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

Disclaimer and Trademarks

Client Technical Professionals DB2 for z/OS

Christian Daser
Christian.Daser@de.ibm.com

Georg Kistenberger
kistenberger@de.ibm.com

Agenda

09:30	Einführung
09:45	Migration & Vorteile nach der Migration
10:45	Neue Funktionen zur Qualitätssicherung
11:15	Pause
11:30	Anwendungsoptimierung
12:00	Neue Anforderungen an IT-Sicherheit
12:45	Mittagspause
13:30	DB2 auf Zeitreise, Bi-Temporal Data
14:15	Erweiterungen von OLTP bis Warehousing
15:30	Pause
15:45	IBM DB2 Analytics Accelerator
16:00	Unicode in DB2 z/OS
16:15	Abschluss und Q&A
16:45	Veranstaltungsende

Performance & CPU Verbrauch

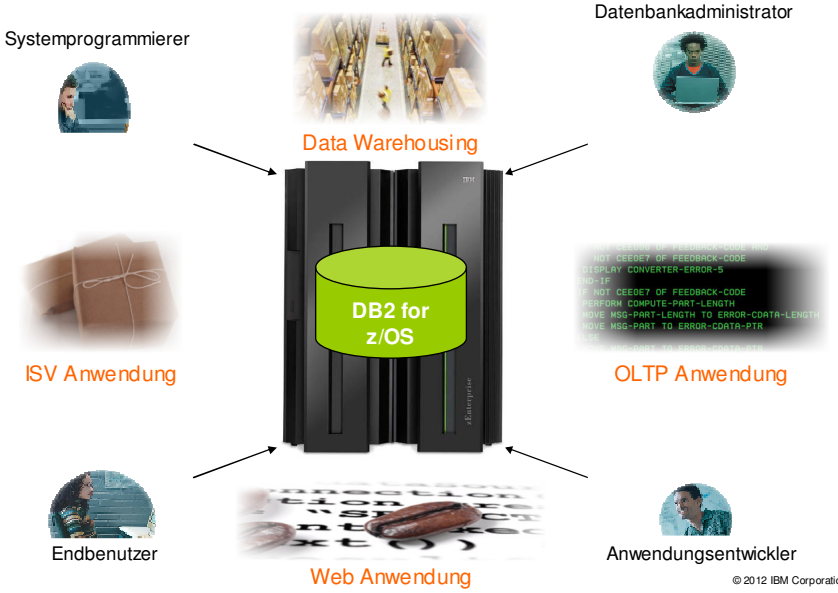
*The most dramatic achievement of DB2 10 comes from **reduced CPU usage**. Compared to previous versions, most customers can achieve out-of-the-box CPU savings of up to five to ten percent for traditional workloads, and up to 20 percent for **some specific[1] workloads**.*

*[1] … **Actual CPU reduction will, of course, vary** depending on the specific customer workload mix…*

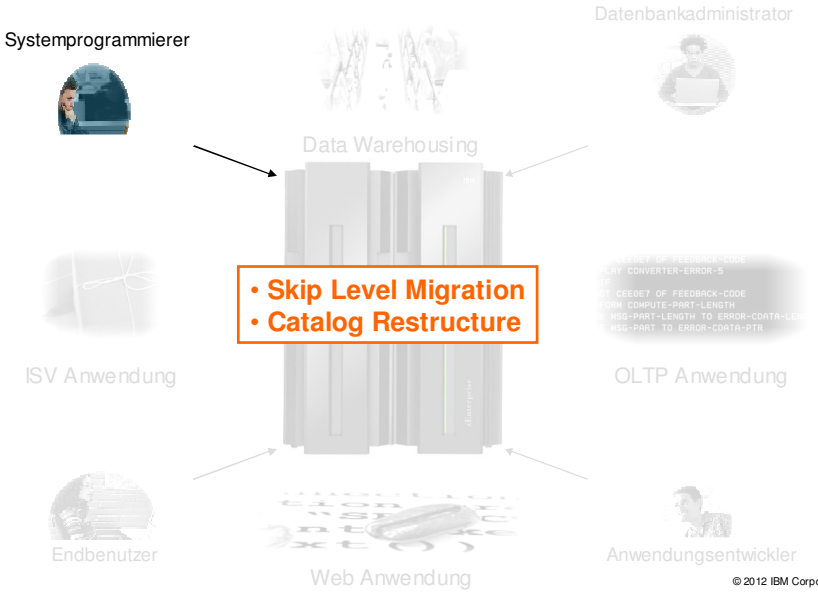
Current expectation is between 0% and 10% CPU savings

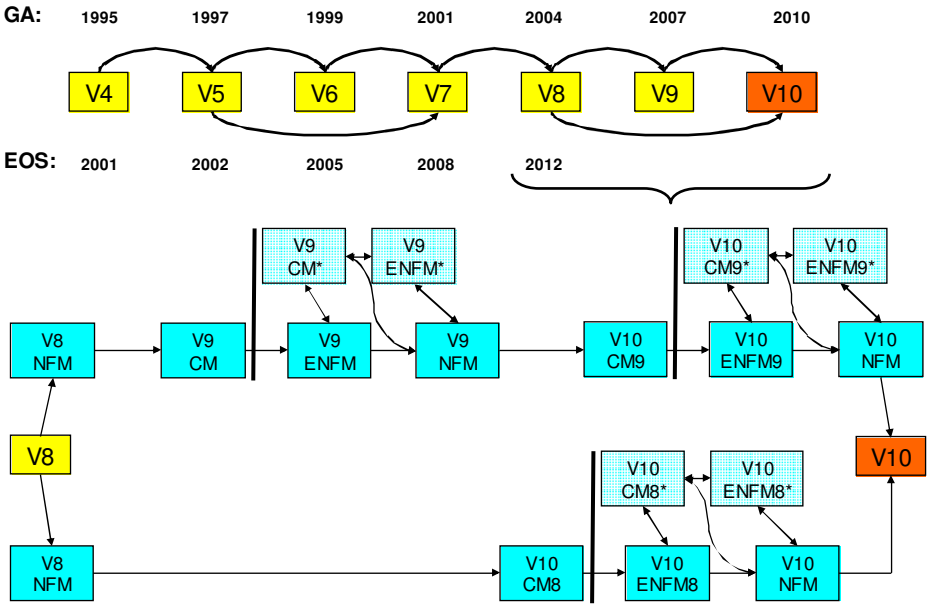
http://www.ibm.com/software/data/db2/zos/db2-10/

DB2 V10: Für jeden was dabei!



DB2 V10: Migration





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 minimum Release für V10 CM
  - 9.7 Fixpack 3a ist Basis für neue Funktionen
  - 9.7 Fixpack 6 ist empfohlen für neue Funktionen
- IMS V10, CICS V3.1, Websphere V6
- Optim (Data Studio, pureQuery, ...) ab 2.2
- File Manager for z/OS 12.1
- ...

PTF Stand für Migration

- Start V10 Migration mit aktueller Maintenance
  - Last RSU level + HIPERs + PE Korrekturen
- Fallback Toleration SPE 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)
  - RSM APAR OA35885 strongly recommended for DB2 10 plus DB2 z/OS. Corresponding DB2 support for RSM enhancements (PM24723)
    - Enables proper monitoring of REAL storage usage
    - Provides some protection against the system paging or running out of AUX storage

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.
    - V10 zparm: DISALLOW_DEFAULT_COLLID
    - NO: Automatic Rebind erstellt package in DSN_DEFAULT_COLLID_planname
    - YES: Erfordert REBIND PLAN .... COLLID Option
  - Reorg None für LOBs wird nicht mehr unterstützt.
  - Release Bind Option wird bei DRDA Zugriff unterstützt.
  - zPARMs
    - Sehr viele Neuerungen (FlashCopy, Security, Performance ...)
    - Sehr viele Änderungen (PARTKEYU, CHKTYPE, ...)
  - Incompatibility Changes
    - SQL
      - ORDER BY
      - CHAR function
      - TIMESTAMP function
    - Security for DB2 z/OS requester
  - ....

ORDER BY

- DDL and DML

```
CREATE TABLE T1.SZI10T (
COL_INT    INTEGER ,
COL_CHAR   CHAR(10)
) IN ... ;

INSERT INTO    T1.SZI10T
VALUES ( 1 , '1. ROW') ;

INSERT INTO    T1.SZI10T
VALUES ( 2 , '2. ROW') ;
```

ORDER BY

- V9 example:

```
SELECT COL_INT FROM T1.SZI10T
ORDER BY +1 DESC ;

!    COL_INT     !
+----------------+
!              2 !
!              1 !
```

- V10 example:

```
SELECT COL_INT FROM T1.SZI10T
ORDER BY +1 DESC ;

!    COL_INT     !
+----------------+
!              1 !
!              2 !
```


CHAR (decimal) scalar function

- APAR PM29124 ermöglicht V9 compatibility
  - New DSNZPARM BIF_COMPATIBILITY

```
SELECT COL1
      ,CHAR(COL1)
      ,HEX(CHAR(COL1))
      ,SUBSTR(CHAR(COL1), 2, 4) APPL_SUBSTR
FROM DB2_TABLE;
```

```
+-------------------------------------------------+   +-------------------------------------------------+
!  COL1   ! CHAR_V9 ! HEX_CHAR_V9  ! APPL_SUBSTR !   !  COL1   ! CHAR_V10 ! HEX_CHAR_V10 ! APPL_SUBSTR !
+-------------------------------------------------+   +-------------------------------------------------+
!     9   !  0009,  ! 40F0F0F0F96B ! 0009        !   !     9   ! 9        ! F94040404040 !             !
!    99   !  0099,  ! 40F0F0F9F96B ! 0099        !   !    99   ! 99       ! F9F9404040040 ! 9           !
!   999   !  0999,  ! 40F0F9F9F96B ! 0999        !   !   999   ! 999      ! F9F9F9404040 ! 99          !
!  9999   !  9999,  ! 40F9F9F9F96B ! 9999        !   !  9999   ! 9999     ! F9F9F9F94040 ! 999         !
+-------------------------------------------------+   +-------------------------------------------------+
```

- Problem occurs after REBIND / BIND PACKAGE in V10 CM
- A new trace record, IFCID 366, has been added to trace when the old format is returned by DB2. This can be used to identify which applications need to be changed to handle the new format returned in DB2 10.
- Add schema SYSCOMPAT_V9 to PATH BIND option or CURRENT PATH special register.

TIMESTAMP function – Watch out for PM48741

- **1. example**

```
SELECT TIMESTAMP(CHAR(CURRENT_DATE- 1 DAYS, EUR) CONCAT '00.00.00.000000' )
FROM SYSIBM.SYSDUMMY1;
```

- **„the fact that '14.08.201100.00.00.000000' was allowed as input to the TIMESTAMP built-in function in V9 was a bug. None of our supported timestamp formats allows the date portion to begin with anything other than the year.**

- **2. example**

```
WHERE CURRENT TIMESTAMP > '2011-08-18 07:30:10'        => OK
WHERE CURRENT TIMESTAMP > '2011-08-18-07.30.10.000000' => OK
WHERE CURRENT TIMESTAMP > '2011-08-18-07.30.10.000000' => OK
WHERE CURRENT TIMESTAMP > '2011-08-18 07.30.10.000000' => OK
WHERE CURRENT TIMESTAMP > '2011-08-18 07:30:10'        => OK
WHERE CURRENT TIMESTAMP > '2011-08-18-07:30:10'        => -180
```

- **According to the books only support for the following timestamp strings:**
  - yyyy-mm-dd-hh.mm.ss.nnnnnn
  - yyyy-mm-dd hh:mm:ss.nnnnnn (the ODBC and JDBC string representation)
  - ..

Security for DB2 z/OS requester

- DB2 z/OS requester bekommt gleiche Security Voraussetzungen wie DB2 DRDA Client
  - Connecting Userid benötigt Execute Privilege für Package
- PM37300 erlaubt alte private protocol semantics in V10
  - zparm PRIVATE_PROTOCOL=AUTH
- Umstellung der Security entsprechend DRDA Clients
  - zparm nur in V10

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 New Function Mode: Catenfm

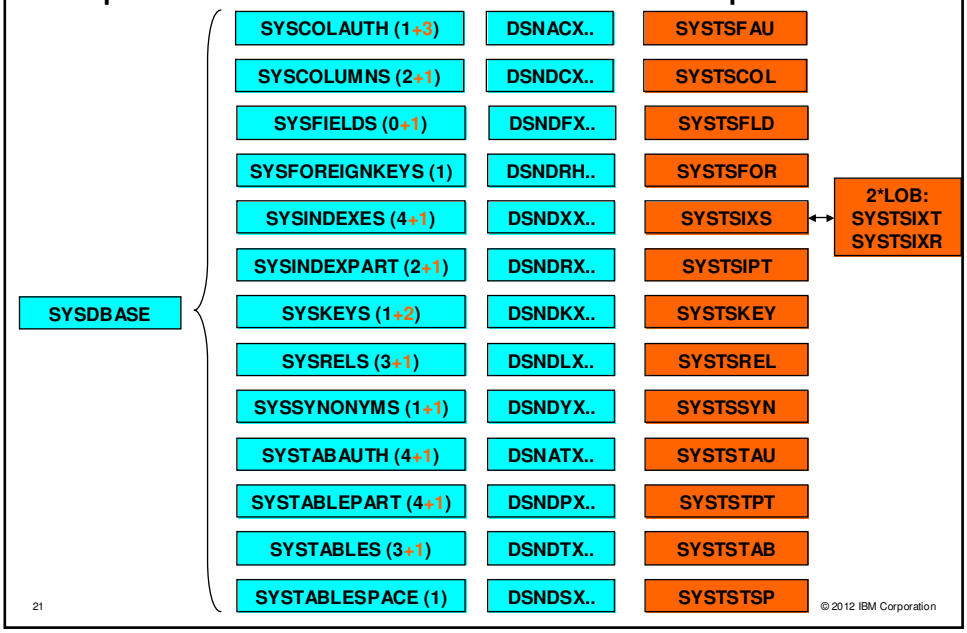
- Umbau sehr vieler Catalog/Directory Tablespaces
- Warum ?
  - Contention auf Catalog/Directory durch parallel DDL/Bind usw.
  - 64GB Grenze

Catalog Änderungen in V10

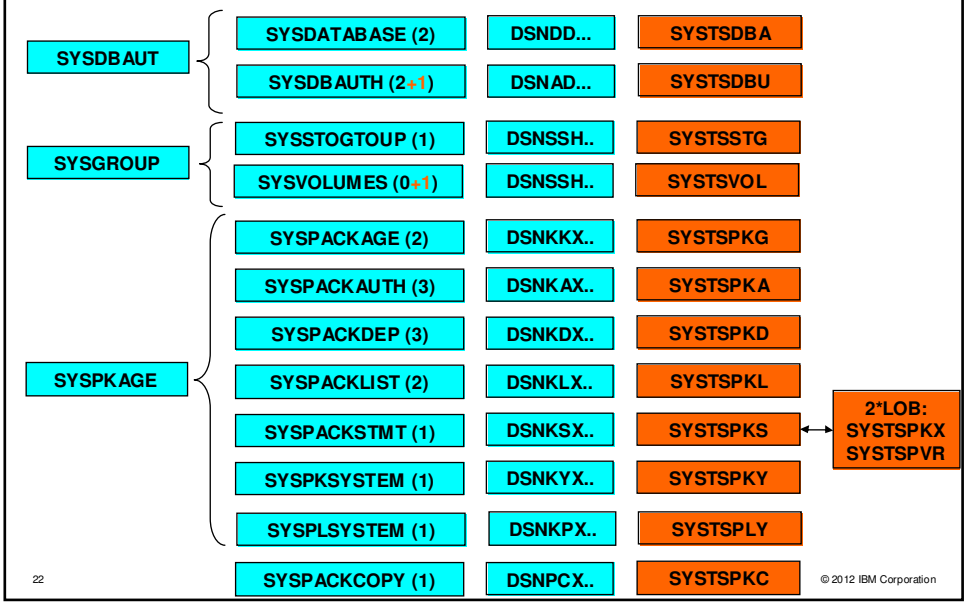
- Unveränderte Tablespaces / Tables
  - SYSALTER
  - SYSCONTX
  - SYSCOPY: Weiterhin als einziger Tablespace in EBCDIC
  - SYSDDF
  - SYSEBCDC
  - SYSGPAUT
  - SYSGRTNS
  - SYSHIST
  - SYSJAVA
  - SYSROLES
  - SYSRTSTS
  - SYSSEQ
  - SYSSEQ2
  - SYSSTATS
  - SYSSTR
  - SYSTARG
  - SYSUSER
  - SYSXML

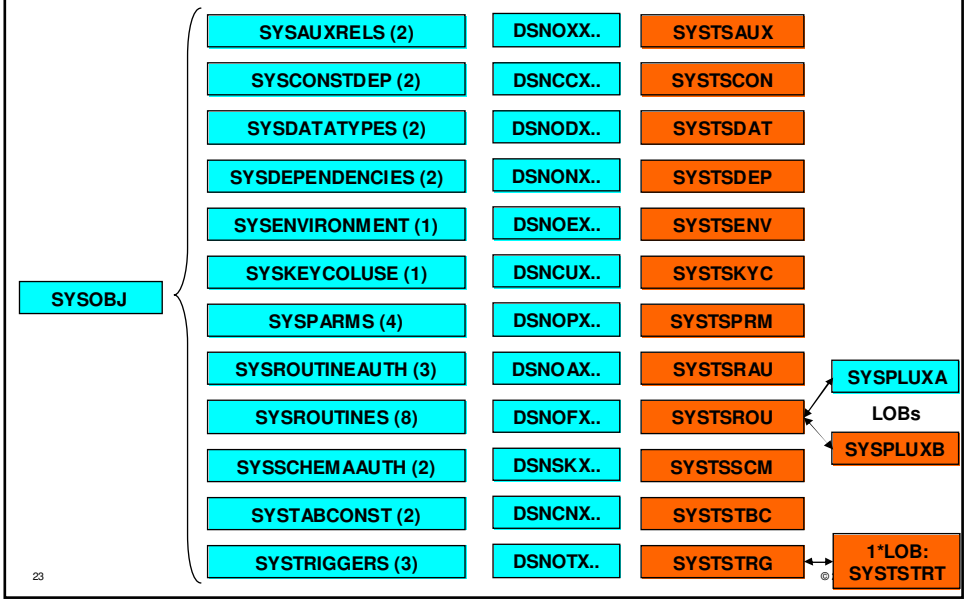
Catalog Änderungen in V10: Konvertierung

- Bisherige Tablespaces
  - SYSDBASE
  - SYSDBAUT
  - SYSGROUP
  - SYSOBJ
  - SYSPKAGE
  - SYSPLAN
  - SYSVIEWS
- Neue Tablespaces mit migrierten Tabellen (~44)
  - SYSTS...

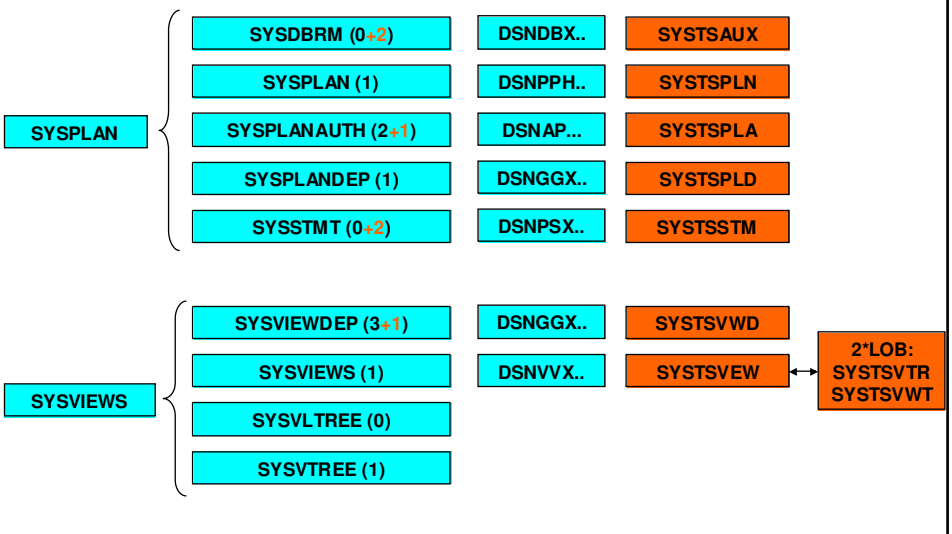


DB2 V10 – Im Einsatz, wo andere längst aufgeben...





