) INCLUDE (KUNDENNR, KASKO);
```

- Alternativ

```
ALTER INDEX I1 ADD INCLUDE COLUMN (KUNDENNR); ...  
DROP INDEX I2;
```

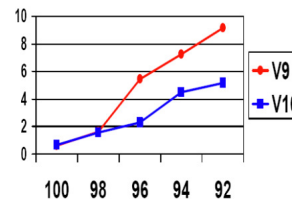
- Index I1 geht in rebuild-pending state

## Sequentielle Inserts & Data Scans



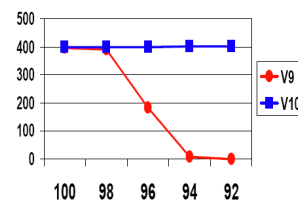
175

Query Time (seconds)



Cluster ratio

Dynamic Prefetch I/Os



Cluster ratio

© 2010 IBM Corporation

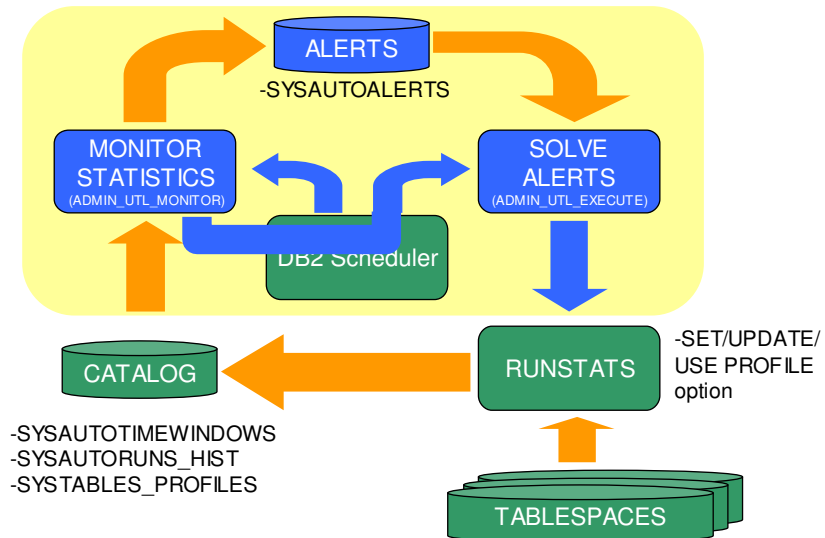
## DB2 in Synergie mit zEnterprise

- DB2 Performance, Skalierbarkeit
- z196
  - Schnellere CPUs, mehr Speicher
    - **V10 Erweiterungen**
      - In-Memory Bufferpools (PGSTEAL = NONE)
      - Virtual Storage Constraint & latch contention relief
  - Compression Verbesserungen
  - 192M L4 Cache
  - 1 MB page size Verbesserungen
    - **V10 nutzt erstmalig 1 MB fixed page frames**
      - Bufferpools mit PGFIX=YES
      - LFAREA Definition in IEASYSxx
    - Potential für CPU Einsparung (1-4% gemessen)
- V10 mit mehr zIIP Nutzung
  - Utilities (Runstats) und im I/O Bereich

176

© 2010 IBM Corporation

## AutoStats design



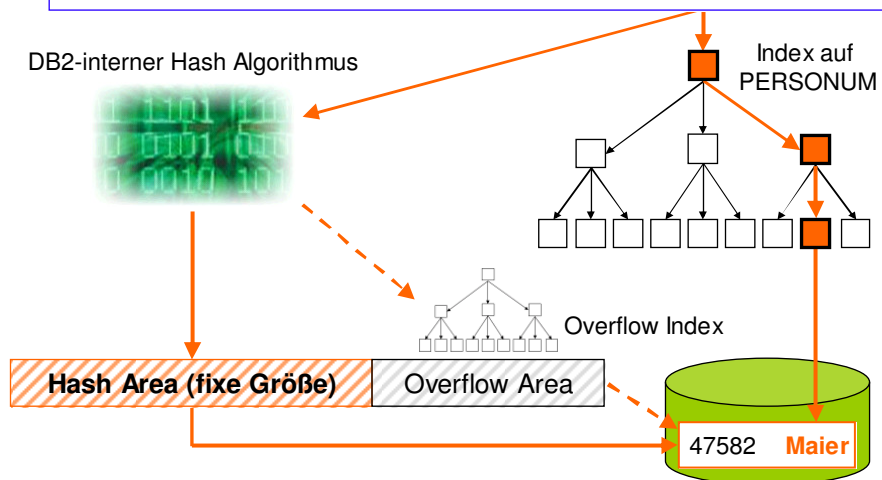
177

© 2010 IBM Corporation

## Index-Zugriffe sind nicht immer der effizienteste Weg

```
SELECT NACHNAME FROM OLTP.PERSONTB WHERE PERSONUM = 47582
```

DB2-interner Hash Algorithmus



178

© 2010 IBM Corporation

## Vorteile, Nachteile und Bedingungen für Hash Access

- Vorteile:
  - In Summe **weniger I/O-Operationen** notwendig
  - Bessere Antwortzeiten und geringerer CPU-Verbrauch
  - Evtl. DROP von unnötigen Indexes möglich
- Nachteile:
  - **Höherer Speicherplatzbedarf** (bis zu Faktor 2)
- Voraussetzungen:
  - **Universal Table Space mit Reordered Row Format**
