Willkommen zum „IBM DB2 Newsletter“

Liebe Leserinnen und Leser,

ja ich weiß, lang lang ist es her, das der letzte Newsletter herausgekommen ist. Aber nun ist es wieder soweit. Die erste Ausgabe für 2014 ist fertig.

Der Frühling hat inzwischen Einzug gehalten, nachdem der Winter mehr oder weniger übersprungen wurde. Ostern steht vor der Tür, die Sommerzeit ist aktiviert und eine Stunde verloren gegangen. Dafür ist es Abends aber länger hell. Damit nicht noch weitere Stunden sinnlos verloren gehen haben wir hier wieder Wissen für Sie gebündelt. Viel Spaß beim Lesen auf der Terasse, im Büro.

Für Fragen und Anregungen unsere Kontaktadresse: db2news@de.ibm.com.

Ihr TechTeam

Inhaltsverzeichnis

Artikel: Neues in DB2 V10.5

Zusätzlich zu der BLU Acceleration gibt es in DB2 V10.5 auch noch folgende Neuerungen:

- Unterstützung von Tabellen mit längerer Datensatzlänge als die Pagesize ermöglicht
- Ausschluss von NULL-Werten bei Indizes
- Anlegen UNIQUE constraint auf NULL Werte
- Index auf Spaltenfunktionen
- 3 teiliger Name bei Federated Datenobjekten
- Verbessertes HADR Monitoring (Bereits in V10.1 gab es eine grundlegende Änderung der Ausgabe beim Kommando `db2pd -hadr -db <DBNAME>`, mehr dazu in der nächsten Ausgabe)

Extended Row Size

- der Datenbank Parameter EXTENDED_ROW_SIZE muss auf ENABLE gesetzt werden
- Die maximal Länge für Datensätze in Tabellen mit extended row size enabled darf 1048319 bytes nicht überschreiten.
- Damit eine Tabelle extended Row Size verwenden kann, muss mindestens eine VARCHAR/VARGRAPHIC Spalte mit einer Länger > 1 vorhanden sein
- Ob Extended Row Size für eine Tabelle aktiviert ist oder nicht, kann durch folgende Spalte in SYSCAT.TABLES abgefragt werden. Die 2te Spalte enthält Informationen über den Prozentsatz der extended rows.

```
EXTENDED_ROW_SIZE               SYSIBM    CHARACTER                     1      0 No
PCTEXTENDEDROWS                 SYSIBM    REAL                          4      0 No
```

- Während der Insert-/Update Operationen werden Teile des Datensatzes (VARCHAR und VARGRAPHIC Felder) als LOBS ausgelagert, falls die Länge des Datensatzes die Pagesize übersteigt
- Aufgrund der Auslagerung als LOB, kann der Zugriff auf diese ausgelagerten Daten langsamer sein
- Dies ist besonders sinnvoll für Tabellen mit vielen bzw. langen VARCHAR Felder.
- Wird dies mit einer Column Organized Tabelle ausprobiert, kommt folgender Fehler:

```
db2 create table djs.test "(col1 integer, col2 varchar(4000), col3 varchar(4000)) in
    TSP_DJS_4K"
DB21034E  The command was processed as an SQL statement because it was not a
valid Command Line Processor command.  During SQL processing it returned:
SQL0670N  The statement failed because the row size of the resulting table
would have exceeded the row size limit. Row size limit: "3920". Table space
name: "TSP_DJS_4K". Resulting row size: "4000".  SQLSTATE=54010

Default table organization             (DFT_TABLE_ORG) = COLUMN
```

Ausschluss von NULL-Werten

- Für dieses Feature gibt es eine neue EXCLUDE NULL KEYS Klausel im CREATE INDEX Statement.
- Indizes können dadurch extrem verkleinert werden.
- Beispiel:

```
CREATE INDEX MYSCHEMA.MYIDX1 on MYSCHEMA.MYTABLE (myCOL1) EXCLUDE NULL KEYS
```

  - Dieser Index passt gut für viele Datensätze mit NULL-Werten
  - Macht den Index extrem klein und damit die Suche schneller

Unique Constraints auf NULL-Werte

- Auch hier wird das Konstruct des `EXCLUDE NULL KEYS` Klausel im CREATE INDEX Statement verwendet.
- Bisher war ein UNIQUE Index nicht möglich für Spalten, die NULLable waren, da bei bisherigen Constraints die NULL Werte einheitlich – als ein Wert – behandelt wurden und somit nur ein Datensatz NULL enthalten durfte.
- Mit dem neuen Construct, können beliebig viele NULL Datensätze enthalten sein.
- Beispiel:

```
CREATE UNIQUE INDEX MYSCHEMA.MYIDX2 on MYSCHEMA.MYTABLE (myCOL1) EXCLUDE NULL KEYS
```