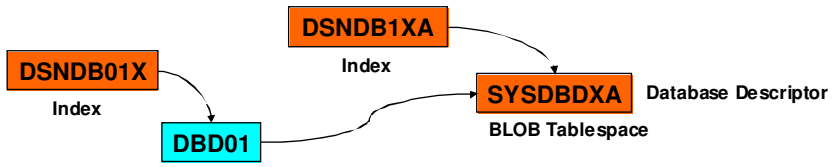
DB2 V10 – Im Einsatz, wo andere längst aufgeben...

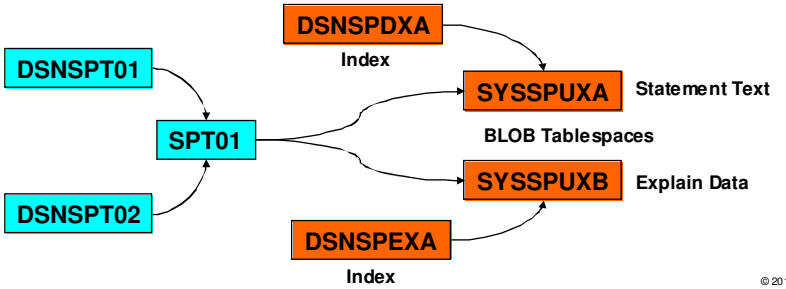


Catalog Änderungen in V10

- Neue Tablespaces mit neuen Tabellen (~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





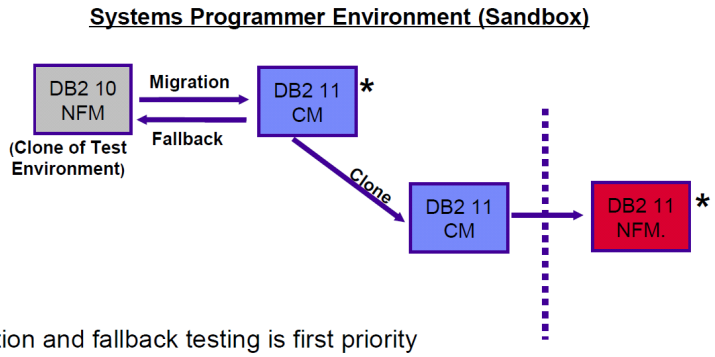
Allgemeine Änderungen im Catalog/Directory

- Keine Links oder Hashes
- SMS managed (schon im CM)
- 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
- Directory tablespace SPT01 Performance-/ Space-Optimierung
  - Inline LOBs (PM27073 / PM27811)
  - zPARM Compress_SPT01 macht damit weiterhin Sinn
  - BIND performance

Migration Planning

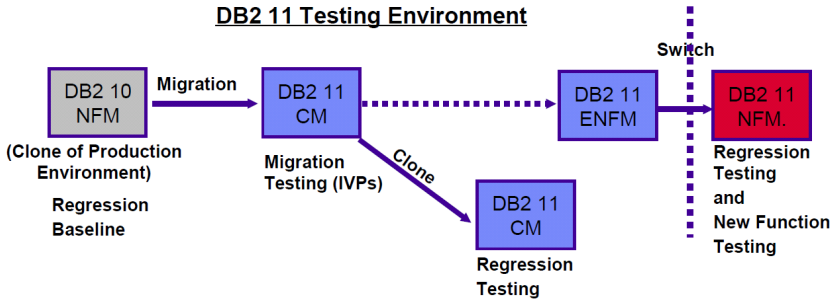
- Review the Installation Guide checklists
- Check HW and SW prerequisites
  - Plan for 10 – 30% more Realstorage
  - Check Information APAR
- Install Fallback SPE on all V9 NFM members / systems
- Install early code
- Contact vendors
  - Assess ISV readiness and requirements
- Install current recommended service level for V10 test environment
  - Plus HIPER, PE corrections
- Run pre-migration job
- Check and correct release incompatibilities
  - Installation Guide
- Check removed and deprecated features
- Fallback testing before moving from DB2 V10 CM to ENFM
- Consider processing peaks: month/quarter/year ends for CM regression tests
- Use Planmanagement EXTENDED for static sql applications
- Save performance and access path information
- Before REBIND in DB2 10
  - Complete performance workload under V9 NFM; document performance profile
  - Complete performance workload under V10; document performance profile
  - Compare DB2 9 profile (#SQL, #Getpages, cpu times ....) with V10 profile
- ....

Recommended Parallel Test Strategy – Part 1



- Migration and fallback testing is first priority
- Then clone / copy DB2 11 CM to a second DB2 11 CM that gets migrated to NFM
  - Use of IBM Cloning Tool available for the duration of the DB2 11 ESP
- This provides two targets (*) for problem re-creation and PTF testing during the ESP, one in DB2 11 CM, one in DB2 11 NFM

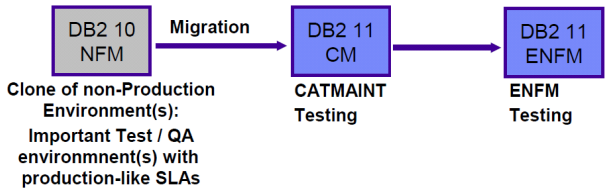
Recommended Parallel Test Strategy – Part 2



- DB2 10 NFM: Baseline for regression testing
- Migration testing: DB2 11 CM
- Regression testing in DB2 11 CM clone
  - Before rebinds
  - After rebinds
  - Optim Query Capture Replay (OQCR) may be helpful here *
- Regression testing in DB2 11 NFM
- New feature / function testing in DB2 11 NFM

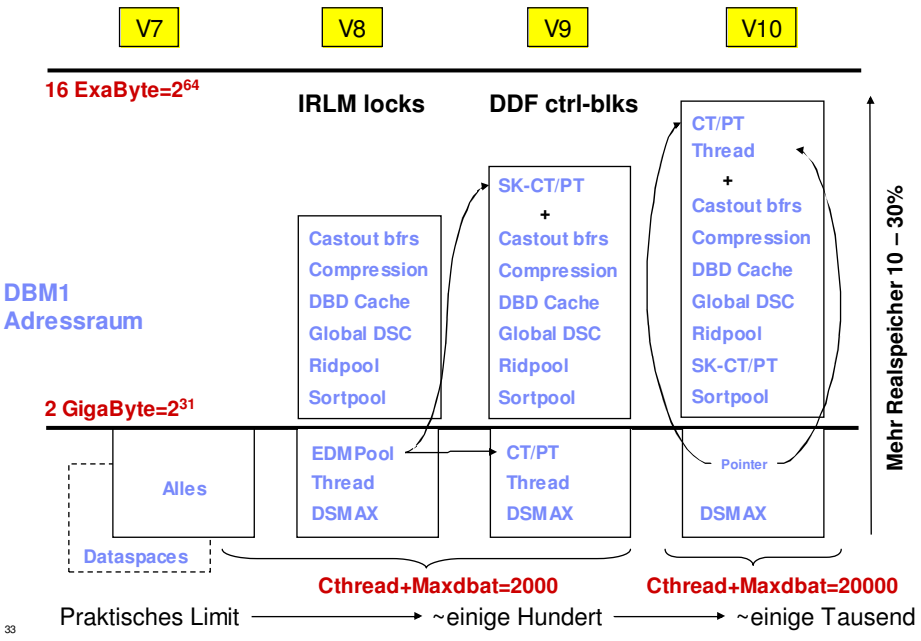
Recommended Parallel Test Strategy – Part 3

Pre-migration Catalog Migration Testing

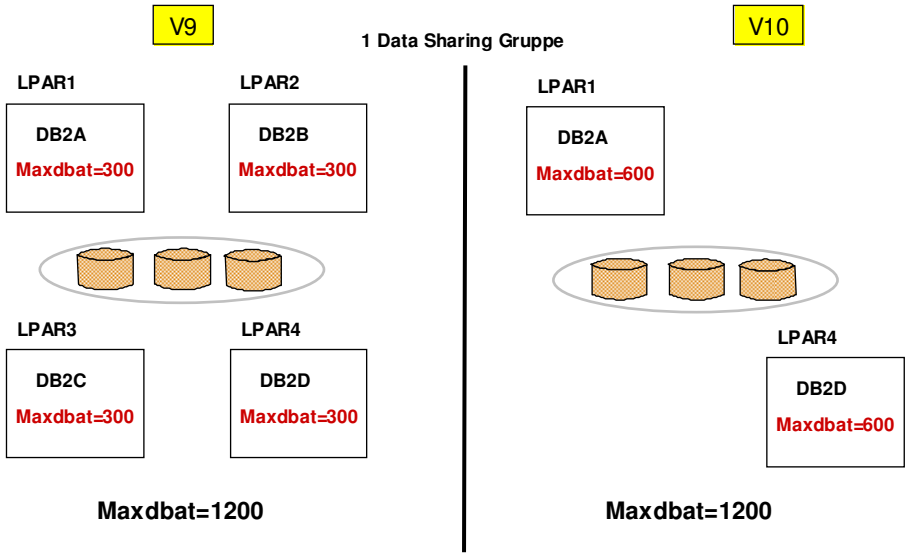


- Verify CATMAINT and ENFM will work
- Flush out errors
- Learn how long each process takes

DB2 V10: Vorteile nach der Migration

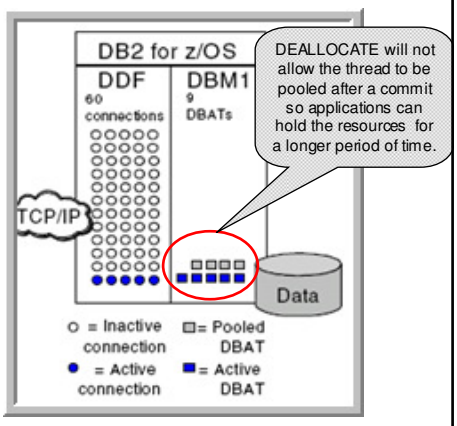


DBM1 Storage Relief: Konsolidierungsbeispiel



DDF – High Performance DBAT

- Designüberlegungen
  - Catalog Restructure
    - Mehr konkurrierende Zugriffe
    - Row-level locking
  - VSCR
    - Thread storage / EDM pool above the bar
  - Aufheben von RELEASE COMMIT Verhalten für DRDA
- Vorteil: CPU Reduzierung
  - Viele kurzlaufende Transaktionen
- vs mehr Real Storage und reduced concurrency
  - Thread storage wird länger gehalten
  - REBIND/BIND und DDL
- Aktivierung
  1. RELEASE (DEALLOCATE)
     - Holds thread active across commits, limiting pooling, until thread is terminated
     - Package / Tablespace locks
     - default mit JDBC 3.59+ / 4.9+ packages / PM15292/3
  2. MODIFY DDF PKGREL(BNDOPT/COMMIT)
     - Beeinflusst inactive thread Verhalten
     - DDF service task (120 sec)



DDF Online Changes

- **Availability Verbesserungen - PM26480**
  - Dynamisches Hinzufügen, Verändern und Löschen von bis zu 40 Location Aliase
    - MODIFY DDF ALIAS command
      ADD, DELETE
      START, STOP, CANCEL
    - Port, IP-address … Definitionen
    - Mehr Flexibilität bei der Zuordnung von Application Servern ….
  - Online CDB
    - Neue Connections übernehmen Veränderungen an SYSIBM.LOCATIONS, IPNAMES und IPLIST
    - DISPLAY LOCATION command output

Neue DDF connection / thread Überwachung / Kontrolle

- Immer mehr remote Anwendungen greifen auf DB2 z/OS als DB-Server zu!
  - Kontrolle auf Application Server Ebene liegt außerhalb des DB-Verantwortungsbereichs
  - DB2 Connect gateways verlieren an Bedeutung
  - Client Driver Komponenten funktional identisch oder besser, flexibler, geringerer Installationsaufwand ....
  - Omegamon XE Extended Insight for DB2 PE
    - Monitoring basiert auf Client driver Komponenten
- Erlaubt DB2 System-Administration feingranulare Überwachung und Begrenzung von Remote Verbindungen
  - Heute im wesentlichen auf systemweite Systemeinstellungen beschränkt
    - MAXDBAT, CONDBAT, IDTHTOIN

System Monitoring für DRDA Anwendungen

- **Granulare Überwachung und Begrenzung von Connections / Threads**
- **Profil-Definitionen in DSN_PROFILE_TABLE**
  - Wer, welche Anwendung bzw. Application Server soll überwacht und ggf. begrenzt werden?
  - Client-IP address, Domain name, Auth-id, Product-id, Collect-id, Package, Location, Client-Userid, Client-Workstation, Client-Application
  - Watch out for APAR PM28500 für Location, Location Alias, Client-Userid ....
- **Festlegung der Begrenzung und Aktionen in DSN_PROFILE_ATTRIBUTES**
  - Function keywords
  - MONITOR CONNECTIONS
    - Überwacht Connection Anzahl von einer bestimmte IP-Adresse oder Domain name
  - MONITOR THREADS
    - Überwacht aktive thread Anzahl anhand unterschiedlicher Kriterien
  - MONITOR IDLE THREADS
    - Überwacht die idle time für aktive threads anhand unterschiedlicher Kriterien
  - Events
    - EXCEPTION, WARNING, DIALGLEVEL, ...
    - Statistic trace record (IFCID 402)
- **Aktivierung**
  - START PROFILE command
    - Member-specific
    - Initial und Refresh von Profilen
  - Profile mit column PROFILE_ENABLED = ‚Y‘

Beispiele zu System Monitoring für DRDA Anwendungen

SYSIBM.DSN_PROFILE_TABLE

ROLE	AUTHID	PLANNAME	COLLID	PKGNAME	IPADDR	PROFILEID
null	*null*	*Null*	*null*	*null*	9.12.5.67	12
null	*null*	*null*	PRODCOLID	PRODPKG	*null*	14

SYSIBM.DSN_PROFILE_ATTRIBUTES

PROFILED	KEYWORD	ATTRIBUTE1	ATTRIBUTE2	ATTRIBUTE3	ATTRIBUTE_ TIMESTAMP	REMARKS
12	MONITOR CONNECTIONS	WARNING	100	*null*	2011-02-23...	Issue msg after 100 connections
14	MONITOR IDLE THREADS	EXCEPTION	30	*null*	2011-02-23...	Cancel after 30 seconds idle
14	MONITOR THREADS	EXCEPTION	10	*null*	2011-02-23...	Cancel threads greater than 10

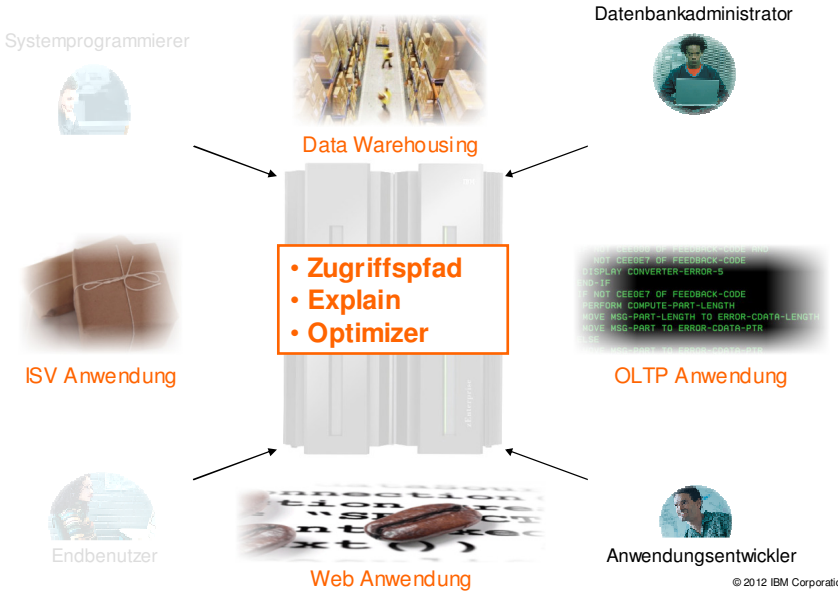
SQL Performance Monitoring und Problemfindung

- Identifikation von „Problem SQL“ teilweise schwierig
- Eindeutige Zuordnung fehlt
  - Timeouts, Deadlocks, Unavailability Resources messages
  - Traces
- V10 Erweiterung
  - **Statement Level information** für static & dynamic SQL
  - Messages, Traces und Performance Daten
    - BIND/REBIND in V10 NFM
      Synchronisiert SYSIBM.SYSPACKSTMT und SPT01
    - Dynamic SQL erfordert Dynamic Statement Cache
      Statement-ID mit Wert 1 für dynamic DDL und Lock Table statements
  - Eindeutiger Wert pro DB2 system

Verbessertes Monitoring auf Statement Ebene

- **STMT_ID** Information
  - Traces, Messages, Thread-Info token …
  - Neue Monitor Class 29
    - IFCIDs 316/318 für dynamic SQL; 401/400 für static SQL
  - More IFCID changes
    - 350, 124 – SQL statement records
    - 172 – deadlock; 196 - timeout
    - 65/66 – open/close cursor
    - ....
  - Dynamic SQL
    - EXPLAIN STMTCACHE ALL
    - STMT_ID column in DSN_STATEMENT_CACHE_TABLE
  - Static SQL:
    - New STMT_ID column in SYSPACKSTMT
  - STMT_ID für DRDA application requester plus:
    - 2-byte header beinhaltet SQL type (dynamic oder static)
    - 10-byte compilation time in timestamp format (yyyymmddhhmmssnnnnnn)

DB2 V10: Qualitätssicherung



Plan/Package Management

- Neue Column Lastused
  - in SYSPLAN
  - In SYSPACKAGE/SYSPACKCOPY
  - zPARM DISABLE_EDMRTS (PM37672)
- Plan Stability wurde erweitert
  - 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

Weiternutzung alter Zugriffspfade von V9

- Erweiterungen mit PM25679, z.B.
  - Explain(Only) bei REBIND
  - Plan Table Column BIND_EXPLAIN_ONLY
- BIND/REBIND… APREUSE (ERROR|NONE)
  - DB2 versucht den alten Zugriffspfad für alle SQLs zu nutzen
  - Dokumentation in Message DSNT286I
  - Explain(Yes): Information in PLAN_TABLE Column REMARKS und HINT_USED
- BIND/REBIND… APCOMPARE (WARN|ERROR|NONE)
  - Neuer Zugriffspfad wird erzeugt
  - Änderungen werden dokumentiert in DSNT285I
  - Explain(Yes): Information in PLAN_TABLE Column REMARKS

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.db2z10.doc.perf/src/tpc/db2z_managingqueryplans.htm

APREUSE / APCOMPARE

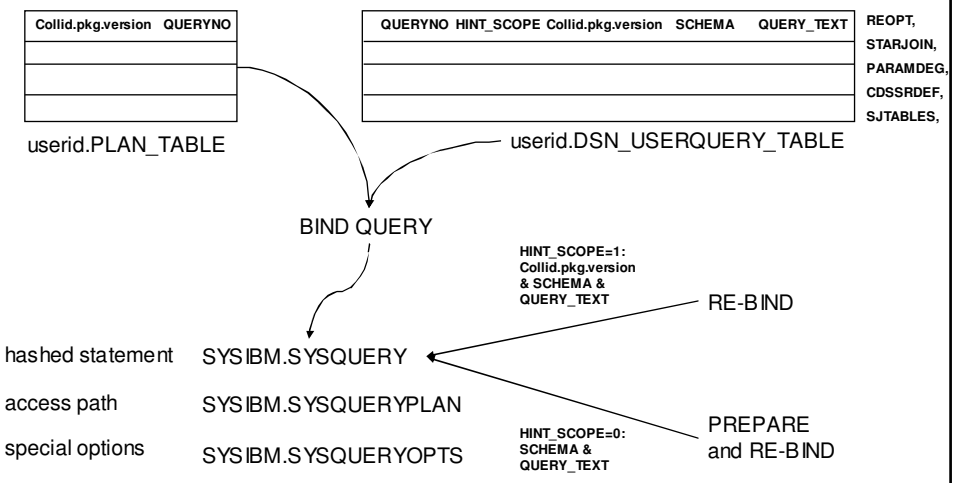
- Bei BIND wird nach folgende Kriterien gesucht
  - Location
  - Collection
  - Name
  - Version (oder zurückliegende Versionen)
- Bei zurückliegenden Versionen wird der Statement-Text verglichen (White-Spaces werden ignoriert)

```
DSNT285I  -csect-name REBIND FOR PACKAGE = package-name,
           USE OF APCOMPARE RESULTS IN:
           x STATEMENTS WHERE COMPARISON IS SUCCESSFUL
           y STATEMENTS WHERE COMPARISON IS NOT SUCCESSFUL
           z STATEMENTS WHERE COMPARISON COULD NOT BE PERFORMED.

DSNT286I  -csect-name REBIND FOR PACKAGE = package-name,
           USE OF APREUSE RESULTS IN:
           x STATEMENTS WHERE APREUSE IS SUCCESSFUL
           y STATEMENTS WHERE APREUSE IS EITHER NOT SUCCESSFUL
             OR PARTIALLY SUCCESSFUL
           z STATEMENTS WHERE APREUSE COULD NOT BE PERFORMED
           …
```