  - SELECT mit **Equal Predicate** (oder IN-List) auf alle Hash Key Columns mit Zugriff auf einzelne Rows

## Potentielle Anwendungsszenarien für Hash Access

- **Unique Key** ist vorhanden und wird genutzt
- **Tabellengröße** bekannt und relativ **statisch**
- **Viele Index Levels** vorhanden (mehr als 3)

Geeignete Beispielfälle



Versicherungspolizen  
Geldautomaten  
Stücklisten

Ungeeignete Beispielfälle



Kontobewegungen  
Bestellübersichten  
Kundenlisten

## Hashing einrichten per CREATE TABLE Statement

```
CREATE TABLE OLTP.PERSONTB
(PERSONUM INTEGER NOT NULL,
...)
ORGANIZE BY HASH UNIQUE
(PERSONNUM) HASH SPACE 1M;
```

- DB2 erstellt **automatisch Overflow Index** (dient ggf. auch zur Sicherstellung der Uniqueness des Primary Keys)
- HASH SPACE ~ geschätzte Größe der Tabelle
- Bis zu 64 Columns:
  - Definiert als NOT NULL
  - Summe der Length Attribute höchstens 255
  - Keine LOB, DECFLOAT oder XML Columns

## Hashing einrichten per ALTER TABLE Statement

```
ALTER TABLE OLTP.PERSONTB
ADD ORGANIZE BY HASH UNIQUE
(PERSNUM) HASH SPACE 2G;
```

- Logische Änderung sofort zur Sicherstellung der Uniqueness
- Physische Reorganisation erst nach REORG:
  - Table Space steht auf Advisory REORG Pending (**AREOR**)
  - Overflow Index steht auf REBUILD Pending (**PSRBD**)

NAME	TYPE	PART	STATUS
OLTP.TS01	TS	0001	RW,AREOR
OLTP.TS01	TS		RW,AREOR
PERSONTB	IX	L0001	RW
PERSONTB	IX	L*	RW

## Nachträgliche Änderungen an der Hash Organization

- Ändern des **HASH SPACE** Werts:
  - ALTER ORGANIZATION SET HASH SPACE (*integer*) K / M / G
  - ALTER TABLE ALTER PARTITION (X) HASH SPACE (*integer*)
  - Neuer Wert zieht erst nach REORG (keine Statusänderung)
- Ändern der **Hash Key Columns**:
  - Zunächst ALTER DROP ORGANIZATION nötig
  - Danach neuer ALTER ADD ...
