]>

Rows wider than 32677 bytes

Description

NOTE: This workaround is no longer required as DB2 10.5 row size support was extended to allow you to create a table where its row length can exceed the maximum record length for the page size of the table space.

In DB2 (before DB2 10.5) rows in a table, or a select list to be sorted or temp-ed cannot exceed 32677 bytes. This restriction does not only apply to the actual length of the row but to the statically determined worst case length of the row.

So when you have an Oracle table that exceeds this limit things get tricky.

Solution

There are several options here which depend on the use case:

1. **The table was simply defined in a sloppy way. A telltale is that string columns are of type VARCHAR2(4000).**

   4000 happens to be Oracle's maximum size forVARCHAR2. So chances are that there is no business requirement to actually use 4000. It was simply convenient for the developer to max-out the type.

   The preferred solution in this case is to actually derive the true requirement.

   One way to do that is look at the data. E.g. running; SELECT MAX(LENGTH(c1)) FROM T will quickly give a fairly good idea on what is really being used.

   So if reducing the length of the VARCHAR2 columns is an option it is by far the best solution.

2. **The table has some big string fields. That is there may be VARCHAR2(4000) but they are actually required.**

   You can convert these columns into CLOB(4000) instead and perhaps provide a suitable INLINE LENGTH to ensure the common cases are actually stored within the row.

   If there are predicates other than LIKE such as IN, or equal defined on these columns you need to cast the columns back to varchar. Here is an example:

   ```
   CREATE TABLE T(pk NUMBER(10) NOT NULL PRIMARY KEY, c1 VARCHAR2(20000), c2
   VARCHAR2(20000));
   ```

   turns into:

   ```
   CREATE TABLE T(pk NUMBER(10) NOT NULL PRIMARY KEY, c1 CLOB(20000) INLINE LENGTH 15000,
   c2 CLOB(20000) INLINE LENGTH 15000);

   SELECT pk FROM T WHERE c1 LIKE 'hello%';

   SELECT c2 FROM T WHERE TO_CHAR(c1) = 'world';
   ```

Can be converted automatically: NO

Rows with too many columns

Description

In DB2, using the default of 4-KB pages for table spaces, tables are restricted to 500 columns. Larger page sizes (8 KB, 16 KB, and 32 KB) support 1012 column.

So when you have an Oracle table that exceeds this limit things get tricky.

Solution

It is recommended to split the table into two.

E.g.: A table `T(pk1, pk2, c1, c2, ... c1000)` can be split into

`T1(pk1, pk2, c1, ...c500)` and `T2(pk1, pk2, c501, ...c1000)`

and a view can be used to concatenated the result set from both tables:

```
CREATE VIEW T AS SELECT T1.pk1, T1.pk2, c1, ... c1000 FROM T1, T2 WHERE T1.pk1 = T2.pk1 AND T1.pk2 = T2.pk2
```

Note that the view T is READ ONLY. If T must be updatable and the application cannot easily be changed then it may be possible to use INSTEAD OF trigger (written in SQL PL!) to model the UPDATE/DELETE/INSERT.

Can be converted automatically: NO

UROWID data type is not supported

Description
At this time, UROWID is not a supported data type.

Solution
Create additional column in each table that requires this data type. The value for each row should be generated by getting a new value from a sequence that will be shared by all tables.

Can be converted automatically: NO

TIMESTAMP WITH TIMEZONE data type and SYSTIMESTAMP function

Description
At the moment, TIMESTAMP WITH TIMEZONE data type is not supported by DB2.

SYSTIMESTAMP is a function with return type TIMESTAMP WITH TIMEZONE and it is also not supported by DB2.

```
SELECT SYSTIMESTAMP FROM dual
```

Solution
Possible alternatives:

1. **Use of an additional column.**

   Create an additional column that will contain the timezone component, and include the value of this column in all time calculations.
2. **Keep database server on UTC time.**

   This way all time values in the database would be in UTC.
3. If timezone component is not really necessary, SYSTIMESTAMP can be replaced with CURRENT TIMESTAMP.

Can be converted automatically: YES. with Database Conversion Workbench (DCW).