  - Dieser Index ermöglicht mehr als ein NULL Wert in den Datensätzen

Index auf Spaltenfunktionen

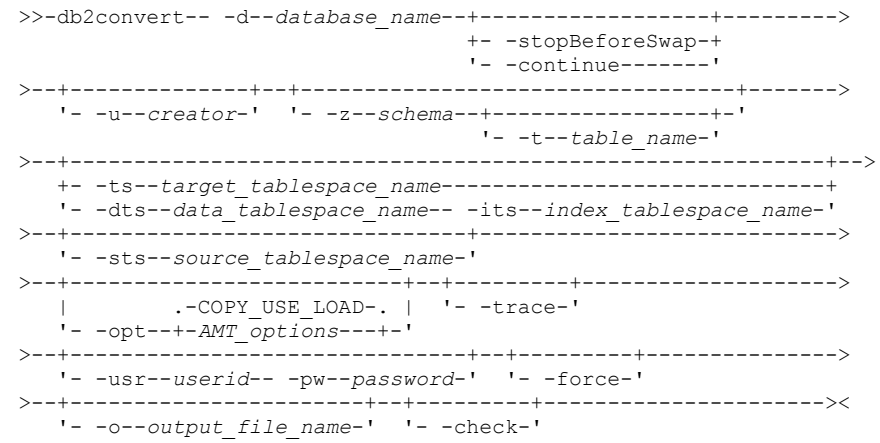
- Vor DB2 10.5 konnten Expressions nur dann indiziert werden, wenn diese mit einer Generated Column „maskiert" wurden.
- Mit der DB2 V10.5 können die Expressions (Funktionen) direkt beim `CREATE INDEX` Statement angegeben werden. Weiterhin sind Expressions auch in der INCLUDE list bei Unique Indizes erlaubt.
- Dies vereinfacht die Tabellen-Wartung.
- Einschränkungen darauf sind unter anderem
  - Die Rückgabewerte der Expressions müssen skalar sein, d.h. kein XML oder LOBs/LONGs
  - In der Expression muss mindestens eine Spalte enthalten sein
  - Expression darf keine Subqueries oder Fehler Tolerante Nested-Table beinhalten
  - Keine Verwendung von Aggregationsfunktionen bzw. User Defined Funktionen (UDFs), nicht deterministische Funktionen, oder Funktionen mit externen Aktivitäten
- Beispiel:
  ```
  CREATE INDEX MYSCHEMA.MYIDX3 on MYSCHEMA.MYTABLE (upper(myCOL1), myCOL2+myCOL3)
  ```

Vereinfachter Zugriff auf Federated Objekte

- Vor der DB2 V10.5 mussten für Federated Objekte Nicknames erstellt werden.
- Mit DB2 V10.5 kann der Nickname weggelassen werden (Wrapper, Server und Usermapping sind jedoch weiterhin notwendig).
- SYNTAX: `SELECT * FROM <SERVER>.<REMOTE_SCHEMA>.<REMOTE_OBJECT>`

TechTipp: Migration von Row- zu Column-organized Tabelle

Die Umstellung von row nach column organized für eine Tabelle oder alle Tabelle durchzuführen, kann das Tool db2convert verwendet werden. Der Syntax sieht wie folgt aus:



Um dieses Tool auszuführen, wird SQLADM oder DBADM benötigt. Da dieses Tool intern mit der Funktion ADMIN_MOVE_TABLE arbeitet, wird execute Recht für eben diese benötigt. Weiterhin wird noch CREATE TABLE benötigt, sowie SELECT auf der Quell-Tabelle.

Folgende Tabellen können nicht mit diesem Tool migriert werden:

- Range Clustered Tabellen
- Typed Tabellen
- Materialized Query Tabellen
- Temporäre Tabellen
- Tabellen in nicht AUTOMATIC Tablespaces
- Tabellen mit generierten Spalten bzw. LOBs (BLOB, CLOB, DBCLOB, XML)

Rangepartitionierte Tabellen, Multi-Dimensional und ITC Tabellen müssen mit der force Option migriert werden.

Trigger und Indizes müssen nach der Migration angelegt werden, während Foreign Key und Check Constraints mit der NOT ENFORCED Option angelgt wurden.
Während der Migration wird temporär der doppelte Platz der zu migrierenden Tabelle benötigt (Quell- und Zieltabelle).