CURRENT EXPLAIN MODE

- Bietet die Möglichkeit dynamisches SQL automatisch zu explainen
- 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

Statement-Level AccessPath / OptParameter Hints



Auswertung / Nutzung

- Spalte HINT_USED in PLAN_TABLE
  - mit Keyword SYSQUERYPLAN #queryid
- Rebind

```
  REBIND PACKAGE ( DB.SZI10P11.(V1) )      APRETAINDUP(NO)
DSNX105I  § REBIND SQL WARNING
           USING SYSADM AUTHORITY
           PLAN=(NOT APPLICABLE)
           DBRM=SZI10P11
           STATEMENT=108
           SQLCODE=394
           SQLSTATE=01629
           TOKENS=
```

- Prepare

```
DSNT404I SQLCODE = 394, WARNING:  USER SPECIFIED OPTIMIZATION HINTS USED
DURING ACCESS PATH SELECTION
DSNT418I SQLSTATE   = 01629 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXOPCO SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = 20  0  1  1093230542  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'00000014'  X'00000000'  X'00000001'  X'41295FCE'
X'000
```

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: 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: Matching ab der zweiten Spalte
  - 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
    - V9: multiple IX Access
    - V10: Accesstype=NR, Sort Avoidance

```
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)

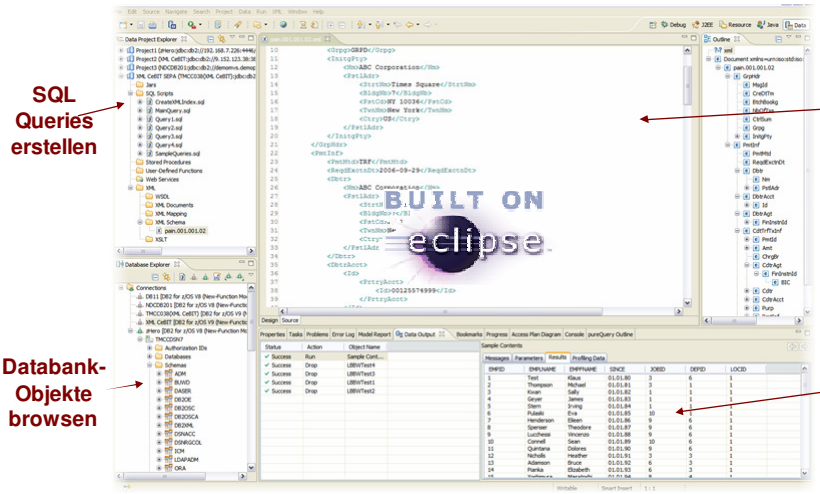
- Multi-Row-Fetch ist erlaubt
  - Nur für Read Only Cursors
- Workfile kann gemeinsam benutzt werden
  - Nur für CPU Parallelism
- Effektivität bei Parallelism:
  - Aufteilung ist nicht mehr abhängig z.B. von den Partitionsgrenzen:
    - Ziel ist gleichmäßige Aufteilung auf Basis Rekordanzahl
    - „Dynamic 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“

Weitere Optimizer Themen

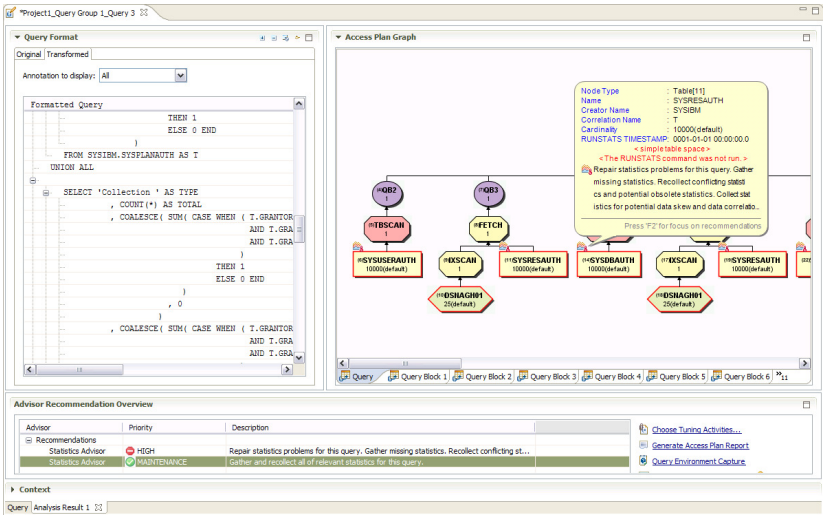
- APAR PM56845: ZPARM OPT1ROWBLOCKSORT
  - ENABLE : When OPTIMIZE FOR 1 ROW is used with a query, DB2 will disable sort access paths when a no-sort choice is available.
  - DISABLE (default): DB2 will strongly discourage sort access paths and is unlikely to choose sort access paths, but there will be a chance a sort access path can win. This is the behavior which was present in DB2 9 and prior releases.
- Pushdown von “teuren” Stage 2 Predikaten
  - Index Screening oder Stage 1 für
    - Arithmetic expressions, date-time expressions
    - scalar built-in functions, CASE and CAST
  - Erkennbar in DSN_FILTER_TABLE.PUSHDOWN
  - Index Matching nur durch IndexOnExpression
- Prüfung der RTS bei fehlenden Statistiken (bei BIND, PREPARE)
  - Tabelle mit 0 rows oder VOLATILE, NPGTHRSH
  - Externalisiert in DSN_COLDIST_TABLE

IBM Data Studio und OptimQueryWorkloadTuner

Empfohlen für proaktives Tuning und Problembearbeitung bei Optimizer Themen



Zugriffspfadanalyse, Service SQL und StatsAdvice



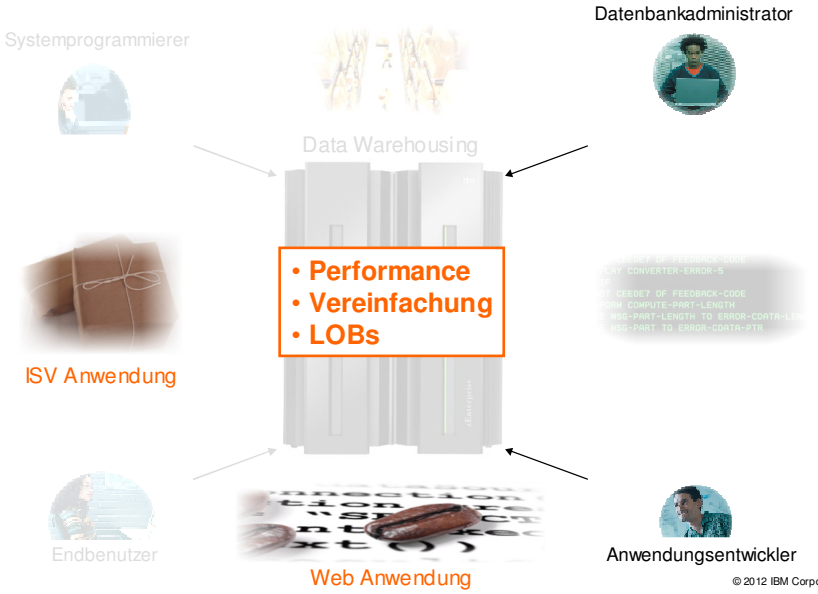
Contents of Eclipse-based Query Tuning offerings

	Data Studio	InfoSphere Optim Query Tuner for	Infospshere Optim Query Workload Tuner
Queries from all sources	✓	✓	✓
Reports	✓	✓	✓
Query Formatter	✓	✓	✓
Access Plan Graph	✓	✓	✓
Query Statistics Advisor	✓	✓	✓
Query Annotation		✓	✓
Visual Plan Hint		✓	✓
Query Index Advisor		✓	✓
Query Advisor		✓	✓
Access Path Advisor		✓	✓
Workload Statistics			✓
Workload Index Advisor			✓
Workload Query Advisor			✓

Agenda

09:30	Einführung
09:45	Migration & Vorteile nach der Migration
10:45	Neue Funktionen zur Qualitätssicherung
11:15	Pause
11:30	Anwendungsoptimierung
12:00	Neue Anforderungen an IT-Sicherheit
12:45	Mittagspause
13:30	DB2 auf Zeitreise, Bi-Temporal Data
14:15	Erweiterungen von OLTP bis Warehousing
15:30	Pause
15:45	IBM DB2 Analytics Accelerator
16:00	Unicode in DB2 z/OS
16:15	Abschluss und Q&A
16:45	Veranstaltungsende

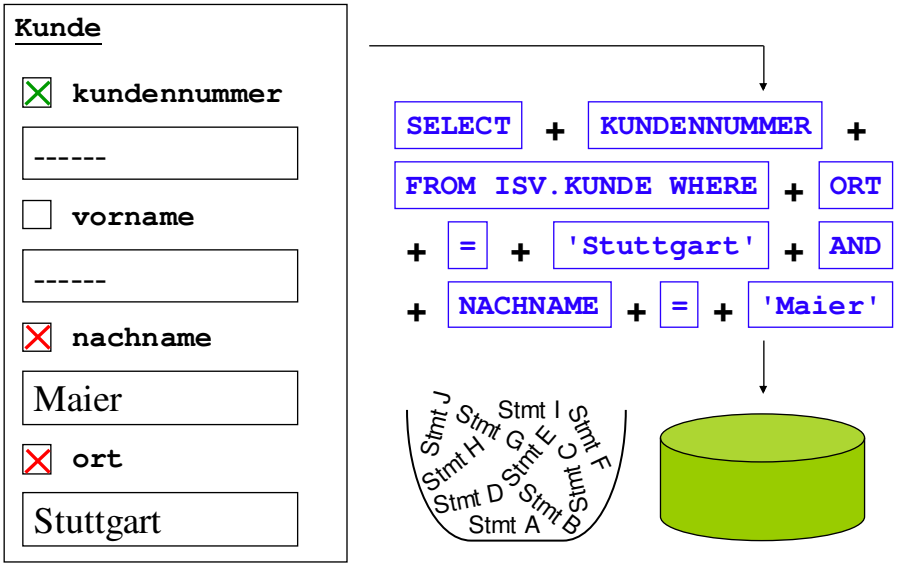
DB2 V10: Anwendungsoptimierung



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



Änderungsmöglichkeit ist jedoch oft nicht gegeben

- Bisherige Ausgangslage:
  - X Schlechte Coding Practices oder Notwendigkeit für hochdynamisches SQL mit String Concatenation
  - X 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 = &
```

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

Nutzung von Literal Replacement überprüfen

DSN_STATEMENT_CACHE_TABLE

STMT_ID	LITERAL_REPL	STMT_TEXT	...
264	R	SELECT … WHERE KUNDENNR = &	…
265	D	SELECT … WHERE KUNDENNR = &	…

Nutzung nachvollziehbar
über EXPLAIN Table

Dynamische Anwendungen erhöhen die Komplexität

Kunde

- [x] kundennummer
- [ ] vorname
- [ ] nachname
- [x] anrede
- [ ] titel
- [x] strasse
- [x] hausnummer
- [x] postleitzahl
- [x] ort
- [ ] festnetznummer
- [ ] mobilfunknummer
- [x] emailadresse

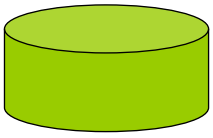
...

```
UPDATE ISV.KUNDE SET ...
```

$2^x - 1$

unterschiedliche Kombinationen

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



Entwicklung & Betrieb leiden unter dieser Komplexität

- Bisherige Lösungen:
  - Sehr viele einfache SQL Statements
    - X Hoher Programmieraufwand & Sicherheitsrisiken
    - X Ineffiziente Nutzung Dynamic Statement Cache
  - Wenige komplexe SQL Statements
    - X Hoher Programmieraufwand mit unnötigen Zugriffen
    - X 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);
```

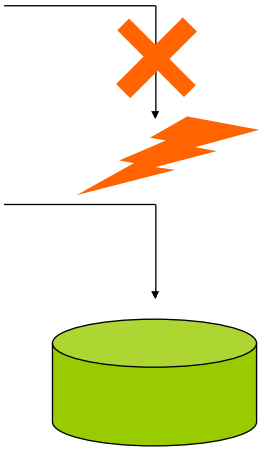
Mit Parameter-Markern, z.B. in Java (mit JCC-Property enableExtendedIndicators)

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

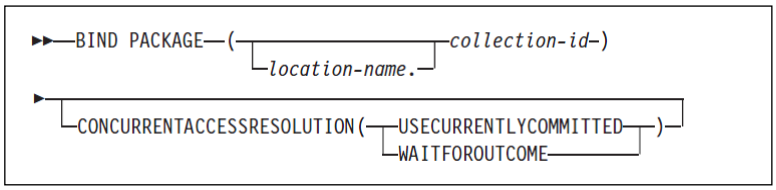
Anwendung B

Bisher keine zufriedenstellende Lösung vorhanden

- Bisherige Alternativen:
  - Isolation Level Uncommitted Read
    - X Unter Umständen Verarbeitung von inkonsistenten Daten
  - Isolation Level Cursor Stability oder höher
    - X SQLCODE -913 … CAUSED BY DEADLOCK OR TIMEOUT
- Lösung mit DB2 10 für READ Zugriffe:
  - **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**)
  - Universal Tablespace

Einsatz von Currently Committed in Anwendungen

Als Bind-Parameter für statisches SQL

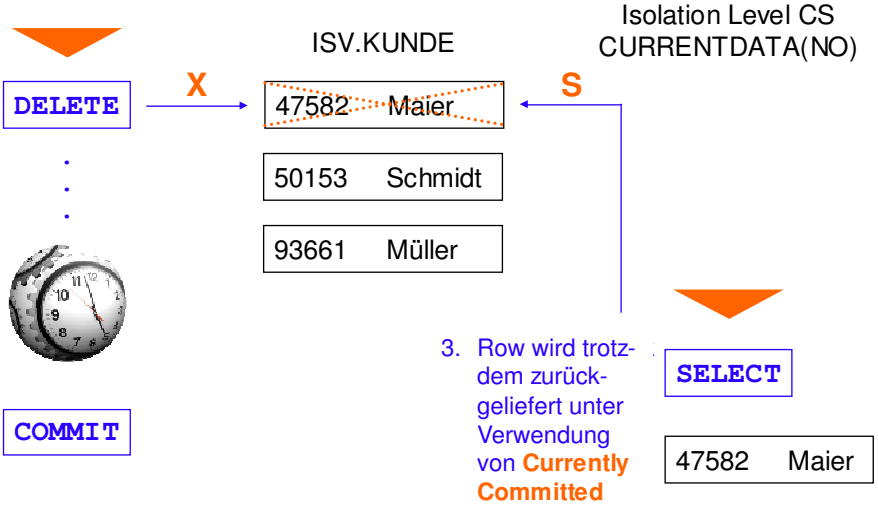


Auf Connection-Ebene in Java (oder mit JCC-Property **concurrentAccessResolution**)

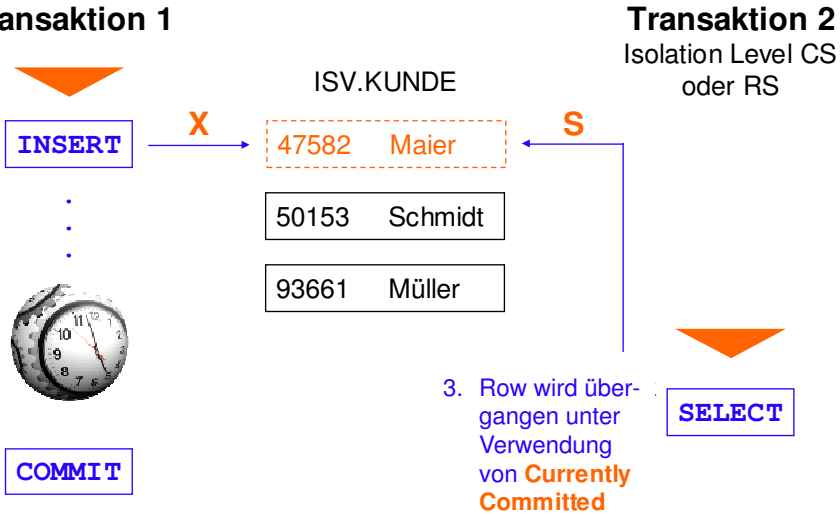
```
((DB2Connection)con).setDBConcurrentAccessResolution(1);
```

Beim PREPARE als Attribut:
- USE CURRENTLY COMMITTED
- WAIT FOR OUTCOME

Currently Committed Beispielszenario I – DELETE

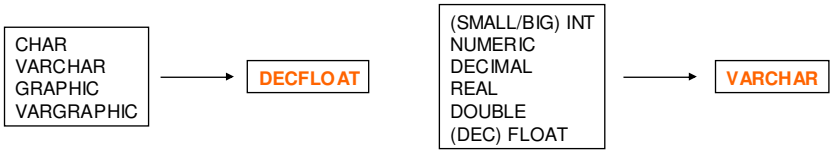


Currently Committed Beispielszenario II – INSERT



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**



Implicit Casting – Numerische zu String-Datentypen

Quell-Datentyp	Ziel-Datentyp
SMALLINT	VARCHAR(6)
INTEGER	VARCHAR(11)
BIGINT	VARCHAR(20)
NUMERIC/DECIMAL	VARCHAR(precision+2)
REAL	VARCHAR(24)
FLOAT	VARCHAR(24)
DOUBLE	VARCHAR(24)
DECFLOAT	VARCHAR(42)

Der Ziel-Datentyp **VARCHAR** kann in der Folge mit anderen Character- und Graphic-Datentypen verglichen werden

Implicit Casting – String zu numerischen Datentypen

Quell-Datentyp	Ziel-Datentyp
CHAR	DECFLOAT(34)
VARCHAR	DECFLOAT(34)
GRAPHIC	DECFLOAT(34)
VARGRAPHIC	DECFLOAT(34)
CHAR/VARCHAR FOR BIT DATA	N/A
BINARY	N/A
VARBINARY	N/A
BLOB	N/A
CLOB	N/A
DBCLOB	N/A

Der Ziel-Datentyp **DECFLOAT** kann in der Folge mit anderen numerischen Datentypen verglichen werden

DB2-interne Ablage für kleine LOBs unvorteilhaft

- Bisheriges Ablageverfahren:
  - X 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:
  - X 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 Speicher-bedarf 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
```

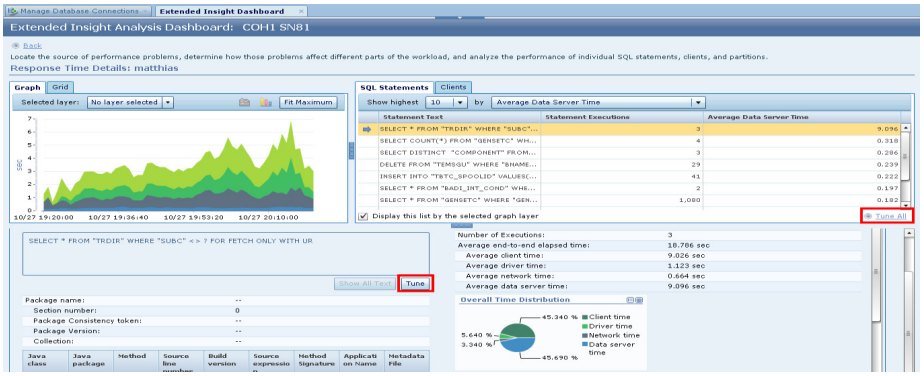
Zusätzliche Erweiterungen im Anwendungsumfeld