- **Löschen** der Hash Organization:

```
ALTER TABLE OLTP.PERSONTB DROP ORGANIZATION;
```

- Funktioniert nicht, wenn Primary Key Constraint existiert und der Overflow Index als entsprechender Index dient
- Setzt den Table Space auf REORG Pending (**REORG**) und löscht den Overflow Index

## Hinweise zur Größenabschätzung für HASH SIZE

- HASH SIZE hängt von der Row und Page Size ab und führt zu schlechterer Performance (wegen übermäßiger Overflow Index Nutzung) falls auf lange Sicht zu klein
- Neuer **REORG** Parameter **AUTOESTSPACE** lässt DB2 den HASH SPACE Wert mit Hilfe der Real Time Statistics (RTS) bestimmen (Default ist YES)
- Vergleich von SYSINDEXSPACESTATS.TOTALENTRIES mit SYSTABLESPACESTATS.TOTALROWS.
  - **TOTALENTRIES** sollte kleiner als **10% von TOTALROWS** sein
- Vergleich von SYSTABLESPACESTATS.DATASIZE mit SYSTABLESPACE.HASHSPACE:
  - **HASHSPACE** sollte mind. **20% größer** sein als **DATASIZE**

## Prüfung der Nutzung von Hash Access beim Zugriff

- **PLAN\_TABLE** Column **ACCESSTYPE**:
  - **H** oder **HN** bedeutet Hash Access
  - **MH** bedeutet Nutzung des Overflow Index
- **SYSIBM.SYSTABLESPACESTATS**:
  - Column **HASHLASTUSED** gibt das Datum des letzten Hash Zugriff an
  - Column **REORGHASHACCESS** gibt an wie oft der Zugriff per Hash erfolgt ist
- **SYSIBM.SYSINDEXSPACESTATS**:
  - Column **REORGINDEXACCESS** gibt an wie oft der Overflow Index genutzt wurde

## Weitere Bemerkungen zur Nutzung von Hash Access

- **Kein UPDATE** auf Hash Key Werte möglich, stattdessen DELETE und INSERT nötig

THE UPDATE OPERATION IS INVALID BECAUSE THE CATALOG DESCRIPTION OF COLUMN OLTP.PERSONTB.PERSONUM INDICATES THAT IT CANNOT BE UPDATED. SQLCODE=-151

- **Hash Access** und **Clustering Indexes** sowie **Member Cluster** schließen sich aus (damit auch APPEND YES)
- Bei Multi-Row Fetch wird Hash Access nur genutzt, falls die Query eine IN Clause enthält, die mehrere Rows liefert
- Kein REORG auf Partition-Ebene möglich nach ALTER ADD HASH ORGANIZATION oder DROP ORGANIZATION



## DB2 V10: Was ist für Sie dabei ?

Systemprogrammierer



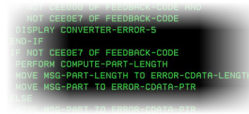
Datenbankadministrator



Data Warehousing



ISV Anwendung



OLTP Anwendung



Endbenutzer



Anwendungsentwickler

Web Anwendung



189

© 2010 IBM Corporation

## Agenda

- 09:00 Einführung
- 09:15 Migration & Vorteile nach der Migration
- 10:15 Neue Funktionen zur Qualitätssicherung
- 10:45 Pause
- 11:00 Anwendungsoptimierung
- 11:35 Neue Anforderungen an IT-Sicherheit
- 12:15 Mittagspause
- 13:00 DB2 auf Zeitreise
- 14:00 Pause
- 14:15 Erweiterungen von OLTP bis Warehousing
- 15:45 Abschluss
- 16:00 Veranstaltungsende

190

© 2010 IBM Corporation

DB2 V10 – Im Einsatz, wo andere längst aufgeben...



## IBM Data Studio für Datenbank-Admins, Anwendungsentw., ... kostenfreie Komponente!

**SQL Queries erstellen**

**Mit XML Files arbeiten**

**Datenbank-Objekte browsen**

**Daten browsen**

Aktuell basierend auf Eclipse 3.4.2 und dem Data Tools Platform (DTP) Framework

192

© 2010 IBM Corporation

DB2 V10 – Im Einsatz, wo andere längst aufgeben...