Um alle Tabellen der Datenbank zu konvertieren wird das Tool wie folgt aufgerufen:
```
db2convert -d <DBNAME>
```

Eine einzelne Tabelle wird durch die Optionen -z für Schema und -t für Tabellenname eingeschränkt.
```
db2convert -d <DBNAME> -z <USER> -t <TABELLE> -ts <TABLESPACE>
```

Die Ausgabe sieht zum Beispiel so aus:
```
  Table            NumRows  RowsComm InitSize(MB)  FinalSize(MB)  CompRate(%)  Progress(%)
  ---------------- -------- -------- -------------- -------------- ------------ ---------------
  USER.TABLE1      1500000  0        105.47         0.26            99.76       0

  Total Pre-Conversion Size (MB): 105.47
  Total Post-Conversion Size (MB): 0.26
  Total Compression Rate (Percent): 99.76
```

TechTipp: Load Problem

Neulich wurde ich von einem Kollegen angerufen, bei dem ein LOAD auf einer partitionierten Datenbank wegen Filesystem voll abgebrochen war und
er kam nicht mehr an seine Tabelle ran.

Ein Versuch den Load zu beenden führte zu folgender Fehlermeldung:
```
db2 "load from /dev/null of del terminate into TEST.TABLE nonrecoverable"

  Agent Type     Node     SQL Code     Result
_____________________________________________________________________
  LOAD           000      +00000000    Success.
_____________________________________________________________________
  LOAD           001      -00027902    Init error. Table unchanged.
_____________________________________________________________________
  LOAD           002      +00000000    Success.
_____________________________________________________________________
  LOAD           003      +00000000    Success.
_____________________________________________________________________
  RESULTS:       3 of 4 LOADs completed successfully.
_____________________________________________________________________

Summary of LOAD Agents:

Number of rows read         = 0
Number of rows skipped      = 0
Number of rows loaded       = 0
Number of rows rejected     = 0
Number of rows deleted      = 0
Number of rows committed    = 0

SQL27902N  LOAD RESTART/TERMINATE is not allowed on a table that is not in LOAD PENDING state.
```

Ein erneuter Load Versuch oder eine select auf die Tabelle brachte die Meldung, `SQL0668` mit ReasonCode 3, die Tabelle ist in einem Load Pending Zustand.

Also wurde als nächstes mit load Query der Zustand der Tabelle abgefragt, da war aber alles normal. Bevor nun aber die Tabelle gelöscht wird, den `load terminate` auf der Partition mit dem Problem, im o.g. Beispiel auf Partition 1, ausführen. Löschen sollte nur der allerletzte Ausweg sein.

Redirected Restore als Methode der Migration von V9.7 nach 10.5

Im Rahmen der Datenbank-Migration von V9.7 nach 10.5 wurden verschiedene Möglichkeiten ausprobiert.
Als Aufgabenstellung war:

- Migration auf eine neue Datenbank Version
- Migration in neue Umgebung (neue Hardware - Server und Storage), unter Verwendung neuer Filesystem Strukturen

- Umstellung der Datenbank auf AUTOMATIC Storage
- neue Instanz-User und Datenbank Konventionen

Eine Möglichkeit der Migration mit gleichzeitiger Umstellung wäre der Redirected Restore. In diesem Artikel werde ich daher auf dieses Thema näher eingehen.

Vorgehen

Vereinfacht dargestellt, ist folgendes Vorgehen dazu notwendig.

1. Full OFFLINE Backup der Quell-DB
   ```
   $ db2 backup database MYDB

   Backup successful. The timestamp for this backup image is : 20140304083744

   $ ls -l ID*
   -rw-------    1 db2inst1    dbadm 1801134080 Mar 04 08:38 MYDB.0.db2inst1.NODE0000.CATN0000.20140304083744.001
   ```
2. Erzeugen Redirected Restore SQL-Kommandos
   - Skript erstellen
     ```
     $ db2 restore db MYDB redirect generate script restore.MYDB.ddl
     DB20000I  The RESTORE DATABASE command completed successfully.
     ```
   - Skript überarbeiten
     - Auskommentieren/Entfernen der File Info
     - `USING AUTOMATIC STORAGE` einfügen
3. Ausführen überarbeitetes Skript
   ```
   $ db2 -tvf  restore.MYDB.V10.5.ddl
   ```

Randbemerkungen

- Mittels Redirected Restore kann von DMS nach AUTOMATIC STORAGE umgewandelt werden
- SMS Konvertierung nach Automatic Storage wird nicht unterstützt, s.h. #1.1.Mögliche Probleme
- Restore eines Online-Backups für den Instanz-Upgrade ist nicht möglich, s.h. #1.1.Mögliche Probleme
- Wird bei der Migration auch Instanz-User geändert, sollte folgende Variable gesetzt sein: `db2set DB2_RESTORE_GRANT_ADMIN_AUTHORITIES=YES`
  Auszug aus dem Information Center: If DB2_RESTORE_GRANT_ADMIN_AUTHORITIES is set to ON, and you are restoring to a new or existing database, then you will be granted SECADM, DBADM, DATAACCESS, and ACCESSCTRL authorities.