- 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

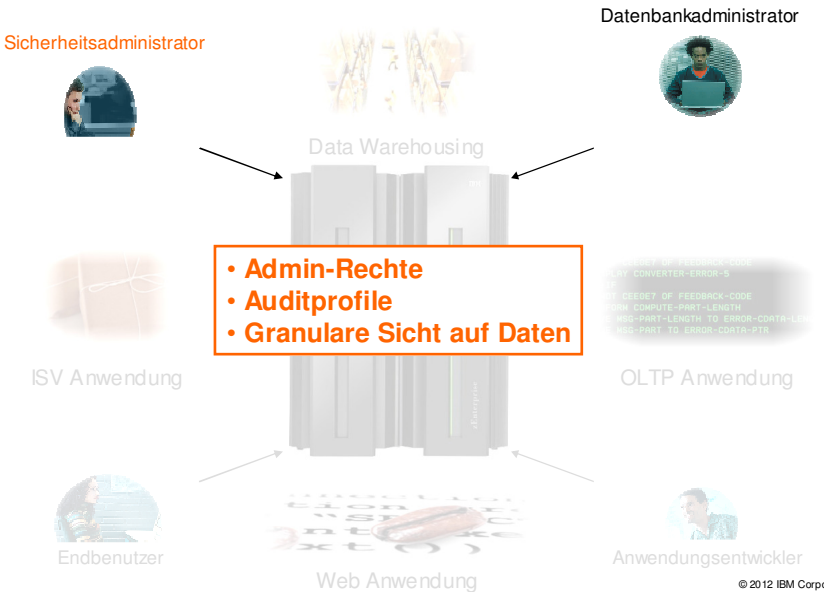
JDBC Treiber	JDBC 2.0	JDBC 3.0	JDBC 4.0
Client: DB2 8.1 FP6 z/OS: PQ80841 (2004, 2.1)	Version: ab 2.x Funktionen: - javax.sql - scrollable cursors - Lob locators - SQL Batch - Neue Datentypen		
Client: DB2 9.1 z/OS: PK32056 (2006, 3.1)		Version: 3.0.x-3.49.x (2.0 Standard +einige 3.0 Features) Funktionen: - PreparedStatements in ConnPool - Database MetaData - BOOLEAN Datentyp - Client reroute, SWB, ConnConc	
Client: DB2 9.5 z/OS: PK85149 (2009, 3.51) PK87569 (2009, 4.7)		Version: 3.50.x – 3.57.x Funktionen: - Autom. generierte Werte - Parameter in CALL Statements - BLOB / CLOB Erweiterungen - multiple local DB2 systems - Type 2 "failover" support - MultiRow INSERT for batching - T2 Multiple Row FETCH - Type 4 SSL Verschlüsselung	Version: 4.x – 4.7.x Funktionen: zusätzlich zu 3.50.x-3.57.x: - Auto Load Treiberklasse - Connection.isValid() - Client Info - RowID - Neues Exception Handling - XML - Annotations
Client: DB2 9.7, 10 z/OS: PM15292 (2010. 3.59) PM15293 (2010, 4.9) PM36838 (2011,4.12) PM65003 (2012, 4.14)		Version: ab 3.61.x - DB2 V10 - Literal Replacement - Currently Committed - Current Explain Mode,...	Version: ab 4.11.x - DB2 V10 - Literal Replacement - Currently Committed - Current Explain Mode,...

Omegamon XE for DB2 Performance Expert 510

Neue Möglichkeiten für End-To-End Montoring, speziell für Java, Dynamic SQL



DB2 V10: IT-Sicherheit



Sicherheitsbetrachtungen

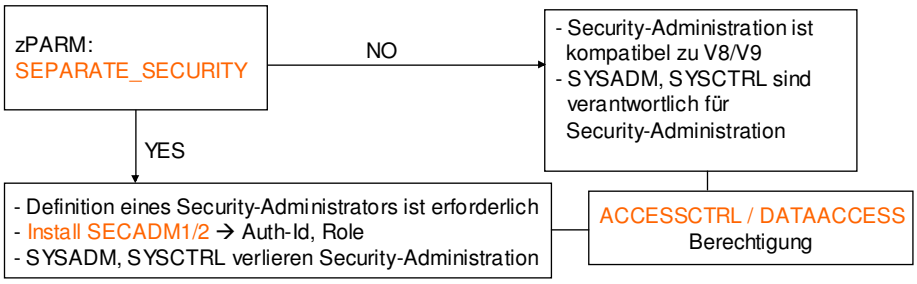
- Sicherheitsbedrohungen von extern & intern
- Granulare Sicht von speziellen Benutzergruppen auf sensitive Daten
- Vermeidung von impliziten Berechtigungen beim Datenzugriff durch priviligierte Benutzer
- Trennung von Verantwortlichkeiten
  - Datenmanagement & Berechtigungsvergabe
- Überwachung & Protokollierung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

Sicherheit & Compliance

- Mehr granulare administrative Berechtigungen
- Schutz von sensitiven Daten
  - Priviligierte 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**

Security-Administration vs. System-Administration



- SECADM übernimmt Management von
  - Sicherheitsfunktionen-/Objekten
    - Trusted Context & Roles
    - Neue Row & Column permission
    - Neue AUDIT tables & policies
  - Berechtigungsvergabe
    - GRANT / REVOKE
    - Neue administrative Berechtigungen
- SECADM hat keine Datenzugriffsberechtigung
- SECADM hat Zugriffsberechtigungen auf den DB2 Katalog

- ACCESSCTRL ermöglicht
  - Berechtigungsvergabe
    - GRANT / REVOKE
    - Ohne neue Admin Berechtigungen
    - Ohne WITH GRANT Option
  - Katalog Zugriffsberechtigungen
  - Auth-Id, Role
- DATAACCESS ermöglicht
  - Datenzugriff
  - Execute on Plan, Packages, procedures ..

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

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;
```

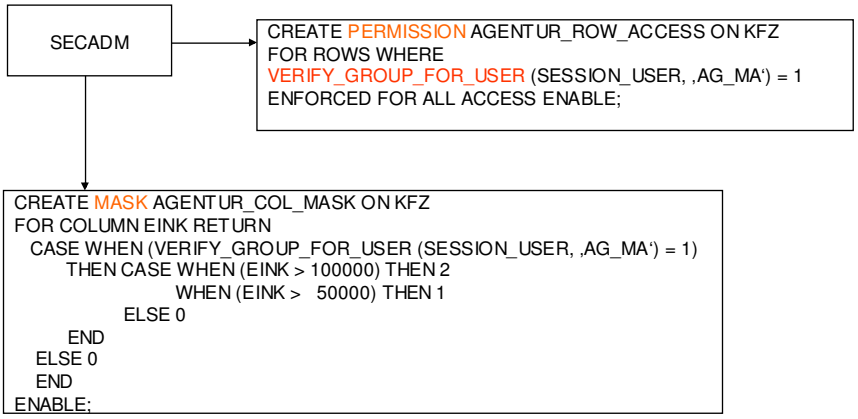

Wer darf die sensitiven Daten verarbeiten?

- 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

Row & Column level access - Beispiel



- Mitarbeiter der Vers.- Agentur (RACF group AG_MA) können auf die KFZ Tabelle zugreifen.
- Maskierung der Einkommenswerte, um eine Transparenz zu vermeiden.
- Optimizer führt Query Rewrite durch.

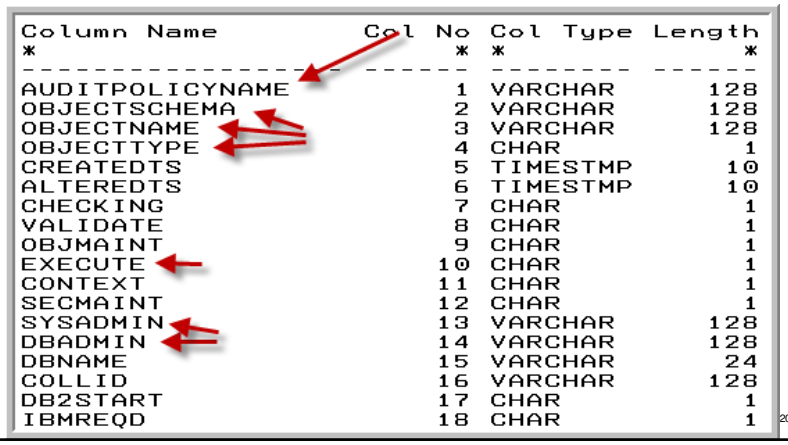
```
ALTER TABLE KFZ
ACTIVATE ROW LEVEL ACCESS CONTROL
ACTIVATE COLUMN LEVEL ACCESS CONTROL;
```


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 AUDTPLCY option
    - Generiert AUDIT traces
  - Autostart von bis zu 8 audit policies
- DISPLAY TRACE mit AUDTPLCY 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	TIMESTMP	10
ALTEREDTS	6	TIMESTMP	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

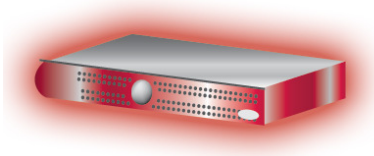
- Erstellen einer AUDITADMIN1 policy zum Auditing von SYSADM und SYSOPR berechtigte Benutzer

```
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.

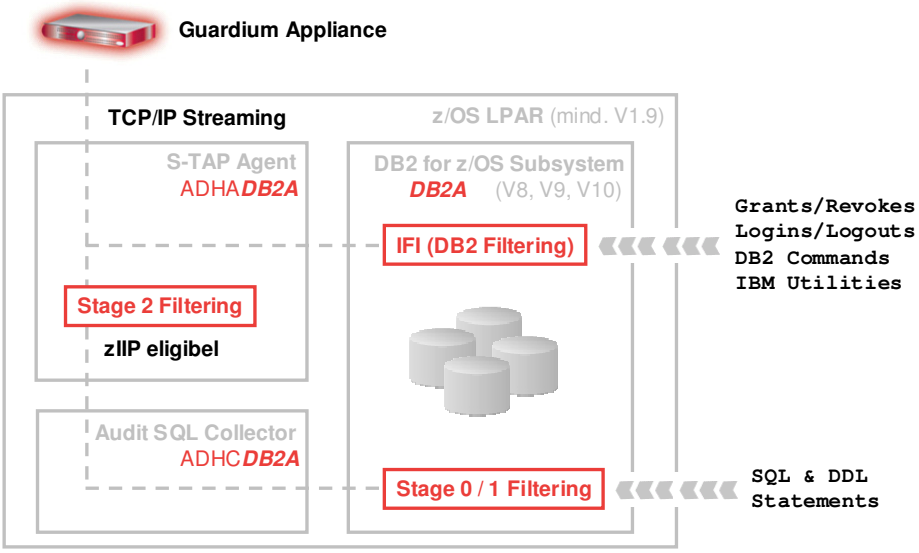
```
INSERT INTO SYSIBM.SYSAUDITPOLICIES
   (AUDITPOLICYNAME, OBJECTSCHEMA, OBJECTNAME, OBJECTTYPE, EXECUTE)
 VALUES('TEST2','TSCHEMA','''E_P%''','T','C');
```

Guardium: Unternehmensweite Audit & Complaince Lösung

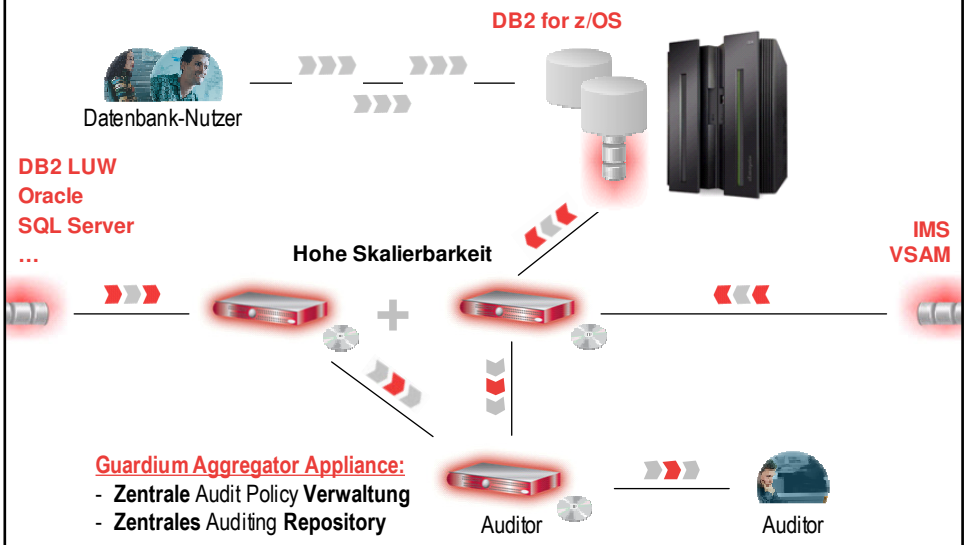


- Guardium S-TAP for DB2 z/OS
  - Gehärtete Appliance mit dedizierten Server
  - Sichere Speicherung der Auditdaten
  - Separate Administration
  - Integriert in die Guardium Enterprise Architecture
  - Proactive Security Test-Suite
  - Zentrales, plattformübergreifendes Audit Repository für unternehmensweite Analysen und Reporting von Mainframe und distributed Server
- Echtzeitmonitoring, Auditing und Reporting
  - SQL Statements mit Host Variablen & Statement Text
  - Berechtigungsvergabe
  - DDL Statements
  - DB2 Commands / Utilities
  - Logins/Logouts
  - Security Exceptions

Funktionsweise des Filtering in der S-TAP Architektur



Einsatz von Guardium in heterogenen Umgebungen



DB2 V10 – User Authentication

- **Client digital Certificates introduced in z/OS R10**
  - AT-TLS secure handshake accomplishes identification and authentication when the client presents its certificate as identification and its proof-of-possession as authentication
- **Password Phrases introduced in z/OS R10**
  - Support of mixed case letters, numbers, special characters, blanks and is between 9 to 100 characters long
  - Can be used instead of traditional 8-character password
  - DB2 Client 9.7 FP4+
- **Distributed Identity propagation introduced in z/OS R11**
  - A distributed identity filter is a mapping association between RACF userid and one or more distributed user identities, as there are known to application servers and defined in distributed user registries
  - Distributed identity filter allows to audit distributed workload / user identities.
- **TCPALVER=SERVER_ENCRYPT**
  - Requires AES or SSL

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
Pause	10:00	10:15	00:15	Alle
Blickwinkel Authentifizierung	10:15	11:00	00:45	Holger Wunderlich
Blickwinkel Autorisierung	11:00	11:35	00:35	Georg Kistenberger
Blickwinkel Verschlüsselung	11:35	12:10	00:35	Georg Kistenberger
Mittagessen	12:10	13:00	00:50	Alle
Blickwinkel Anwendungsentwicklung	13:00	13:35	00:35	Patrick Hempeler
Auditing und Compliance	13:35	14:10	00:35	Carsten Hinz
Pause	14:10	14:25	00:15	Alle
Demo auf System z	14:25	14:45	00:20	Christian Daser
Zusammenfassung und Diskussion	14:45	15:15	00:30	Alle

Agenda

- 09:30 Einführung
- 09:45 Migration & Vorteile nach der Migration
- 10:45 Neue Funktionen zur Qualitätssicherung
- 11:15 Pause
- 11:30 Anwendungsoptimierung
- 12:00 Neue Anforderungen an IT-Sicherheit
- 12:45 Mittagspause
- 13:30 DB2 auf Zeitreise, Bi-Temporal Data
- 14:15 Erweiterungen von OLTP bis Warehousing
- 15:30 Pause
- 15:45 IBM DB2 Analytics Accelerator
- 16:00 Unicode in DB2 z/OS
- 16:15 Abschluss und Q&A
- 16:45 Veranstaltungsende

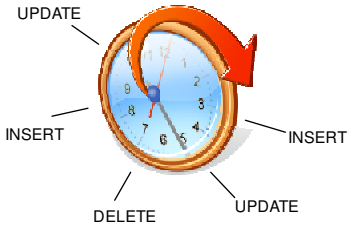
DB2 V10: DB2 auf Zeitreise

Temporal Tables

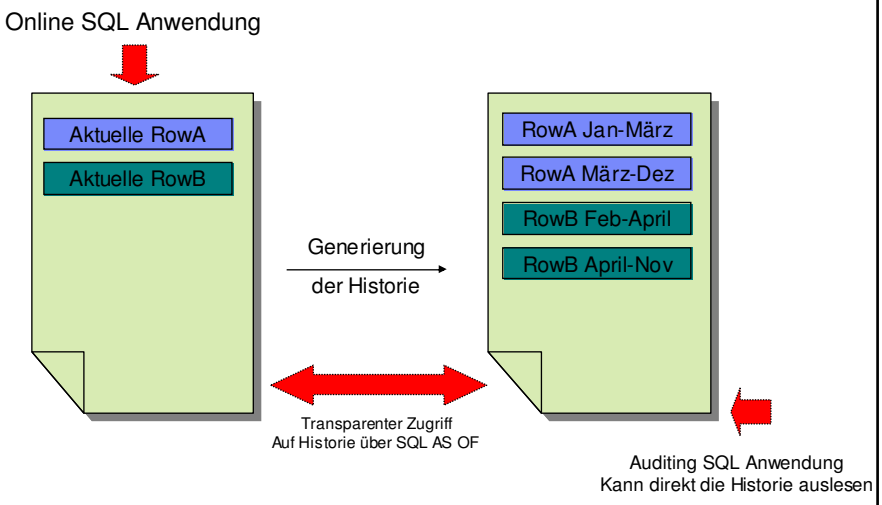
- Inhalte von Tabellen waren bislang ohne zeitlichen Zusammenhang
- In V10 gibt es zwei Erweiterungen
  - Pflege der Historie von Daten (SYSTEM_TIME)
    DB2 legt eine Historisierung von Datensätzen an
  - Zeitliche Spezifikation von Datensätzen (BUSINESS_TIME)
    Anwendung gibt die Gültigkeit von Daten 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



Vereinfachte DDL für SYSTEM_TIME

```
CREATE TABLE KFZ (
          VERTRAGSNUMMER CHAR(4),
          KUNDENNUMMER_ID CHAR(4),
          KASKO_ID CHAR(6),
          JAEHRL_FAHRLEISTUNG 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_FAHRLEISTUNG 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;
```


Beispiele für SYSTEM_TIME

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

Tabelle KFZ

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	T150	18000	2010-12-01	9999-12-30

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

Tabelle KFZ

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	V300	18000	2011-03-01	9999-12-30

Automatisch gepflegte Tabelle KFZ_SYS_HIST

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

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

Tabelle KFZ

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	V300	25000	2011-06-01	9999-12-30

Automatisch gepflegte Tabelle KFZ_SYS_HIST

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

Tabelle KFZ

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	V300	25000	2011-06-01	9999-12-30

Automatisch gepflegte Tabelle KFZ_SYS_HIST

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
```

Tabelle KFZ

VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END

Automatisch gepflegte Tabelle KFZ_SYS_HIST

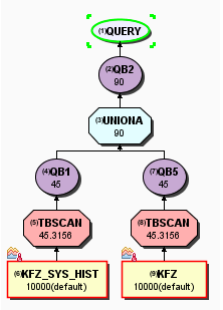
VERTRAG	KUNDE	KASKO	JAEHRL	SYS_START	SYS_END
A123	123	T150	18000	2010-12-01	2011-03-01
A123	123	V300	18000	2011-03-01	2011-06-01
A123	123	V300	25000	2011-06-01	2011-09-15

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

```
T150
```


Komplettes SQL (SYSTEM_TIME)