## DB2 Tools für z/OS

*Alles Neu mit DB2 V10!*

**DB2 Utilities Suite 10** drives down costs with autonomics, page sampling and further offloads processing to **zIIPs** and **FlashCopy**. Developed in conjunction with DB2 10 to provide maximum data integrity and exploit all new functions out of the box.

**Tivoli OMEGAMON XE for DB2 Performance Expert 5.1** extends its *insight into distributed workloads* and offers a robust infrastructure to support DB2 10 subsystem consolidation, with lower monitoring overhead. The recommended performance monitor of **DB2 10!**

**QMF 10** delivers built-in visualizations and reports that dramatically extend the value to end users. A new metadata layer simplifies the process to understand and create reports.

**DB2 Administration Tool 10.1** extends the value of DB2 10 with new capabilities that allow DBAs to quickly exploit DB2 10 features like schema evolution. Reduces the overhead of many routine tasks.

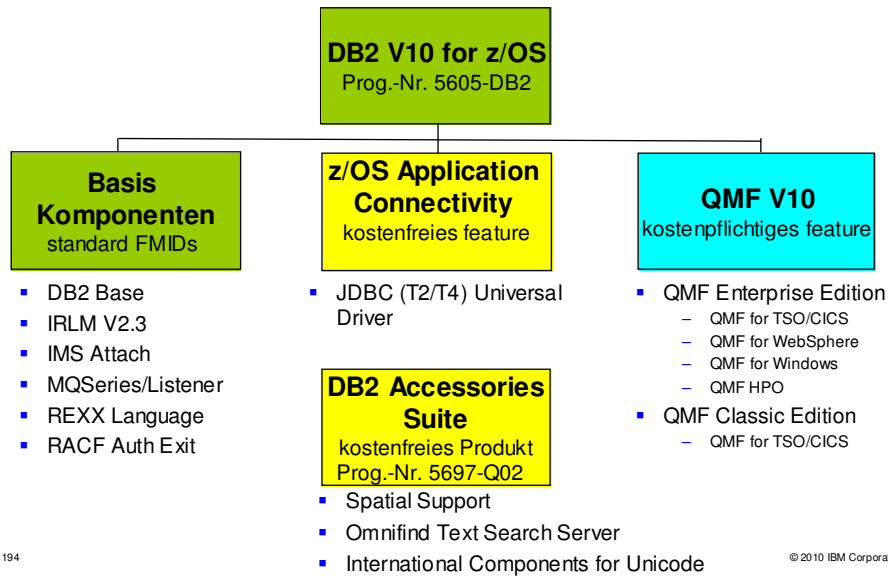
► Weitere Tools bieten DB2 10 Unterstützung per PTF ab GA

Detaillierte Übersicht: <http://www-01.ibm.com/support/docview.wss?uid=swg21409518>

193

© 2010 IBM Corporation

## DB2 V10 Packaging



## IBM V10 Schulungen und Migrationsangebote

- IBM Serviceangebot**
  - DB2 Migration Planning Workshop
  - Migrationsunterstützung
    - V8/V9 nach V10
- IBM Education**
  - Training WebSite: [www.ibm.com/training/de](http://www.ibm.com/training/de)  
→ Information Management
  - Newsletter [db2educ@de.ibm.com](mailto:db2educ@de.ibm.com)
  - DB2 V10 Kurs  
CV31xxxx – 4 Tage



THINK BIG!