Script nach der Überarbeitung

```
cat restore.MYDB.V10.5.ddl
-- ******************************************************************************
-- ** automatically created redirect restore script
-- ******************************************************************************
UPDATE COMMAND OPTIONS USING S ON Z ON MYDB_NODE0000.out V ON;
SET CLIENT ATTACH_MEMBER  0;
SET CLIENT CONNECT_MEMBER 0;
-- ******************************************************************************
-- ** automatically created redirect restore script
-- ******************************************************************************
RESTORE DATABASE MYDB
-- USER  <username>
-- USING '<password>'
FROM '/db2/backup'
TAKEN AT 20140304083744
ON '/db2/dat/'
-- DBPATH ON '<target-directory>'
INTO MYDB
NEWLOGPATH '/db2/log/db2inst1/MYDB/'
-- WITH <num-buff> BUFFERS
-- BUFFER <buffer-size>
-- REPLACE HISTORY FILE
-- REPLACE EXISTING
REDIRECT
-- PARALLELISM <n>
WITHOUT ROLLING FORWARD
-- WITHOUT PROMPTING
;
-- ******************************************************************************
-- ** storage group definition
-- **   Default storage group ID                = 0
-- **   Number of storage groups                = 1
-- ******************************************************************************
-- ******************************************************************************
-- ** Storage group name                        = IBMSTOGROUP
```

```
-- **   Storage group ID                        = 0
-- **   Data tag                                = None
-- *****************************************************************************
-- SET STOGROUP PATHS FOR IBMSTOGROUP
-- ON '/db2/dat/db2inst1/MYDB/NODE0000/NEWSTORAGE'
-- ;
-- *****************************************************************************
-- ** table space definition
-- *****************************************************************************
-- *****************************************************************************
-- ** Tablespace name                         = SYSCATSPACE
-- **   Tablespace ID                           = 0
-- **   Tablespace Type                         = System managed space
-- **   Tablespace Content Type                 = All permanent data. Regular table space.
-- **   Tablespace Page size (bytes)            = 4096
-- **   Tablespace Extent size (pages)          = 32
-- **   Using automatic storage                 = No
-- **   Total number of pages                   = 87042
-- *****************************************************************************
SET TABLESPACE CONTAINERS FOR 0
using AUTOMATIC STORAGE;
-- IGNORE ROLLFORWARD CONTAINER OPERATIONS
--USING ( PATH   '/db2/system/db2inst1/MYDB/db2inst1/NODE0000/SQL00001/SQLT0000.0') ;
-- *****************************************************************************
-- ** Tablespace name                         = TSP_TMP16K
-- **   Tablespace ID                           = 4
-- **   Tablespace Type                         = System managed space
-- **   Tablespace Content Type                 = System Temporary data
-- **   Tablespace Page size (bytes)            = 16384
-- **   Tablespace Extent size (pages)          = 16
-- **   Using automatic storage                 = No
-- **   Total number of pages                   = 1
-- *****************************************************************************
SET TABLESPACE CONTAINERS FOR 4
using AUTOMATIC STORAGE;
-- IGNORE ROLLFORWARD CONTAINER OPERATIONS
--USING ( PATH   '/db2/tmp/db2inst1/MYDB/NODE0000/TSP_TMP16K');
-- *****************************************************************************
-- ** Tablespace name                         = TSP_4K
-- **   Tablespace ID                           = 5
-- **   Tablespace Type                         = System managed space
-- **   Tablespace Content Type                 = All permanent data. Regular table space.
-- **   Tablespace Page size (bytes)            = 4096
-- **   Tablespace Extent size (pages)          = 8
-- **   Using automatic storage                 = No
-- **   Total number of pages                   = 301435
-- *****************************************************************************
SET TABLESPACE CONTAINERS FOR 5
using AUTOMATIC STORAGE;
-- IGNORE ROLLFORWARD CONTAINER OPERATIONS
--USING ( PATH   '/db2/dat/db2inst1/MYDB/NODE0000/TSP_4K');
-- *****************************************************************************
-- ** Tablespace name                         = TSP_8K
-- **   Tablespace ID                           = 6
-- **   Tablespace Type                         = System managed space
-- **   Tablespace Content Type                 = All permanent data. Regular table space.
-- **   Tablespace Page size (bytes)            = 8192
-- **   Tablespace Extent size (pages)          = 8
-- **   Using automatic storage                 = No
-- **   Total number of pages                   = 1
-- *****************************************************************************
SET TABLESPACE CONTAINERS FOR 6
using AUTOMATIC STORAGE;
-- IGNORE ROLLFORWARD CONTAINER OPERATIONS
--USING ( PATH   '/db2/dat/db2inst1/MYDB/NODE0000/TSP_8K');
-- *****************************************************************************
-- ** start redirected restore
-- *****************************************************************************
RESTORE DATABASE MYDB CONTINUE;
-- *****************************************************************************
-- ** end of file
```