```
Formatted Query
( SELECT ISV.KFZ.KASKO_ID AS KASKO
  FROM ISV.KFZ
  WHERE ( ISV.KFZ.VERTRAGSNUMMER = 'A123'
          AND ISV.KFZ.SYS_START <= '2010-12-25 00:00:00'
          AND ISV.KFZ.SYS_END > '2010-12-25 00:00:00'
        )
UNION ALL
  SELECT ISV.KFZ_SYS_HIST.KASKO_ID AS KASKO
  FROM ISV.KFZ_SYS_HIST
  WHERE ( ISV.KFZ_SYS_HIST.VERTRAGSNUMMER = 'A123'
          AND ISV.KFZ_SYS_HIST.SYS_START <= '2010-12-25 00:00:00'
          AND ISV.KFZ_SYS_HIST.SYS_END > '2010-12-25 00:00:00'
        )
)
```



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 Tablespaces
- 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_FAHRLEISTUNG         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

KFZ - Vertrag

Max Mustermann
Musterstraße 16
93777 Musterort
Kundennummer: 123

Versicherung GmbH
Firmenstraße 11
33777 Firmenort

01. Januar

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

Kunde schliesst neuen KFZ Vertrag ab

Neuvertrag

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


```
INSERT INTO KFZ (VERTRAGSNUMMER, KUNDENNUMMER_ID, KASKO_ID, JAEHRL, BUS_START, BUS_END)
         VALUES ('A123', '123', 'T150', 20000, '2011-03-01','9999-12-30');
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	9999-12-30

Neuvertrag

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

IBM

KFZ - Änderung

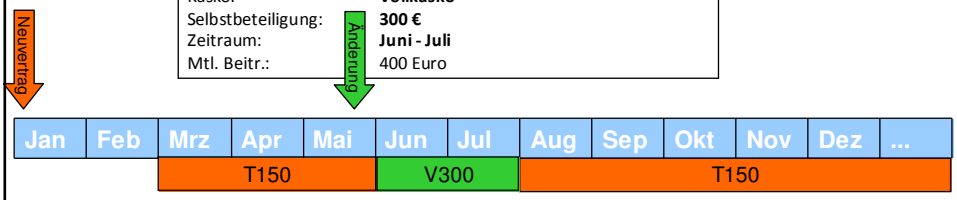
Max Mustermann
Musterstraße 16
93777 Musterort
Kundennummer: 123

Versicherung GmbH
Firmenstraße 11
33777 Firmenort

25. Mai

Vertragsnummer:	**A123**
Wagen:	Mittelklassewagen
Typ:	hsn 0588
Fahrleistung	20.000 km
Kasko:	**Vollkasko**
Selbstbeteiligung:	**300 €**
Zeitraum:	**Juni - Juli**
Mtl. Beitr.:	400 Euro

Für Sommer soll eine Vollkasko existieren



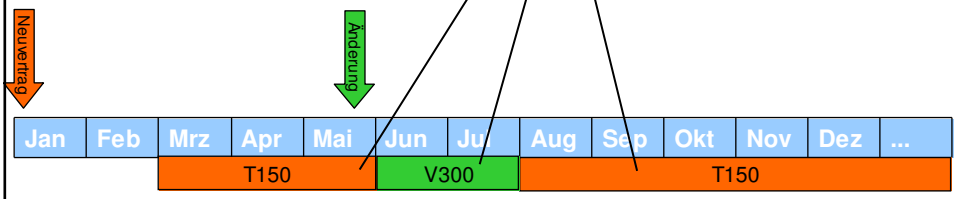
IBM

```
VERTRAG KUNDE KASKO  JAEHRL BUS_START  BUS_END
------- ----- ------ ------ ---------- ----------
A123    123   T150   20000  2011-03-01 9999-12-30
```

```
UPDATE KFZ FOR PORTION OF BUSINESS_TIME FROM '2011-06-01' TO '2011-08-01'
       SET KASKO_ID = 'V300' 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   T150   20000  2011-08-01 9999-12-30
```

SQL Update
SQL Insert
SQL Insert



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					

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					

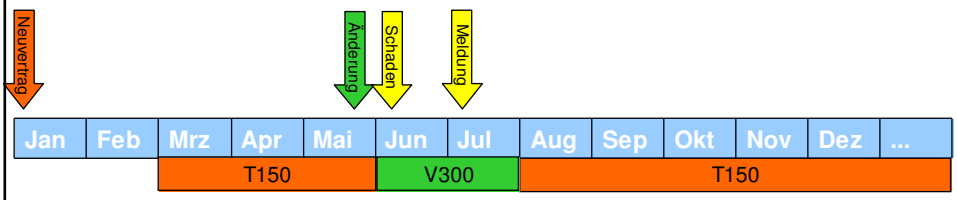
VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END
A123	123	T150	20000	2011-03-01	2011-06-01
A123	123	V300	20000	2011-06-01	2011-08-01
A123	123	T150	20000	2011-08-01	9999-12-30

Existierte für den 7.Juni eine Vollkasko ?

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

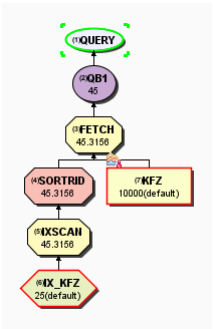
V300

Schaden wird übernommen

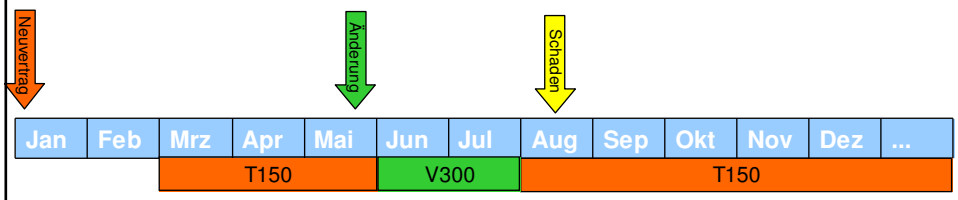


Komplettes SQL (BUSINESS_TIME)

```
Formatted Query
    SELECT ISV.KFZ.KASKO_ID AS KASKO
    FROM ISV.KFZ
    WHERE  ( ISV.KFZ.VERTRAGSNUMMER = 'A123'
             AND ISV.KFZ.BUS_END > '2011-06-07'
             AND ISV.KFZ.BUS_START <= '2011-06-07'
           )
```



Schaden durch Unaufmerk-samkeit am 15.August



KFZ - Schadensmeldung

Max Mustermann
Musterstraße 16
93777 Musterort
Kundennummer: 123

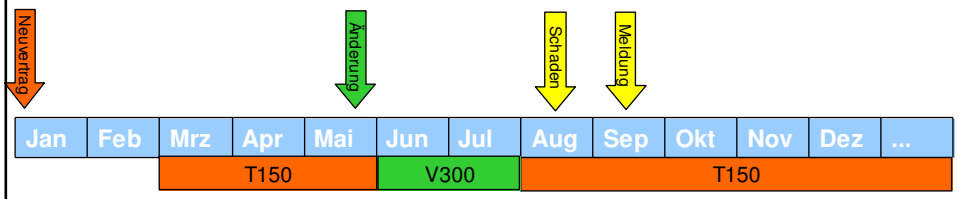
Versicherung GmbH
Firmenstraße 11
33777 Firmenort

15.Sept

Vertragsnummer: **A123**

Bitte um Kostenerstattung
zwecks Schaden vom 15.August

Existierte für den 15.August eine Vollkasko ?



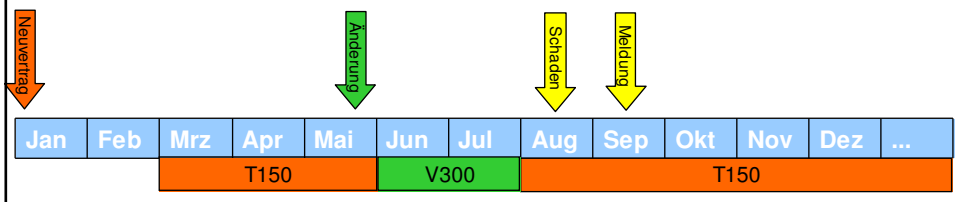
VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END
A123	123	T150	20000	2011-03-01	2011-06-01
A123	123	V300	20000	2011-06-01	2011-08-01
A123	123	T150	20000	2011-08-01	9999-12-30

Existierte für den 15.August eine Vollkasko ?

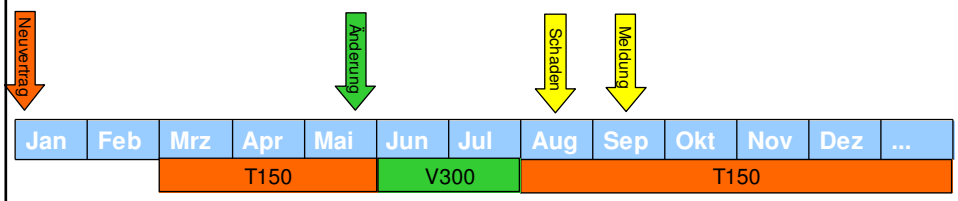
```
SELECT KASKO FROM KFZ FOR BUSINESS_TIME AS OF '2011-08-15' WHERE VERTRAGSNUMMER='A123';
```

T150

Schaden wird abgelehnt



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



IBM

KFZ - Änderung

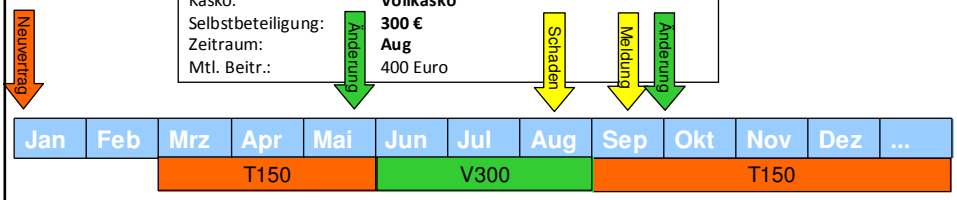
Max Mustermann
Musterstraße 16
93777 Musterort
Kundennummer: 123

Versicherung GmbH
Firmenstraße 11
33777 Firmenort

01. Juli

Vertragsnummer:	**A123**
Wagen:	Mittelklassewagen
Typ:	hsn 0588
Fahrleistung	20.000 km
Kasko:	**Vollkasko**
Selbstbeteiligung:	**300 €**
Zeitraum:	**Aug**
Mtl. Beitr.:	400 Euro

Vollkasko-verlängerung ist in der Tat „liegen-geblieben“ und wird am 01. Oktober rückwirkend prozessiert



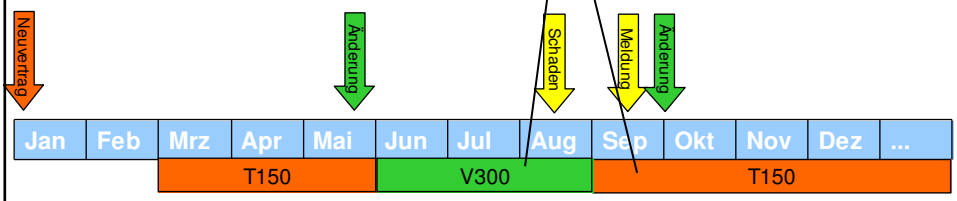
IBM

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	T150	20000	2011-08-01	9999-12-30

```
UPDATE KFZ FOR PORTION OF BUSINESS_TIME FROM '2011-08-01' TO '2011-09-01'
        SET KASKO_ID = 'V300' 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	9999-12-30

SQL Insert
SQL Update



IBM

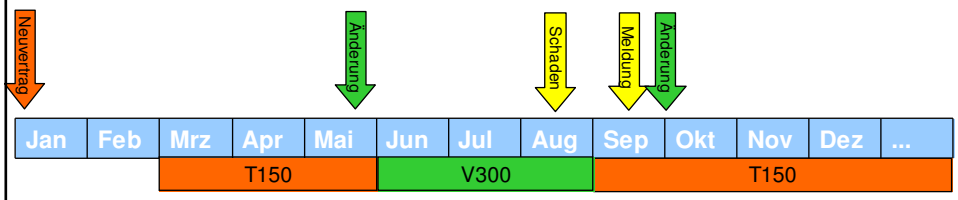
VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END
A123	123	T150	20000	2011-03-01	2011-06-01
A123	123	V300	20000	2011-06-01	2011-08-01
A123	123	V300	20000	2011-08-01	2011-09-01
A123	123	T150	20000	2011-09-01	9999-12-30

Existierte für den 15.August eine Vollkasko ?

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

V300

Schaden wird nun doch übernommen



IBM

Beschwerde

Max Mustermann
Musterstraße 16
93777 Musterort
Kundennummer: 123

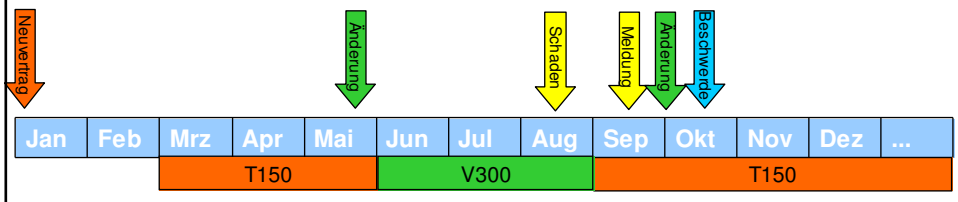
Versicherung GmbH
Firmenstraße 11
33777 Firmenort

15.Oktober

Vertragsnummer: **A123**

Bitte um Klärung des Sachverhalts mit der Falschinformation aus dem CallCenter

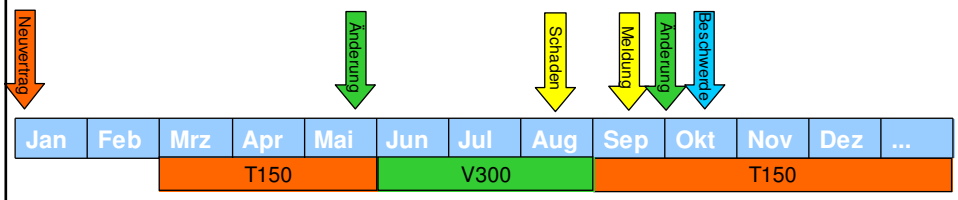
Hat das CallCenter am 15.September die Schadens-meldung korrekt bearbeitet?



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	9999-12-30

Hat das CallCenter am 15.September die Schadens-meldung korrekt bearbeitet?

Kann nicht beantwortet werden



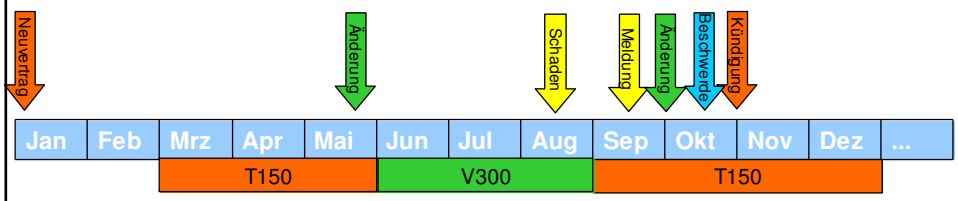
KFZ - Kündigung

Max Mustermann
Musterstraße 16
93777 Musterort
Kundennummer: 123

Versicherung GmbH
Firmenstraße 11
33777 Firmenort

01.November

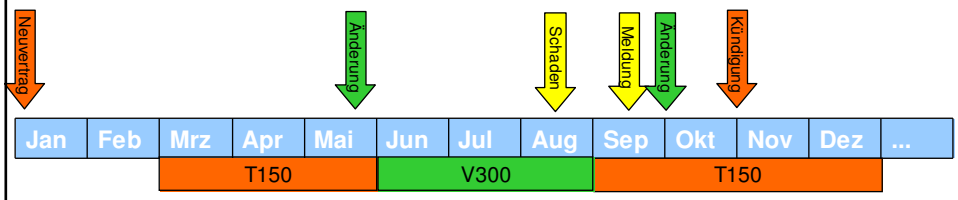
Vertragsnummer: **A123**
Hiermit kündige ich die Versicherung zum Jahresende



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	9999-12-30

```
DELETE FROM KFZ FOR PORTION OF BUSINESS_TIME FROM '2012-01-01' TO '9999-12-30'
          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



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_FAHRLEISTUNG 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_FAHRLEISTUNG 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);
```

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

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	9999-12-30
A123	123	V300	20000	2011-06-01	2011-08-01	2011-05-25	9999-12-30
A123	123	V300	20000	2011-08-01	2011-09-01	2011-10-01	9999-12-30
A123	123	T150	20000	2011-09-01	9999-12-30	2011-10-01	9999-12-30

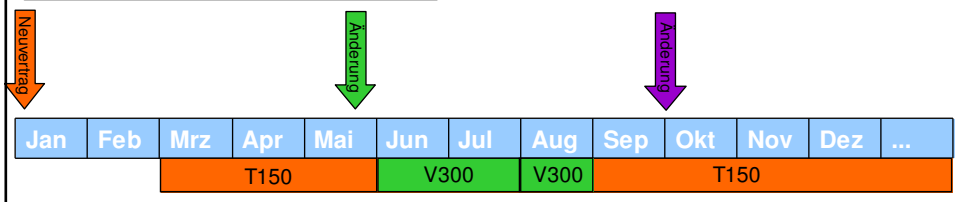
Automatisch gepflegte Tabelle KFZ_SYS_HIST

VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END	SYS_START	SYS_END
A123	123	T150	20000	2011-03-01	9999-12-30	2011-01-01	2011-05-25
A123	123	T150	20000	2011-08-01	9999-12-30	2011-05-25	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 9999-12-30

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 9999-12-30

A123 123 T150 20000 2011-03-01 9999-12-30



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

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	9999-12-30
A123	123	V300	20000	2011-06-01	2011-08-01	2011-05-25	9999-12-30
A123	123	V300	20000	2011-08-01	2011-09-01	2011-10-01	9999-12-30
A123	123	T150	20000	2011-09-01	9999-12-30	2011-10-01	9999-12-30

Automatisch gepflegte Tabelle KFZ_SYS_HIST

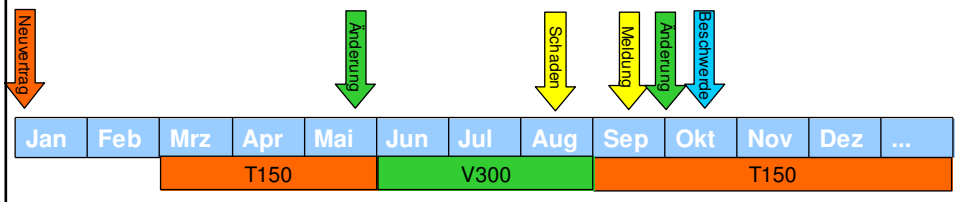
VERTRAG	KUNDE	KASKO	JAEHRL	BUS_START	BUS_END	SYS_START	SYS_END
A123	123	T150	20000	2011-03-01	9999-12-30	2011-01-01	2011-05-25
A123	123	T150	20000	2011-08-01	9999-12-30	2011-05-25	2011-10-01

Hat das CallCenter am 15.September die Schadens-meldung korrekt bearbeitet?