# IBM Information Management Forum

26. und 27. April 2012 im Hotel Berlin

**Infos in XXL!**

Beim IBM Total Solution Event for System z „Think Big!“ vom 26. bis 27. April 2012 und mitten in Berlin

<http://www-05.ibm.com/de/events/im-forum>

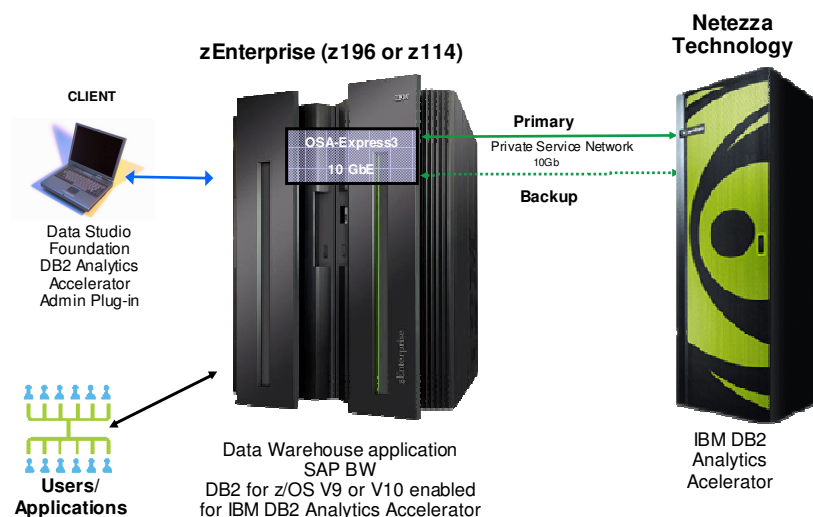
<http://www.ibm.com/de/events/tsez/>

**Frühbucherrabatt**  
bis zum 15. März:  
199 € für 2 Tage.  
► Jetzt sichern!

**From Zero to z Hero - Workshops**

**pro Quartal geplant (voraussichtlich):**  
14.02.-16.02. Ehningen  
08.05.-10.05. Frankfurt  
18.09.-20.09. Düsseldorf  
06.11.-08.11. Ehningen  
<http://www.ibm.com/de/events/fromzerotozhero/2012/>

## IBM DB2 Analytics Accelerator



## DB2 V10 – Verfügbarkeit / Informationen im Web

- DB2 Daten
  - V10 GA / Allgemeine Verfügbarkeit: **22. Oktober 2010**
  - V8 End-of-Service: **30. April 2012**  
<http://www-01.ibm.com/software/data/support/libegeta>
- DB2 V10 Web page  
<http://www.ibm.com/software/data/db2/v10/index.html>  
<http://www.ibm.com/software/data/db2/v10/index.html>
- IBM DB2 Information Management Tools V10 Compatibility  
[http://www-01.ibm.com/support/docview\\_wcs74d-wsg2469536](http://www-01.ibm.com/support/docview_wcs74d-wsg2469536)
- DB2 Information Center  
[http://publib.boulder.ibm.com/infocenter/db2u/infocenter/index.jsp?topic=/com.ibm.db2.zos/doc/ibm\\_zos\\_rbm.htm](http://publib.boulder.ibm.com/infocenter/db2u/infocenter/index.jsp?topic=/com.ibm.db2.zos/doc/ibm_zos_rbm.htm)
- Redbooks / Whitepapers
  - Temporal Data Management in DB2 z/OS  
[http://publib.boulder.ibm.com/infocenter/db2u/infocenter/topicview.jsp?topic=/com.ibm.db2.zos/doc/ibm\\_zos\\_rbm.htm](http://publib.boulder.ibm.com/infocenter/db2u/infocenter/topicview.jsp?topic=/com.ibm.db2.zos/doc/ibm_zos_rbm.htm)
  - DB2 V10 – Proven, simplified and cost effective  
[http://publib.boulder.ibm.com/infocenter/db2u/infocenter/topicview.jsp?topic=/com.ibm.db2.zos/doc/ibm\\_zos\\_rbm.htm](http://publib.boulder.ibm.com/infocenter/db2u/infocenter/topicview.jsp?topic=/com.ibm.db2.zos/doc/ibm_zos_rbm.htm)
  - DB2 V10 for z/OS – Technical Overview  
<http://www.redbooks.ibm.com/redbooks/p247892.html>

## DB2 10 for z/OS – Im Einsatz, wo andere längst aufgeben...