Mögliche Probleme

Restore von Online Backup nicht möglich

Beim Restore eines Online Backups liefert DB2 die Meldung SQL2547N, d.h. für die Version-Migration mittels Backup-Restore ist ein ONLINE Backup nicht ausreichend.

```
$ db2 backup database MYDB ONLINE include logs
Backup successful. The timestamp for this backup image is : 20140304082025


$ db2 restore db MYDB redirect generate script restore.MYDB.ddl
SQL2547N  The database was not restored because the backup image is from a
previous release and requires rollforward recovery.


$ db2 ? SQL2547N
SQL2547N  The database was not restored because the backup image is from
      a previous release and requires rollforward recovery.

Explanation:

The physical log file formats have changed between these releases,
making rollforward impossible.
```

```
User response:

Restore the database using the version of DB2 used to create the
database and rollforward to the end of logs. Take an off-line full
database backup at this time. This new backup image will be restorable
on the new release of DB2.
```

Fehlermeldung beim Restore

Wenn beim Restore die Meldung bezüglich Bad container path ausgegeben wird, liefert db2diag.log mehr Information zu diesem Thema.

```
$ db2 -tvf  restore.MYDB.V10.5.ddl
UPDATE COMMAND OPTIONS USING S ON Z ON MYDB_NODE0000.out V ON
DB20000I  The UPDATE COMMAND OPTIONS command completed successfully.

SET CLIENT ATTACH_MEMBER  0
DB20000I  The SET CLIENT command completed successfully.

SET CLIENT CONNECT_MEMBER 0
DB20000I  The SET CLIENT command completed successfully.

RESTORE DATABASE MYDB FROM '/software/DB2/backup' TAKEN AT 20140304083744 ON '/db2/dat/' INTO MYDB
NEWLOGPATH '/db2/log/db2inst1/MYDB/' REDIRECT WITHOUT ROLLING FORWARD
SQL1277W  A redirected restore operation is being performed. During a table
space restore, only table spaces being restored can have their paths
reconfigured. During a database restore, storage group storage paths and DMS
table space containers can be reconfigured.
DB20000I  The RESTORE DATABASE command completed successfully.

SET TABLESPACE CONTAINERS FOR 0 using AUTOMATIC STORAGE
SQL0298N  Bad container path.  SQLSTATE=428B2
```

Auszug aus dem diaglog-File

```
2014-03-04-08.59.24.979468+060 E1643464A1163        LEVEL: Warning
PID     : 9306212              TID : 9846           PROC : db2sysc
INSTANCE: db2inst1             NODE : 000           DB   : MYDB
APPHDL  : 0-503                APPID: *LOCAL.db2inst1.140304075924
AUTHID  : db2inst1             HOSTNAME: myhost01
EDUID   : 9846                 EDUNAME: db2agent (MYDB)
FUNCTION: DB2 UDB, database utilities, sqludBMResponse, probe:327
MESSAGE : SQL1277W  A redirected restore operation is being performed. During a
          table space restore, only table spaces being restored can have their
          paths reconfigured. During a database restore, storage group storage
          paths and DMS table space containers can be reconfigured.
DATA #1 : SQLCA, PD_DB2_TYPE_SQLCA, 136 bytes
 sqlcaid : SQLCA     sqlcabc: 136   sqlcode: 1277   sqlerrml: 0
 sqlerrmc:
 sqlerrp : sqludBMR
 sqlerrd : (1) 0x00000000      (2) 0x00000000      (3) 0x00000000
           (4) 0x00000000      (5) 0x00000000      (6) 0x00000000
 sqlwarn : (1)       (2)       (3)       (4)        (5)       (6)
           (7)       (8)       (9)       (10)        (11)
 sqlstate:

2014-03-04-08.59.24.999441+060 I1644628A639         LEVEL: Severe
PID     : 9306212              TID : 9846           PROC : db2sysc
INSTANCE: db2inst1             NODE : 000           DB   : MYDB
APPHDL  : 0-503                APPID: *LOCAL.db2inst1.140304075924
AUTHID  : db2inst1             HOSTNAME: myhost01
EDUID   : 9846                 EDUNAME: db2agent (MYDB)
FUNCTION: DB2 UDB, buffer pool services, sqlbSetPoolCont, probe:1316
MESSAGE : ZRC=0x8002003C=-2147352516=SQLB_BAD_CONTAINER_PATH
          "Bad container path"
DATA #1 : <preformatted>
Permanent system managed table spaces cannot be converted to use automatic storage.
```