```
SELECT KASKO FROM KFZ FOR BUSINESS_TIME AS OF '2011-08-15'
                      FOR SYSTEM_TIME AS OF '2011-09-15'
                      WHERE VERTRAGSNUMMER='A123';
```

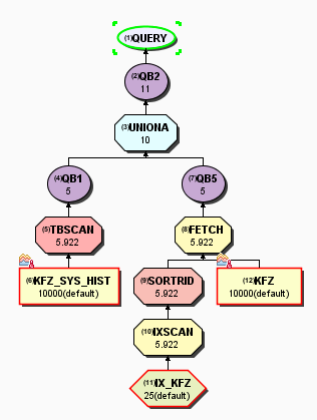
T150

Callcenter hatte aufgrund damaliger Sachlage richtig entschieden



Komplettes SQL (Bi-Temp)

```
Formatted Query
( SELECT ISV.KFZ.KASKO_ID AS KASKO
  FROM ISV.KFZ
  WHERE ( ISV.KFZ.VERTRAGSNUMMER = 'A123'
          AND ISV.KFZ.BUS_END > '2011-06-07'
          AND ISV.KFZ.BUS_START <= '2011-06-07'
          AND ISV.KFZ.SYS_START <= '2011-09-15 00:00:00'
          AND ISV.KFZ.SYS_END > '2011-09-15 00:00:00'
        )
UNION ALL
  SELECT ISV.KFZ_SYS_HIST.KASKO_ID AS KASKO
  FROM ISV.KFZ_SYS_HIST
  WHERE ( ISV.KFZ_SYS_HIST.VERTRAGSNUMMER = 'A123'
          AND ISV.KFZ_SYS_HIST.SYS_START <= '2011-09-15 00:00:00'
          AND ISV.KFZ_SYS_HIST.BUS_START <= '2011-06-07'
          AND ISV.KFZ_SYS_HIST.BUS_END > '2011-06-07'
          AND ISV.KFZ_SYS_HIST.SYS_END > '2011-09-15 00:00:00'
        )
)
```



TIMESTAMP(12)

- Die Genauigkeit des Timestamp wird auf 12 Stellen erweitert

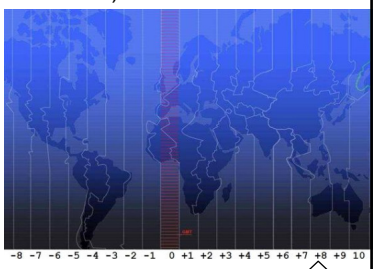
  yyyy-mm-dd-hh.mm.ss.nnnnnnnnnnnn

- Special Register CURRENT TIMESTAMP

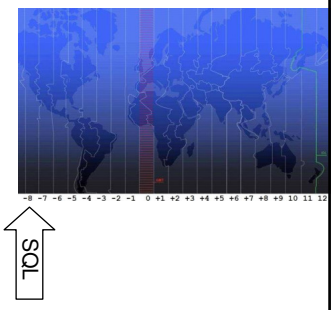
- JDBC Treiber arbeitet in der Regel mit TIMESTAMP(9)
  - Cast nach VARCHAR möglich

DB2 auf Zeitreise

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



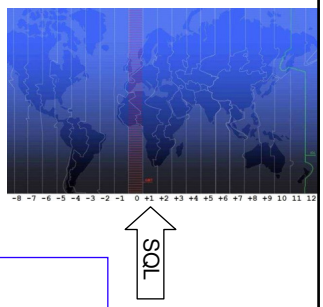
- ± *th:tm*, where
  - ± positives oder negatives Offset
  - *th* Stundenunterschied
  - *tm* Minutenunterschied
- TIMESTAMP WITH TIMEZONE
  - Year-Month-Day-Hour.Minutes.Seconds time zone
  - 2007-11-05-08.00.00-08:00




```
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
```

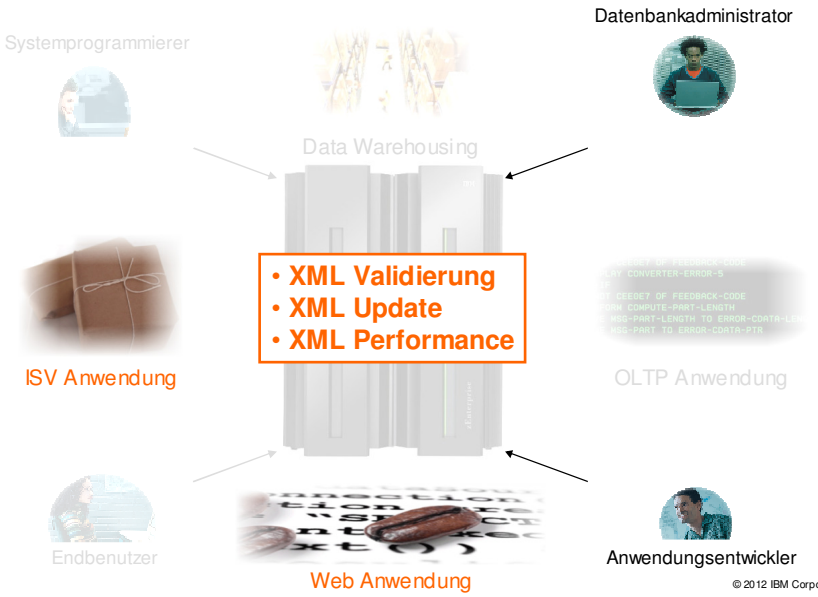
Special Registers



- CURRENT TIMEZONE
  - Unterschied zwischen UTC und DB2 Server
- SESSION TIME ZONE
  - Zeitzone der Anwendung Format '±*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
```

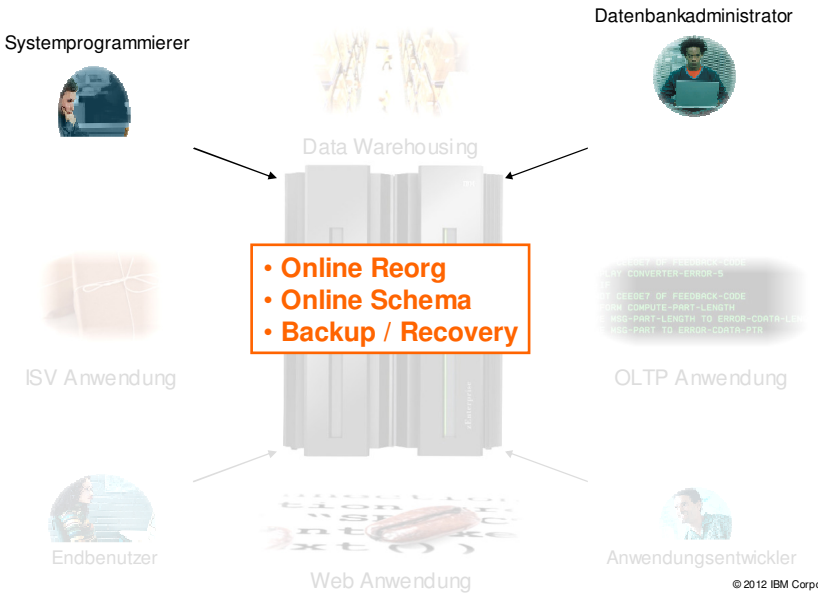

DB2 V10: XML



Agenda

09:30	Einführung
09:45	Migration & Vorteile nach der Migration
10:45	Neue Funktionen zur Qualitätssicherung
11:15	Pause
11:30	Anwendungsoptimierung
12:00	Neue Anforderungen an IT-Sicherheit
12:45	Mittagspause
13:30	DB2 auf Zeitreise, Bi-Temporal Data
14:15	Erweiterungen von OLTP bis Warehousing
15:30	Pause
15:45	IBM DB2 Analytics Accelerator
16:00	Unicode in DB2 z/OS
16:15	Abschluss und Q&A
16:45	Veranstaltungsende

DB2 V10: Utilities



Utility Neuerungen: Online Reorg

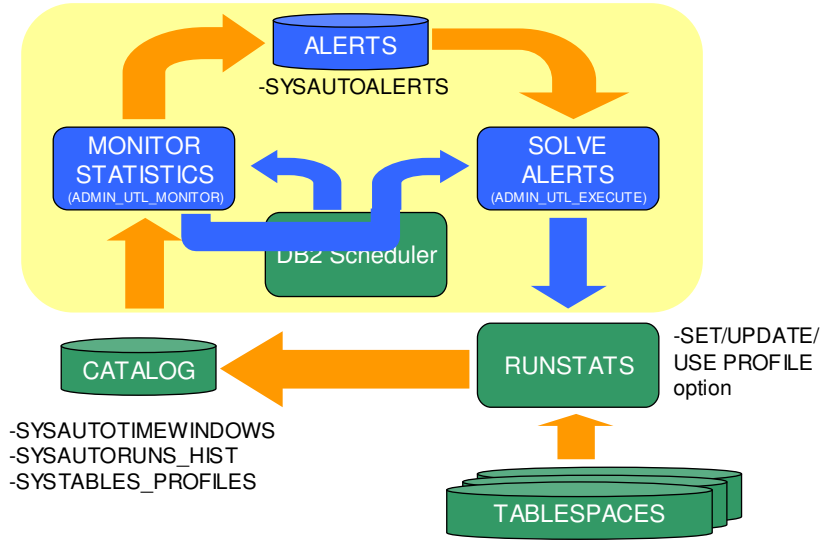
- FORCE Option
  - READERS|ALL|NONE
- AUX Option
  - Einbeziehung von LOB Objekte
- Für alle Catalog und Directory Objekte
  - Ausnahme: SYSUTILX
- Rowformat Option: BRF oder RRF
  - V9: PK85881
- Statistik Update raus aus der SWITCH Phase

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

```
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
STATSTIME = 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
```

AutoStats design



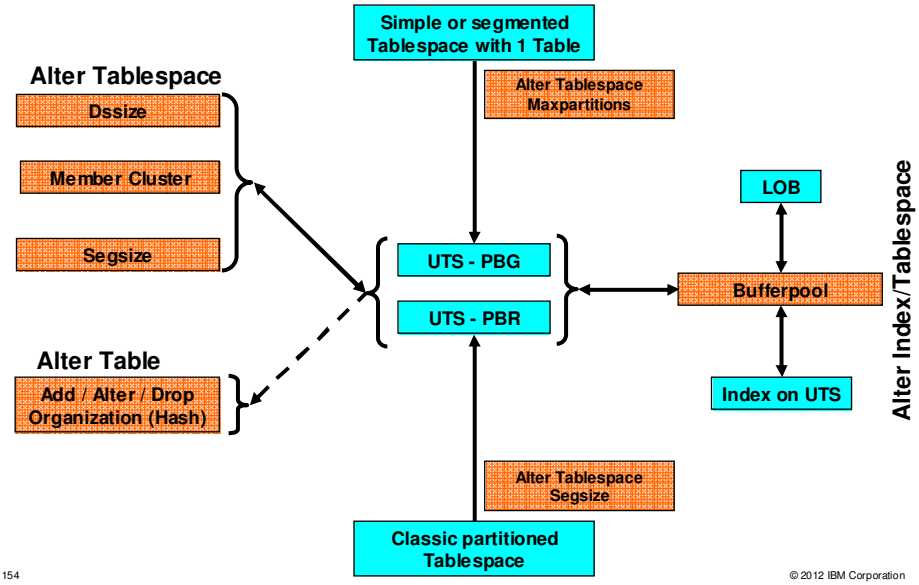
Online Schema Evolution

- 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 Orginal 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
- DSSIZE > 64GB – APAR PM42175

Online Schema Evolution

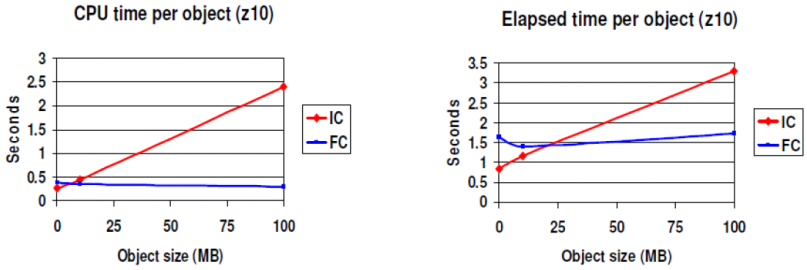


Utility Neuerungen: FlashCopy

- 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

The following REORG INDEX control statement reorganizes the index spaces associated with table space DSN8S81E and creates a FlashCopy image copy of the index.

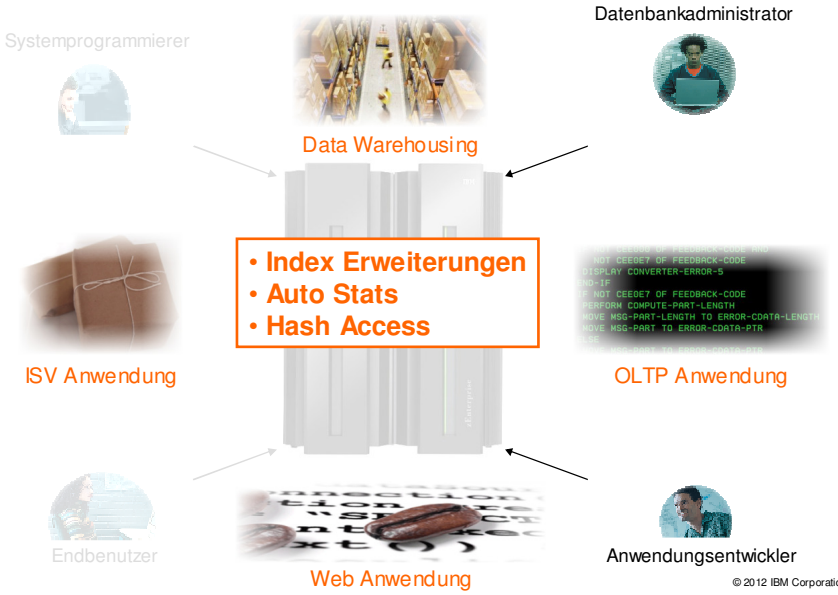
```
//SYSADMA JOB (ACCOUNT),'NAME',NOTIFY=&SYSUID
//*
//UTIL EXEC DSNUPROC,SYSTEM=VA1A,UID='TEMP',UTPROC=''
//DSNUPROC.SYSREC DD DSN=SYSOPS.DSNAME,
// DISP=(NEW,DELETE),
// SPACE=(CYL,(20,20),RLSE),
// UNIT=SYSDA,VOL=SER=SCR03
//DSNUPROC.SYSUT1 DD DSN=SYSOPS.SYSUT1,
// DISP=(NEW,DELETE,DELETE),
// SPACE=(CYL,(9,90),RLSE),
// UNIT=SYSDA,VOL=SER=SCR03
//DSNUPROC.SYSIN DD *
LISTDEF COPY_LIST INCLUDE INDEXSPACES TABLESPACE DSN8D81A.DSN8S81E PARTLEVEL ALL
TEMPLATE SCOPY UNIT(SYSDA) DISP(NEW,CATLG,DELETE)
DSN(DSNT1.&DB..&TS..CPY1.D&TIME.)
TEMPLATE FCOPY UNIT(SYSDA) DISP(NEW,CATLG,DELETE)
DSN(DSNFC.&DB..&TS..P&PA..D&TIME.)
REORG INDEX LIST COPY_LIST SHRLEVEL REFERENCE FLASHCOPY YES
FCCOPYDDN(FCOPY) COPYDDN(SCOPY)
```



Utility Neuerungen: Point in Time Recovery

- BACKOUT Option im Gegensatz zu Recovery mit Image Copy und den Logsätzen
  - Nur log records für backout berücksichtigen
  - Indices dürfen nicht spezifiziert werden
- VERIFYSET Option
  - YES, verifiziert, dass alle abhängigen Objekte zur Base table, wie XML, LOBs, Historie-Tabelle ebenfalls spezifiziert wurden
- ENFORCE Option
  - YES, setzt CHKP / ACHKP falls abhängige Objekte fehlen.

DB2 V10: Performance und Automation



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 „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 Resourcen

- 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

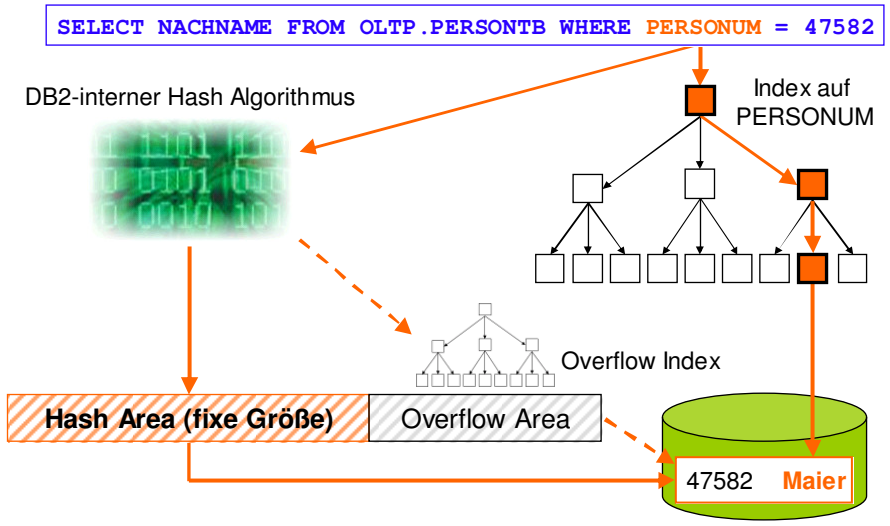
```
CREATE UNIQUE INDEX I1 ON KFZ (VERTRAGSNR) INCLUDE (KUNDENNR, KASKO)
```

- Alternativ

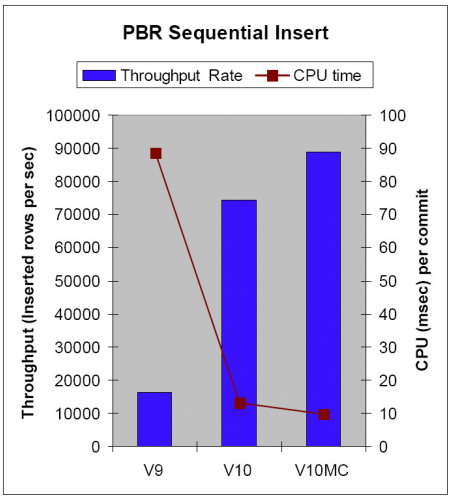
```
ALTER INDEX I1 ADD INCLUDE COLUMN (KUNDENNR); ...
DROP INDEX I2;
```

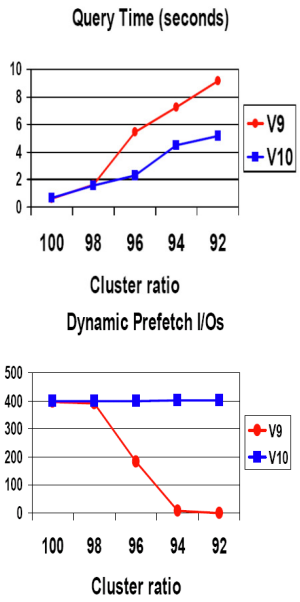
- Index I1 geht in rebuild-pending state

Index-Zugriffe sind nicht immer der effizienteste Weg



Sequentielle Inserts & Data Scans





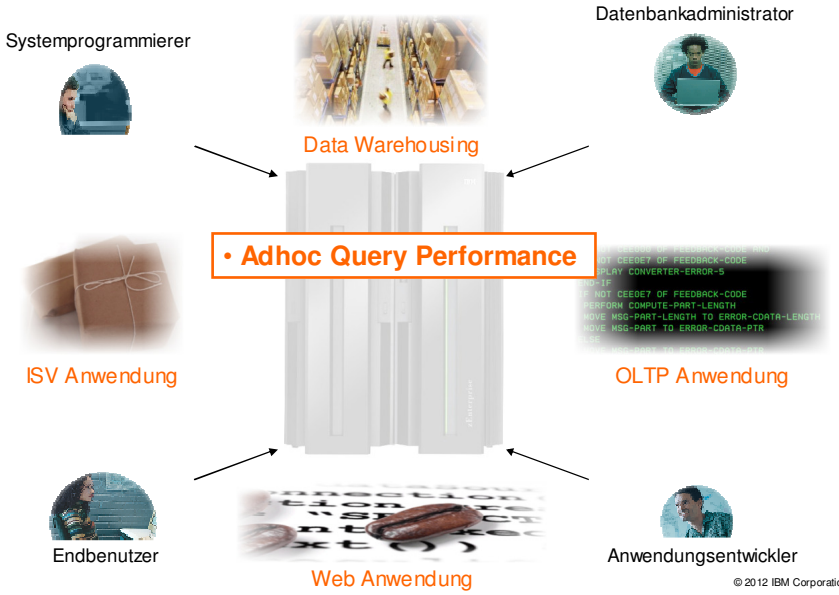
DB2 in Synergie mit zEnterprise

- DB2 Performance, Skalierbarkeit
- zEnterprise
  - 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
- IDAA – IBM DB2 Analytics Accelerator

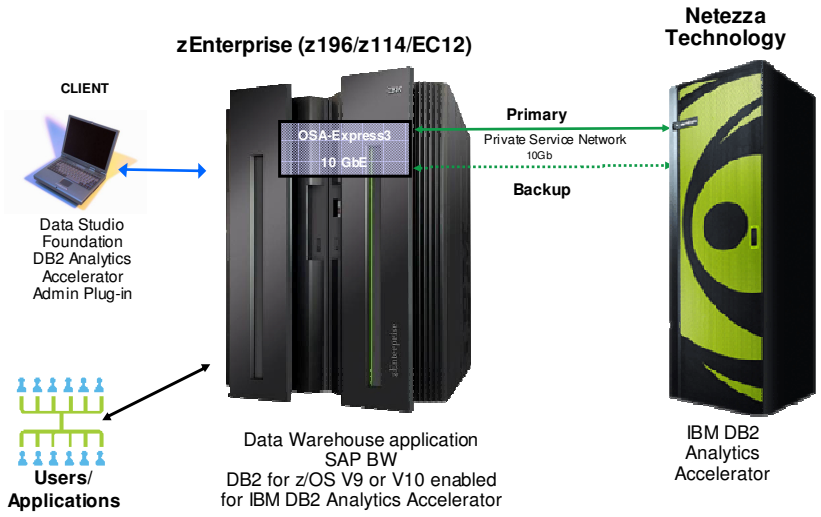
Agenda

09:30	Einführung
09:45	Migration & Vorteile nach der Migration
10:45	Neue Funktionen zur Qualitätssicherung
11:15	Pause
11:30	Anwendungsoptimierung
12:00	Neue Anforderungen an IT-Sicherheit
12:45	Mittagspause
13:30	DB2 auf Zeitreise, Bi-Temporal Data
14:15	Erweiterungen von OLTP bis Warehousing
15:30	Pause
15:45	IBM DB2 Analytics Accelerator
16:00	Unicode in DB2 z/OS
16:15	Abschluss und Q&A
16:45	Veranstaltungsende

IBM DB2 Analytics Accelerator



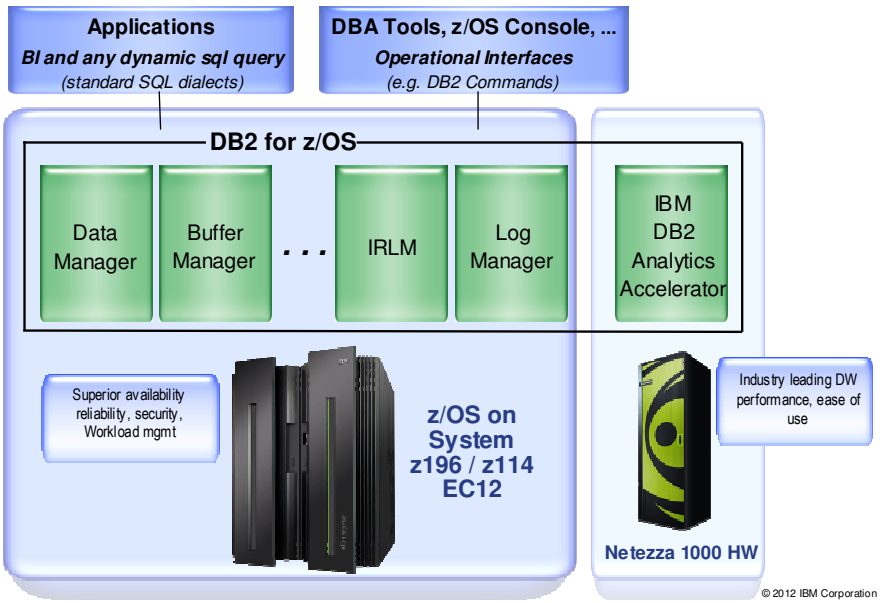
IBM DB2 Analytics Accelerator V3



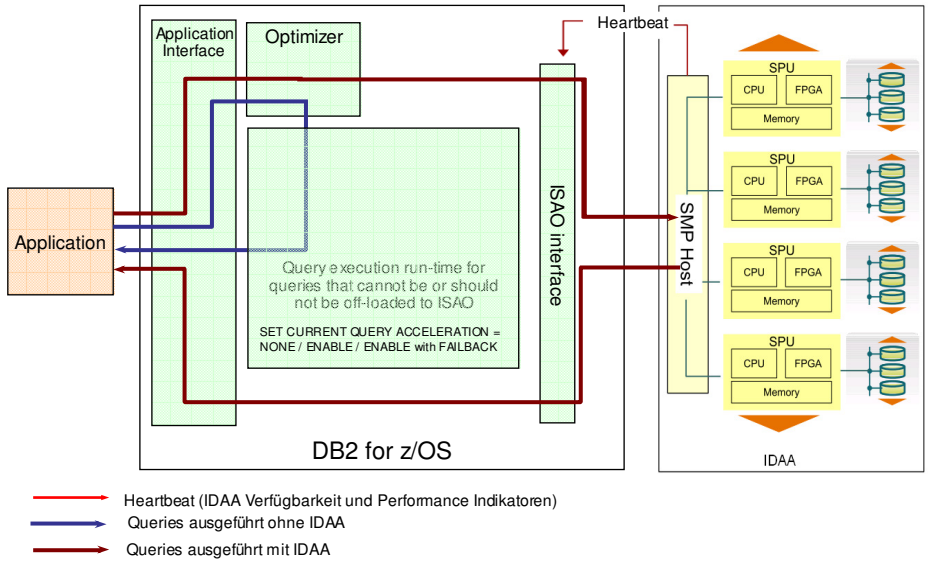
IBM DB2 Analytics Accelerator for z/OS

- **Kerngedanke: Query-Acceleration & Ad-hoc Query Performance**
  - Beschleunigung langlaufender Datenbankabfragen
  - Ad-hoc Query Performance im Sekundenbereich
- **Nutzung von State-of-the-Art Netezza Technologie**
  - Ursprung im Themenkomplex BI & Analytics
  - Appliance like Accelerator
  - Einfache Installation und Inbetriebnahme (1 – 2 Tage)
  - Massiv parallele Verarbeitung durch starke Datenverteilung über eine große Anzahl von kleinen Disks
    - Sehr effiziente Datenkompressionsraten (90%)
  - Mandantenfähigkeit durch Nutzung von unterschiedlichen DB2 z/OS Systemen durch eine Netezza Appliance
- **OLTP meets OLAP**
  - Optimale Kombination vom besten Datenbanksystem für Transaktionsverarbeitung mit dem besten analytischen System
  - Tiefe Integration in DB2 z/OS Architektur
  - DB2 z/OS behält die Ownership der Daten
  - Transparenz gegenüber Applikationen (z.b. auch SAP BW)
    - Applikationen adressieren nachwievor DB2 z/OS
  - Administration und Wartung mittels DB2 z/OS Umgebung
    - Integrative Tools
- **Verbesserung des Preis/Performance Verhältnisses für analytischen Workload**
  - Potential für System z TCO Verbesserung durch Query Offload

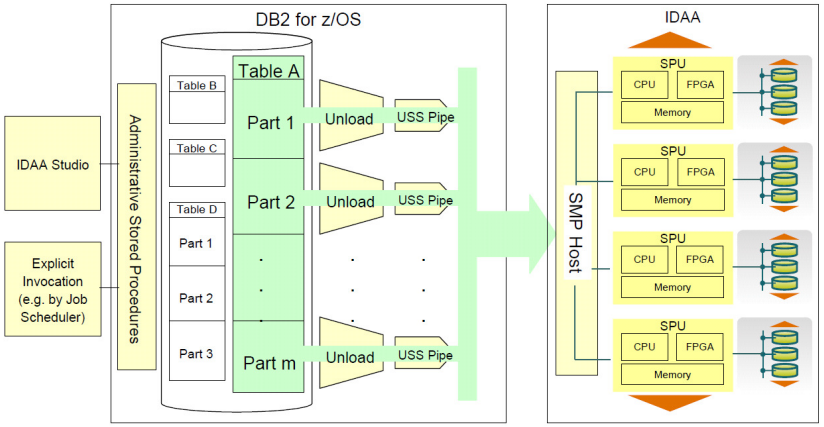
Tiefe DB2 Integration mit zEnterprise



Verarbeitungsablauf einer SQL Abfrage



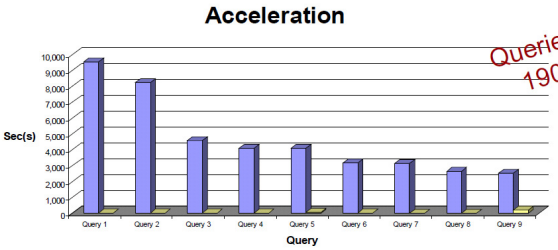
Laden der Daten



- Partitions belonging to the same table can be loaded in parallel
  - User-defined degree of parallelism
- Updates are done on a per-table or per-partition level

V3 – near realtime replication

Beta Programm Ergebnisse!



Queries running up to 1908 times faster

			DB2 Only		DB2 with IDAA		Times Faster
Query	Total Rows Reviewed	Total Rows Returned	Hours	Sec(s)	Hours	Sec(s)	
Query 1	2,813,571	853,320	2:39	9,540	0.0	5	1,908
Query 2	2,813,571	585,780	2:16	8,220	0.0	5	1,644
Query 3	8,260,214	274	1:16	4,560	0.0	6	760
Query 4	2,813,571	601,197	1:08	4,080	0.0	5	816
Query 5	3,422,765	508	0:57	4,080	0.0	70	58
Query 6	4,290,648	165	0:53	3,180	0.0	6	530
Query 7	361,521	58,236	0:51	3,120	0.0	4	780
Query 8	3,425,29	724	0:44	2,640	0.0	2	1,320
Query 9	4,130,107	137	0:42	2,520	0.1	193	13

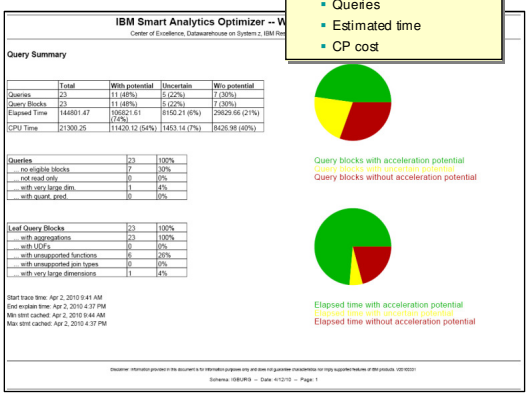
Gibt es ein Workload Potential?

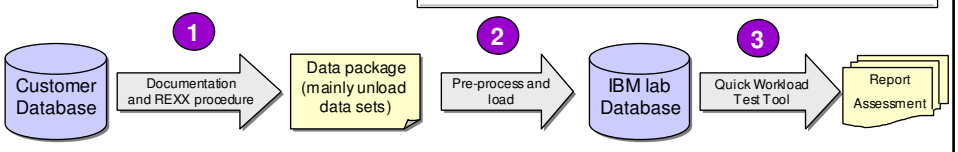
- Langlaufende SQL Abfragen?
  - Dynamic SQL!
- Komplexe SQL Abfragen die beschleunigt werden sollen?
  - Joins, Group by, Order by, aggregate functions ...
  - Große Tabellen, kleines Resultset
- Projekte die einen offload wegen des CPU Verbrauchs von SQL Abfragen erfordern?
- Projekte denen eingeschränkte Mainframe Resourcen zur Verfügung stehen?
- Abläufe bei denen das Batch Zeitfenster ein Engpaß ist und SQL Optimierung ggf. helfen könnte?

Quick Workload Test

- **Customer**
  - Collecting information from dynamic statement cache, supported by step-by-step instruction and REXX script (small effort for customer)
  - Uploading compressed file (up to some MB) to IBM FTP server
- **IBM / Center of Excellence**
  - Importing data into local database
  - Quick analysis based on known DB2 Analytics Accelerator capabilities

Report for a first assessment:
- Acceleration potential for
  - Queries
  - Estimated time
  - CP cost

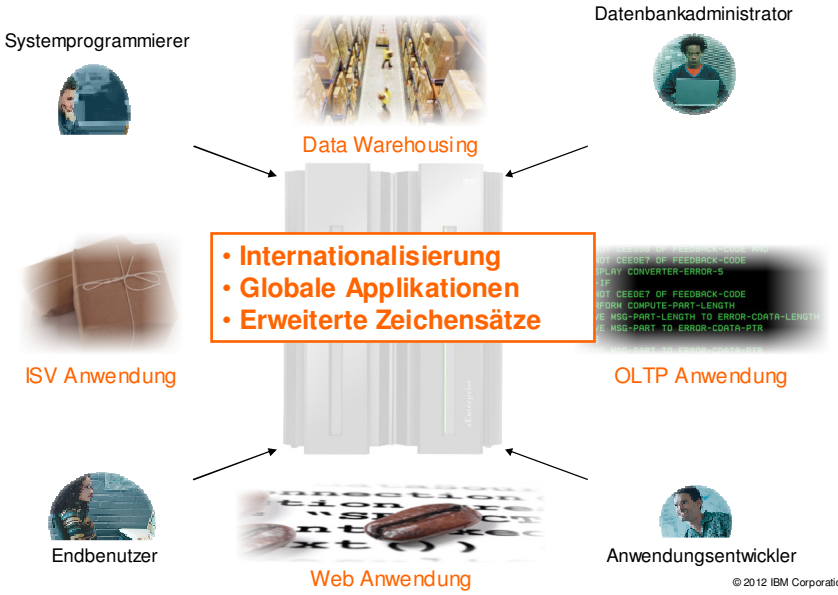




Agenda

09:30	Einführung
09:45	Migration & Vorteile nach der Migration
10:45	Neue Funktionen zur Qualitätssicherung
11:15	Pause
11:30	Anwendungsoptimierung
12:00	Neue Anforderungen an IT-Sicherheit
12:45	Mittagspause
13:30	DB2 auf Zeitreise, Bi-Temporal Data
14:15	Erweiterungen von OLTP bis Warehousing
15:30	Pause
15:45	IBM DB2 Analytics Accelerator
16:00	Unicode in DB2 z/OS
16:15	Abschluss und Q&A
16:45	Veranstaltungsende

Unicode im DB2 z/OS



Agenda

- Unicode Implementierung im DB2
- Allgemeine Betrachtungen zur Einführung von Unicode
- Bildung einer neuen GSE Arbeitsgruppe BUNID „Internationalisierung von legacy Anwendungen“

DB2 z/OS nutzt Unicode!

- **Unicode support seit V7**
- **Unicode Nutzung seit V8**
  - Unicode Catalog – UTF-8
  - Unicode Parser – UTF-8
  - Unicode SQL statements
  - Multiple CCSIDs per SQL
- **Immer mehr Kunden nutzen Unicode im DB2**
  - Heute meistens partielle Nutzung; Ausnahme SAP
  - Komplette Umstellungsszenarien erfordern Koexistenz-Strategie
  - Betrachtung geht über die Datenspeicherung hinaus!

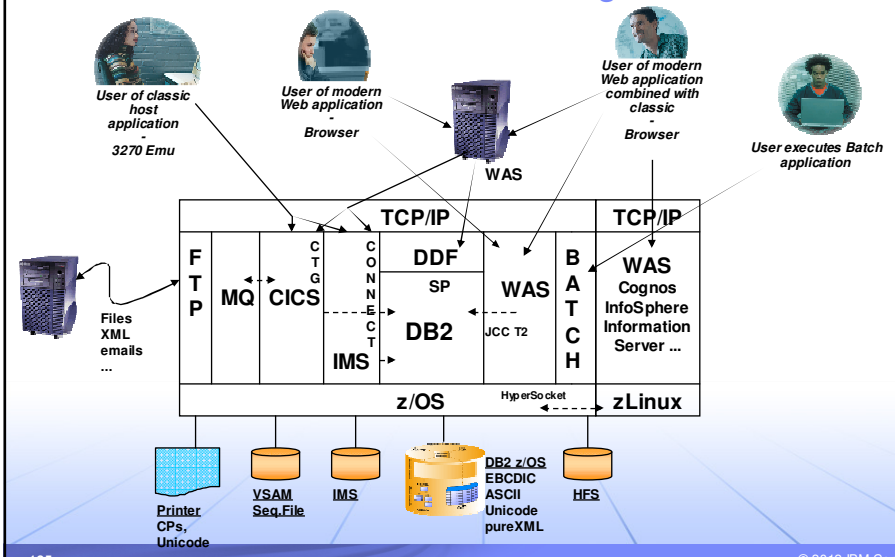
Unicode Implementierung im DB2

- **Unicode Unterstützung für UTF-8 & UTF-16**
  - Systemweite Definition in DSNHDECP
    - USCCSID = 367: special 7-bit ASCII (Definition 'for SBCS')
    - UMCCSID = 1208: **UTF-8**
      - 1-4 bytes; disk space optimization
    - UGCCSID = 1200: **UTF-16**
      - 2 or 4 bytes → Compression
      - 2 bytes for extended latin character sets
        - > Keine Abhängigkeiten hinsichtlich unterschiedlicher Zeichenlängen
        - > Einfacheres Datenmodell; mapping auf Host-Variablen
      - Supported by other products like COBOL …
      - Supported by ISV applications like SAP …

Unicode Implementierung im DB2

- **Unicode Definition für User-Tables**
  - CREATE DATABASE / TABLESPACE … CCSID **UNICODE**
  - CREATE TABLE …
    - Char/VarChar/CLOB FOR SBCS DATA → **CCSID 367**
    - Char/VarChar/CLOB → **UTF-8 CCSID 1208**
      - Längenangabe bezieht sich auf die Anzahl Bytes
    - Graphic/VarGraphic/DBCLOB → **UTF-16 CCSID 1200**
      - Double-byte basierend, d.h. max. 127 Zeichen
- **General Considerations**
  - Sort, Range predicate dependencies
  - Variable Zeichenlänge → speziell bei UTF-8
  - XML datatype ist per default UTF-8

Architektur zur Internationalisierung



Unicode Voraussetzungen

- **Verlustfreie Unicode Verarbeitung**
  - Alle Komponenten einer Anwendung benötigen Unicode support?!
    - Server
    - Programme
    - Datenbanken
    - Dateien
    - Druckertreiber und Drucker
  - Datenübertragung
    - Alle Partner müssen unicode-fähig sein
  - Unicode per default
    - HTTP, SOAP, HTML, WML, XML, Java, PDFs

Allgemeine Betrachtungen & Fragestellungen – 1 -

- **In welchen Umfang soll Unicode genutzt werden?**
  - Beschränkung auf Unicode subsets
    - Lateinische Zeichen in allen Varianten?
    - Kyrillische, griechische ... Zeichen?
    - Eingabe: Sonderzeichenmenues, customized Tastaturbelegungen, autohotkey Programm ...
    - Prüfroutinen
    - ISO6937 (327 Zeichen); WGL4 (652 Zeichen), XÖV (492 Zeichen), CEN/TC304 EES: Extended European Subset (ISO IEC 10646-1, 3109 Zeichen) ...
    - Vorgaben
  - Beschränkung auf einzelne Use/Business Cases
    - Welche Anwendungsanforderung muss erfüllt werden?
      - Sicherstellen das die Post ankommt? Erfüllung einer Richtlinie? ...
- **Koexistenz von Daten mit ASCII/EBCDIC und Unicode Speicherung**
  - Ggf. Start mit der Datenbank DB2 ≥
    - Anwendung bedient zunächst Untermenge an Zeichen
    - Automatische Konvertierung zwischen DB2 und Anwendung
      - Application encoding scheme / Declare Variable SQL statement
    - Joins zwischen EBCDIC / ASCII und Unicode Tabellen möglich
      - Jedoch keine Unterstützung von RI-Definitionen mit string datatypes
      - Requirement MR012908405 / GGDB208012
    - Encoding scheme support auf Tablespace (Tabellen) Ebene
      - Requirement für Unicode Column-level support bzw. Unicode string datatype
      - MR122011716 / GGUNI11001 ...

Allgemeine Betrachtungen & Fragestellungen – 2 -

- **UTF-8 vs UTF-16**
  - Host languages, Java nutzen UTF-16
  - DB2 unterstützt UTF-8 / UTF-16
    - UTF-8 bedeutet variable Länge (i.d.R. 1 – 3 bytes)
    - UTF-16 i.d.R. „feste“ Länge (2-bytes)
      - vereinfachtes Datenmodell
  - Internet Technologie (WAS, HTML, XML ...) i.d.R. per default UTF-8 based
  - Deutsche Textdokumente i.d.R. mit sehr geringen Speichermehrbedarf in UTF-8

Allgemeine Betrachtungen & Fragestellungen – 3 -

- **Suchen nach diakritischen Zeichen**
  - Mehrfachspeicherung
    - Transliteration & Original ≥
  - Volltextsuche
  - SQL BIF NORMALIZE_STRING
  - Unicode Normalization Services for z/OS ≥
- **Suchfunktionen im IMS DB über SSA (Segment Search Area) sind nur eingeschränkt möglich**
  - Equal ist o.k.; Range Abfragen jedoch nicht
- **Landesspezifische Sortierung**
  - Unicode Collation Services for z/OS
  - SQL BIF COLLATION_KEY
- **Umgebungen die keine codepages kennen (z.B. TSO nutzt PComm Codepage)**
- **3270-Emu / PComm ermöglicht keinen Unicode support**

Allgemeine Betrachtungen & Fragestellungen – 4 -

- **Werden eigene Codepages in Applikationen verwendet?**
  - Selbst-entwickelte Konvertierungsroutinen
  - Umsetzen von Zeichen innerhalb eines Programmes
- **Werden Bit-Daten in string Datentypen abgelegt?**
  - Sollten in CHARACTER FOR BIT DATA oder VAR/BINARY datatypes gespeichert sein
- **Welche codepages werden heute verwendet?**
- **Wie sieht die Anwendungsarchitektur aus?**
- **Welche Programmschnittstellen existieren?**
- **Für welche Komponenten wird Unicode support benötigt?**

Konstituierende Sitzung bei der GAD!