Der Hinweis besagt, das SMS Tablespace nicht umgewandelt werden kann in Automatic Storage.

TechTipp:Migration nach UTF-8 Datenfelder

Da der Platzbedarf für Charachter Felder in einer Unicode Datenbank um Faktor 2-4 steigen kann, sollen vor Umstellung einer Datenbank die Felder identifiziert werden, die potentiell vergrößert werden sollten.
Zudem soll die Datensatzbreite gegen die aktuelle Pagesize des verwendeten Tablespace geprüft werden.

Implementierung:

Von folgenden Randbedingungen wird ausgegangen:

- es werden die Datentypen CHARACTER, VARCHAR, CLOB und LONG VARCHAR berücksichtigt
- es wird mit Faktor 2 gerechnet (dies ist ausreichend für Umstellungen auf Umlaute und europäische

Zeichensätze)

Folgende Abfragen werden erstellt:

1. Übersicht aller Felder der entsprechenden Datentypen
2. Übersicht der Tabellen mit aktueller berechneter Datensatzbreite und erweiterter Datensatzbreite bei Verdoppelung der Feldbreite der identifizierten Felder
3. Status der Tabellen bezüglich der Pagesize, es wird geprüft, ob diese durch eine erweiterter Datensatzbreite überschritten wird.
   Hierbei werden auch rang (table) partitioned tables berücksichtigt (andere Referenz auf Tablespace ID)

Folgende Voraussetzungen werden benötigt

- Connect zur Datenbank
- SELECT Berechtigungen auf Catalog Views

Umgebung:

- getestet auf Linux DB2 9.7
- und 10.1

Auruf:

```
db2 -tvf 01_check.sql -z 01_check.log
```

Die Dateien "test_*" dienten zur Anlage der Tabellen und der verschiedenen Konstellationen.

Ausgabe des 01_check.sql

```
db2blu1@db2blu:~> db2 -tf 01_check.sql
------------------------------------------------------------------------------
Column overview:
All columns of type 'CHARACTER','VARCHAR','CLOB' or 'LONG VARCHAR'
------------------------------------------------------------------------------

TABSCHEMA       TABNAME                         COLNAME     LENGTH  LENGTH_DOUBLE
--------------- ------------------------------- ----------- ------- -------------
...
--------------------
Table overview
--------------------

TABSCHEMA       TABNAME                         ORIG_COLLEN NEW_COLLEN
--------------- ------------------------------- ----------- -----------
...
--------------------
Table status
--------------------
TABSCHEMA     TABNAME   PAGESIZE    LONG_NEWLENGTH NEW_COLLEN  STATUS          STATUS_LONG_VARCHAR
------------- --------- ----------- -------------- ----------- --------------- -------------------------
DB2ADMIN      BELEG            4096              0         223 OK              OK
DB2ADMIN      T1               4096              0         209 OK              OK
DB2ADMIN      T11              4096         261600        7696 EXTEND_PAGESIZE LONGVARCHAR_LIMIT_REACHED
DB2ADMIN      T3               4096              0        7992 EXTEND_PAGESIZE OK
DB2ADMIN      T8K              8192              0           4 OK              OK
DB2ADMIN      T9               4096         261600           0 OK              LONGVARCHAR_LIMIT_REACHED

----------------------------------------
Table status on range partitioned tables
----------------------------------------
TABSCHEMA       2                               PAGESIZE    LONG_NEWLENGTH NEW_COLLEN  STATUS          STATUS_LONG_VARCHAR
--------------- ------------------------------- ----------- -------------- ----------- --------------- -------------------------

  0 record(s) selected.
```