GSE-Working Group
BUNID
Internationalisierung
von Legacy Anwendungen

Kick Off-Tagung
21.11.2011

BUNID Tagungen

- **Wahl des Chairman Boards**
  - Chairman: Claus Weidenfeller – GAD
  - Stellv. Chairman: Dr. Jörg Ortmann – TK
  - IBM Liason: Georg Kistenberger
- **Verabschiedung der neuen GSE Working Group BUNID**
  - Abstract – Statuten
- **Projekterfahrungen**
  - Unicode Einführung bei einem Kunden im öffentlichen Bereich
    - CICS / COBOL / DB2 durch IBM/BP
      - COBOL Umsetzung nach Unicode
      - DB2 Datenbank bereits in Unicode vorhanden
  - Unicode4TKeasy bei der Techniker Krankenkasse
    - Unicodeumsetzung anhand eines Business Cases
- **Produktbetrachtung / Unterstützung für Unicode**
  - Unicode Internals
  - Standardisierung von Unicode Subsets
  - Unicode support in Host Languages
  - Unicode support in Assembler
  - Unicode support beim AFP Druck
- **Erfahrungsaustausch ist „key“ !!!**

GSE BUNID Guide Tagungen

- **Letzte Tagung vom 8. – 10. Oktober bei der TK!**
- **Auswirkungen von Unicode und Codepage-Einstellungen auf Cobol und DB2 - IBM GTS/SWG & TK**
- **z/OS Components and Unicode support - IBM z/OS Dev.**
- **nPA – Auswirkungen auf die IT - Bundesdruckerei**
- **Erfahrungen bei der Umstellung eines Teilprojektes auf Unicode - DBSystel**
- **Konvertierung und Komprimierung von Unicode Daten - Limes Datentechnik**
- **Angewandte z/OS Unicode Services - IBM SO**
- **Unicode – Auswirkung auf Sortierung - IBM BA Student**
- **Global View on Unicode support, Projects and Requirements - IBM SVL**
- **AFP - Unicode Unterstützung - Ricoh**
- **Application Development – zTools support for Unicode - IBM SWG**
- **Verwendung von diakritischen Zeichen mit DataStudio - IBM SWG**

Scenario 1: Unicode im Programm & EBCDIC in der DB

- **Cobol Program Umwandlung in US (CCSID 1140 entspricht 037 plus „€“)**
  DB2 Installation in Deutschland
  Anwendung soll auch für französische Anwender nutzbar sein
  Schrittweise Einführung von Unicode im Programm

COBOL Program:
-> *HVin* *PIC X → DB2 Hostvariable*
-> **HVout, HVproc** **PIC N → DB2 Hostvariable (CCSID 1200)**
-> *Filevar, Outvar* *PIC X → Program variable*

...
-> DECLARE VARIABLE :HVproc, :HVout CCSID 1200
-> SELECT Col3, Col4 INTO :HVout, :HVproc FROM TB1
WHERE Col1 = :HVin
AND Col2 = ‚literalÜ' → SQL Parser 1140 -> 1208

...
-> GET Filevar FROM file1 → No conversion
-> HVproc = ‚WertÜ' → Conversion 1200 <-> 1140
-> HVproc = Filevar → Conversion 1200 <-> 1140

...
-> MOVE FUNCTION DISPLAY-OF (HVout, 1147) TO Outvar
-> Display (Outvar) →korrekte Darstellung / PComm CCSID 1147

1. *COBOL Compiler with CODEPAGE 1140*
   → CCSID für Source statements und Literale
2. *DB2 Coprocessor NOSQLCCSID*
   DB2 Precompiler CCSID (1140)
3. *BIND with ENCODING(1147)*
   → CCSID für Daten in DB2 Hostvariable
4. *DB2 Runtime*
   Col1 = :HVin → data conv 1141 <-> 1147
   Col2 = ‚literalÜ' → data conv 1141 <-> 1208
   Col3/4 = :HVout/proc → data conv 1141 <-> 1200

DB2 Tabellen
→ EBCDIC encoding für tablespace
→ Daten in CCSID 1141 gespeichert

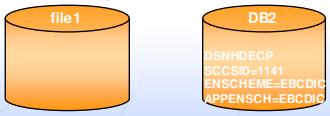
- **Fazit:**
  → MOVE latin_data TO national geht, aber nicht in umgekehrter Richtung
  → COBOL Intrinsic function erforderlich
  → DECLARE VARIABLE für UTF-16 Definition erforderlich, da National Variable Definition in COBOL und SQL Package EBCDIC encoding definitiert ist (ansonsten sqlcode -189).

Scenario 2: Unicode in der DB & EBCDIC im Programm

- **Cobol Program Umwandlung in US (CCSID 1140 entspricht 037 plus „€“)**
- Schrittweise Einführung von Unicode in der Datenbank
- **Anwendung soll auch für französische Anwender nutzbar sein**

COBOL Program:
-> HVin, HVout, HVproc PIC X → DB2 Hostvariable
-> Filevar, Outvar PIC X → Program variable
...
-> DECLARE VARIABLE :HVproc CCSID 1140
-> SELECT Col3, Col4 INTO :HVout, :HVproc FROM TB1
WHERE Col1 = :HVin
AND Col2 = ‚literalÜ' → SQL Parser 1140 -> 1208
...
-> GET Filevar FROM file1 → No conversion
...
-> HVproc = ‚WertÜ' → No conversion required
-> HVproc = Filevar → No conversion required
...
-> MOVE HVout TO Outvar →No conversion
-> Display (Outvar) →korrekte Darstellung / PComm CCSID 1147

1. *COBOL Compiler with CODEPAGE 1140*
   → CCSID für Source statements und Literale
2. *DB2 Coprocessor NOSQLCCSID*
   DB2 Precompiler CCSID (1140)
3. *BIND with ENCODING(1147)*
   → CCSID für Daten in DB2 Hostvariable
4. **DB2 Runtime**
   Col1/3 = :HVin/out → data conv 1208 <-> 1147
   Col2 = ‚literalÜ' → data conv 1208 <-> 1208
   Col4 = :HVproc → data conv 1208 <-> 1140



-> DB2 Tabellen
→ Unicode encoding für tablespace
→ Daten in CCSID 1208 gespeichert

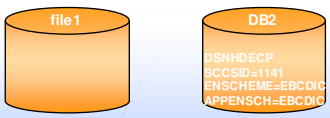
- **Fazit:**
  → DB2 unterstützt UTF-8 und UTF-16 abhängig von verwendeten datatypes in der DB2 Tabelle
  → SQL Package pro Land; abgestimmt mit PComm Einstellung

Scenario 3: Unicode everywhere

- **Cobol Program Umwandlung & DB2 Installation in Deutschland**
- Durchgängige Unicode Implementierung zur Unterstützung eines erweiterten Zeichensatzes – diakritische Zeichen

COBOL Program:
-> Outvar PIC N → Program variable (CCSID 1200)
-> HVin, HVout, HVproc PIC N → DB2 Hostvariable (CCSID 1200)
-> Filevar PIC X → Program variable
...
-> SELECT Col3, Col4 INTO :HVout, :HVproc FROM TB1
WHERE Col1 = :HVin
AND Col2 = ‚literalÜ' → SQL Parser 1141 -> 1208
...
-> GET Filevar FROM file1 → No conversion
-> HVproc = ‚WertÜ' → Conversion 1200 <-> 1141
-> HVproc = Filevar → Conversion 1200 <-> 1141
...
-> MOVE HVout TO Outvar
-> Display (Outvar) → Browser / GUI mit Unicode support

1. **COBOL Compiler with CODEPAGE 1141**
   → CCSID für Source statements und Literale
2. **DB2 Coprocessor NOSQLCCSID**
   DB2 Precompiler CCSID (1141)
3. **BIND with ENCODING(1200)**
   → CCSID für Daten in DB2 Hostvariable
4. **DB2 Runtime**
   Col1 = :HVin → data conv 1208 <-> 1200
   Col2 = ‚literalÜ' → data conv 1208 <-> 1208
   Col4 = :HVout/proc → data conv 1208 <-> 1200



-> DB2 Tabellen
→ Unicode encoding für tablespace
→ Daten in CCSID 1208 gespeichert

- **Fazit:**
  → DB2 und SQL Package encoding in Unicode
  → COBOL National Variablen Deklarationen
  → PComm/3270 Interface wird ersetzt durch Browser/GUI front-end

ICAO 9303 / ISO 7501

– *http://www.hasbrouck.org/documents/ICAO9303-pt2.pdf*

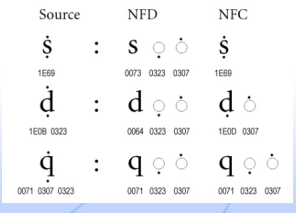
APPENDIX 3 to Section III

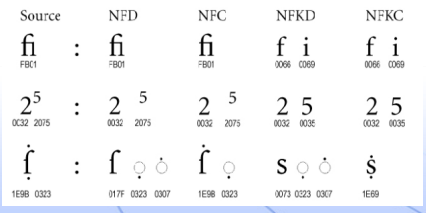
TRANSLITERATIONS RECOMMENDED FOR USE BY STATES

Sequence number	*National character*	*Description*	*Recommended transliteration*
A.	**Transliteration of multinational characters**		
1	Á	A acute	A
2	À	A grave	A
3	Â	A circumflex	A
4	Ä	A diaeresis	AE
5	Ã	A tilde	A
6	Ă	A breve	A
7	Å	A ring	AA

Normalization

Form	Description
Normalization Form D (NFD)	**Canonical Decomposition**
Normalization Form C (NFC)	**Canonical Decomposition, followed by Canonical Composition**
Normalization Form KD (NFKD)	**Compatibility Decomposition**
Normalization Form KC (NFKC)	**Compatibility Decomposition, followed by Canonical Composition**





Quelle: http://unicode.org/reports/tr15/

Scenario 1: Unicode in der DB & EBCDIC im Programm

- Cobol Program Umwandlung in US (CCSID 1140 entspricht 037 plus „€“)
- **Schrittweise Einführung von Unicode in der Datenbank**
- Anwendung soll auch für französische Anwender nutzbar sein

COBOL Program:
-> HVin, HVout, HVproc PIC X → DB2 Hostvariable
-> Filevar, Outvar PIC X → Program variable
...
-> DECLARE VARIABLE :HVproc CCSID 1140
-> SELECT Col3, Col4 INTO :HVout, :HVproc FROM TB1
WHERE Col1 = :HVin
AND Col2 = ‚literalÜ‘ → SQL Parser 1140 -> 1208
...
-> GET Filevar FROM file1 → No conversion
...
-> HVproc = ‚WertÜ‘ → No conversion required
-> HVproc = Filevar → No conversion required
...
-> MOVE HVout TO Outvar →No conversion
-> Display (Outvar) →korrekte Darstellung / PComm CCSID 1147

1. *COBOL Compiler with CODEPAGE 1140*
   → CCSID für Source statements und Literale
2. *DB2 Coprocessor NOSQLCCSID*
   DB2 Precompiler CCSID (1140)
3. *BIND with ENCODING(1147)*
   → CCSID für Daten in DB2 Hostvariable
4. **DB2 Runtime**
   Col1/3 = :HVin/out → data conv 1208 <-> 1147
   Col2 = ‚literalÜ‘ → data conv 1208 <-> 1208
   Col4 = :HVproc → data conv 1208 <-> 1140

-> DB2 Tabellen
→ Unicode encoding für tablespace
→ Daten in CCSID 1208 gespeichert

- Fazit:
- → DB2 unterstützt UTF-8 und UTF-16 abhängig von verwendeten datatypes in der DB2 Tabelle
- → SQL Package pro Land; abgestimmt mit PComm Einstellung

199

Scenario 2: Unicode everywhere

- Cobol Program Umwandlung & DB2 Installation in Deutschland
- **Durchgängige Unicode Implementierung zur Unterstützung eines erweiterten Zeichensatzes – diakritische Zeichen**

Change
COBOL Program:
-> Outvar PIC N → Program variable (CCSID 1200)
-> HVin, HVout, HVproc PIC N → DB2 Hostvariable (CCSID 1200)
-> Filevar PIC X → Program variable
...
-> SELECT Col3, Col4 INTO :HVout, :HVproc FROM TB1
WHERE Col1 = :HVin
AND Col2 = ‚literalÜ‘ → SQL Parser 1141 -> 1208
...
-> GET Filevar FROM file1 → No conversion
-> HVproc = ‚WertÜ‘ → Conversion 1200 <-> 1141
-> HVproc = Filevar → Conversion 1200 <-> 1141
...
Change
-> MOVE HVout TO Outvar
-> Display (Outvar) → Browser / GUI mit Unicode support

1. **COBOL Compiler with CODEPAGE 1141**
   → CCSID für Source statements und Literale
2. **DB2 Coprocessor NOSQLCCSID**
   DB2 Precompiler CCSID (1141)
3. **BIND with ENCODING(1200)**
   → CCSID für Daten in DB2 Hostvariable
4. **DB2 Runtime**
   Col1 = :HVin → data conv 1208 <-> 1200
   Col2 = ‚literalÜ‘ → data conv 1208 <-> 1208
   Col4 = :HVout/proc → data conv 1208 <-> 1200

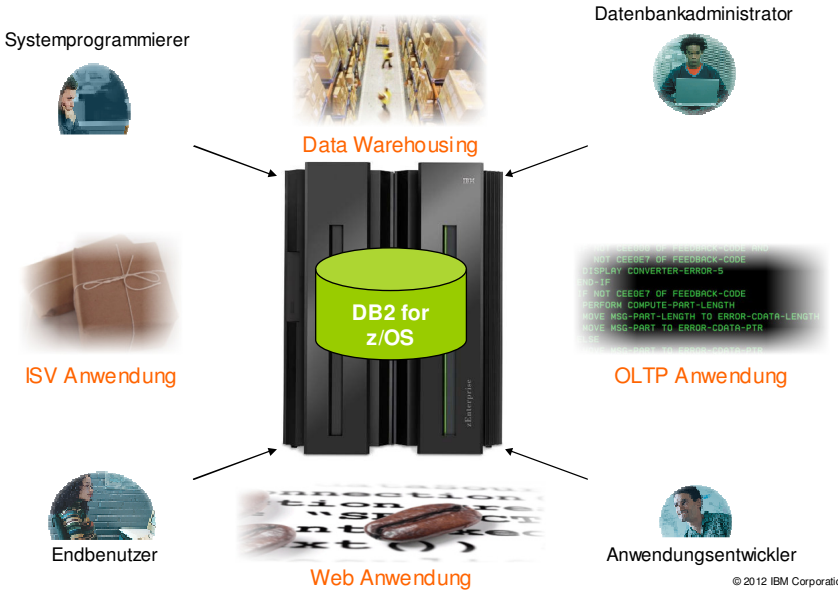
Change
Change
Change

-> DB2 Tabellen
→ Unicode encoding für tablespace
→ Daten in CCSID 1208 gespeichert

- Fazit:
- → DB2 und SQL Package encoding in Unicode
- → COBOL National Variablen Deklarationen
- → PComm/3270 Interface wird ersetzt durch Browser/GUI front-end

200

DB2 V10: Was ist für Sie dabei ?

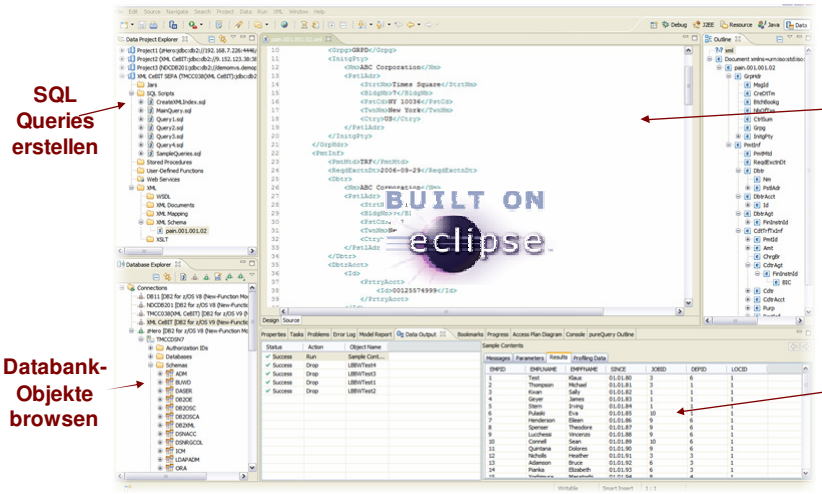


Agenda

09:30	Einführung
09:45	Migration & Vorteile nach der Migration
10:45	Neue Funktionen zur Qualitätssicherung
11:15	Pause
11:30	Anwendungsoptimierung
12:00	Neue Anforderungen an IT-Sicherheit
12:45	Mittagspause
13:30	DB2 auf Zeitreise, Bi-Temporal Data
14:15	Erweiterungen von OLTP bis Warehousing
15:30	Pause
15:45	IBM DB2 Analytics Accelerator
16:00	Unicode in DB2 z/OS
16:15	Abschluss und Q&A
16:45	Veranstaltungsende

IBM Data Studio 3.1.1

kostenfreie Komponente! Empfohlen für Problembearbeitung bei Optimizer Themen



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

DB2 V10 Packaging

DB2 V10 for z/OS
Prog.-Nr. 5605-DB2

Basis Komponenten
standard FMIDs

- DB2 Base
- IRLM V2.3
- IMS Attach
- MQSeries/Listener
- REXX Language
- RACF Auth Exit

z/OS Application Connectivity
kostenfreies feature

- JDBC (T2/T4) Universal Driver

QMF V10
kostenpflichtiges feature

- QMF Enterprise Edition
  - QMF for TSO/CICS
  - QMF for WebSphere
  - QMF for Windows
  - QMF HPO
- QMF Classic Edition
  - QMF for TSO/CICS

DB2 Accessories Suite
kostenfreies Produkt
Prog.-Nr. 5697-Q02

- Spatial Support
- Omnifind Text Search Server
- International Components for Unicode

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 → Information Management
  - Newsletter db2educ@de.ibm.com
  - DB2 V10 Kurs CV31xxxx – 4 Tage

DB2 V10 – Verfügbarkeit / Informationen im Web

- DB2 Daten
  - V10 GA / Allgemeine Verfügbarkeit: **22. Oktober 2010**
  - V8 End-of-Service: **30. April 2012**
  - V9 End-of-Service: **27.Juni 2014**

    http://www-01.ibm.com/software/data/support/lifecycle/
- DB2 V10 Web page

  http://www.ibm.com/software/data/db2/zos/db2-10/

  http://www.ibm.com/software/data/db2/zos/index.html
- IBM DB2 Information Management Tools V10 Compatibility

  http://www-01.ibm.com/support/docview.wss?uid=swg21409518
- DB2 Information Center

  http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp?topic=/com.ibm.db29.doc/db2prodhome.htm
- Redbooks / Whitepapers
  - Temporal Data Management in DB2 z/OS

    http://publibz.boulder.ibm.com/zoslib/pdf/A_Matter_of_Time_-_DB2_zOS_Temporal_Tables_-_White_Paper_v1.4.1.pdf
  - DB2 V10 – Proven, simplified and cost effective

    http://publibz.boulder.ibm.com/zoslib/pdf/db2tll_for_zos.pdf
  - DB2 V10 for z/OS – Technical Overview

    http://www.redbooks.ibm.com/abstracts/sg247892.html

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