Skript: 01_check.sql

```
--- IBM Deutschland GmbH 2014
--- Peter Schurr
--- peter.schurr@de.ibm.com
--- Created: 2014/02/06
--- Task: Check columns which need attention on changing to Unicode
--- tested on DB2 9.7 Linux, DB2 9.5 Linux
--- About Table Overview:
--- LONGVARCHAR_LIMIT_REACHED means LONG VARCHAR colums exceed limit of 32000 K (fix length independent on pagesize)
--- EXTEND_PAGESIZE means other columns than LONG VARCHAR exceed limit of current pagesize after migration to Unicode
echo ------------------------------------------------------------------------------ ;
echo Column overview: ;
echo All columns of type 'CHARACTER','VARCHAR','CLOB' or 'LONG VARCHAR' ;
echo ------------------------------------------------------------------------------ ;
SELECT
  a.tabschema, a.tabname , colname,
  LENGTH,
  LENGTH * 2 as LENGTH_DOUBLE
FROM syscat.columns c, syscat.tables a
WHERE
  a.tabschema NOT LIKE 'SYS%'
AND type = 'T'
AND c.tabschema = a.tabschema
AND c.tabname = a.tabname
AND TYPENAME IN ( 'CHARACTER','VARCHAR','CLOB','LONG VARCHAR' )
ORDER BY a.tabschema, a.tabname
WITH UR
;
---
echo ------------------- ;
echo Table overview ;
echo ------------------- ;
```

```
SELECT
  a.tabschema, a.tabname ,
          SUM(LENGTH + CASE WHEN ( TYPENAME IN ( 'CHARACTER','VARCHAR','CLOB','LONG VARCHAR' ) AND ( NULLS = 'Y' )) THEN 1 ELSE 0 END
+ CASE WHEN TYPENAME IN ( 'VARCHAR')  THEN 4 ELSE 0 END ) AS orig_collen,
          SUM(      CASE WHEN TYPENAME IN ( 'CHARACTER','VARCHAR','CLOB','LONG VARCHAR' ) THEN LENGTH * 2 ELSE LENGTH END + CASE WHEN
( TYPENAME IN ( 'CHARACTER','VARCHAR','CLOB','LONG VARCHAR' ) AND ( NULLS = 'Y' )) THEN 1 ELSE 0 END  + CASE WHEN TYPENAME IN
( 'VARCHAR')  THEN 4 ELSE 0 END         ) AS new_collen
FROM syscat.columns c , syscat.tables a
WHERE
  a.tabschema NOT LIKE 'SYS%'
AND type = 'T'
AND TYPENAME NOT IN ( 'CLOB', 'BLOB' )
AND c.tabschema = a.tabschema
AND c.tabname = a.tabname
GROUP BY a.tabschema, a.tabname
WITH UR ;
---
echo ------------------- ;
echo Table status ;
echo ------------------- ;
WITH t1 ( tabschema, tabname, pagesize, long_newlength, new_collen )
AS (
SELECT a.tabschema, a.tabname , pagesize,
          SUM( CASE WHEN TYPENAME IN ( 'LONG VARCHAR' ) THEN LENGTH * 2 ELSE 0 END + CASE WHEN ( TYPENAME IN ( 'LONG VARCHAR' ) AND (
NULLS = 'Y' )) THEN 1 ELSE 0 END ) AS long_newlength ,
  SUM( CASE WHEN TYPENAME IN ( 'CHARACTER','VARCHAR','CLOB' ) THEN LENGTH * 2 ELSE CASE WHEN TYPENAME IN ( 'LONG VARCHAR' ) THEN 0
ELSE LENGTH  END END + CASE WHEN ( TYPENAME IN ( 'CHARACTER','VARCHAR','CLOB' ) AND ( NULLS = 'Y' )) THEN 1 ELSE 0 END  + CASE WHEN
TYPENAME IN ( 'VARCHAR')  THEN 4 ELSE 0 END       ) AS new_collen
FROM syscat.columns c, syscat.tables a, syscat.tablespaces t
WHERE
  a.tabschema NOT LIKE 'SYS%'
AND type = 'T'
AND c.tabschema = a.tabschema
AND c.tabname = a.tabname
AND a.tbspace = t.tbspace
AND TYPENAME NOT IN ( 'BLOB', 'CLOB' )
GROUP BY a.tabschema, a.tabname, pagesize
)
SELECT tabschema, tabname, pagesize, long_newlength, new_collen ,
  case when new_collen >
            case pagesize
            when 4096 then 4005
            when 8192 then 8101
            when 16384 then 16293
            when 32768 then 32677
            else 0 end
            then 'EXTEND_PAGESIZE' else 'OK' end as status  ,
  case    when long_newlength > 32000
          then 'LONGVARCHAR_LIMIT_REACHED' else 'OK' end as status_long_varchar
  FROM t1
WITH UR
;
echo ---------------------------------------- ;
echo Table status on range partitioned tables;
echo ---------------------------------------- ;
WITH t2 ( tabschema, tabname, pagesize, long_newlength, new_collen )
AS (
SELECT a.tabschema, a.tabname , pagesize,
          SUM( CASE WHEN TYPENAME IN ( 'LONG VARCHAR' ) THEN LENGTH * 2 ELSE 0 END + CASE WHEN ( TYPENAME IN ( 'LONG VARCHAR' ) AND (
NULLS = 'Y' )) THEN 1 ELSE 0 END ) AS long_newlength ,
  SUM( CASE WHEN TYPENAME IN ( 'CHARACTER','VARCHAR','CLOB' ) THEN LENGTH * 2 ELSE CASE WHEN TYPENAME IN ( 'LONG VARCHAR' ) THEN 0
ELSE LENGTH  END END + CASE WHEN ( TYPENAME IN ( 'CHARACTER','VARCHAR','CLOB' ) AND ( NULLS = 'Y' )) THEN 1 ELSE 0 END  + CASE WHEN
TYPENAME IN ( 'VARCHAR')  THEN 4 ELSE 0 END       ) AS new_collen
FROM syscat.columns c, syscat.DATAPARTITIONS a, syscat.tablespaces t
WHERE a.tabschema NOT LIKE 'SYS%'
AND c.tabschema = a.tabschema
AND c.tabname = a.tabname
AND a.tbspaceid = t.tbspaceid
AND TYPENAME NOT IN ( 'BLOB', 'CLOB' )
AND (a.tabschema , a.tabname,1 ) NOT IN ( select x.TABSCHEMA , x.TABNAME, count(1) as ct  from SYSCAT.DATAPARTITIONS x group by
TABSCHEMA , TABNAME having count(1) = 1 )
AND DATAPARTITIONID = 0
GROUP BY a.tabschema, a.tabname, pagesize
)
SELECT   tabschema, tabname, pagesize, long_newlength, new_collen ,
  case    when new_collen >
            case pagesize
            when 4096 then 4005
            when 8192 then 8101
            when 16384 then 16293
            when 32768 then 32677
            else 0 end
            then 'EXTEND_PAGESIZE' else 'OK' end as status  ,
  case    when long_newlength > 32000
          then 'LONGVARCHAR_LIMIT_REACHED' else 'OK' end as status_long_varchar
  FROM t2
WITH UR
;
```

DB2 Regionale Usergroup

Das Nächste Treffen der DeDUG (Deutsche DB2 Usergroup) findet am Freitag, den 09.05.2014 in Ehningen statt.

Die Teilnahme ist kostenlos aber eine Anmeldung ist erforderlich.

Unter diesem Link findet sich auch bereits die Agenda des Meetings.

Chats mit dem Labor

Eine Liste der bereits durchgeführten Chats ist hier zu finden.
Die Präsentationen der Chats können dort angeschaut und heruntergeladen werden.

Schulungen / Tagungen / Informationsveranstaltung

Eine Liste der anstehenden Konferenzen ist hier zu finden.

Newsletter Archiv

Wir haben ein weiteres Archiv für den DB2 Newsletter bei

Alte Ausgaben vom DB2-NL sind nun zum Nachlesen in den Archiven zu finden von:

- Bytec
- Cursor Software AG
- Drap
- ids-System GmbH
- Lis.Tec
- ORDIX.

Anmeldung/Abmeldung

Sie erhalten diesen Newsletter bis zur 3ten Ausgabe ohne Anmeldung. Wenn Sie weiterhin diesen Newsletter empfangen wollen, schicken Sie Ihre Anmeldung mit dem Subject „ANMELDUNG" an db2news@de.ibm.com.

Die Autoren dieser Ausgabe

Sollten Sie Anfragen zu den Artikeln haben, können Sie sich entweder direkt an den jeweiligen Autor wenden oder stellen Ihre Frage über den DB2 NL, denn vielleicht interessiert ja die Antwort auch die anderen DB2 NL Leser.

Doreen Stein	Master Certified IT-Spezialist für DB2 LUW, IBM SWG; Chief-Editor DB2NL, Dipl-Inf (TU). djs@de.ibm.com
Peter Schurr	IT Spezialist für DB2 LUW, IBM SWG

Reviewer und Ideenlieferanten:

Frank Berghofer	IBM SWG
Dirk Fechner	IBM SWG
Peter Schurr	IBM SWG