ORACLE 11G VIRTUAL COLUMN

Inderpal S. Johal, Data Softech Inc.

INTRODUCTION

In Oracle 11g has allowed database Tables to have virtual columns. These virtual columns can be more specifically called as derived column as these columns derived their data from other non-virtual columns of the same table. They are like any other table column but their data is not stored in the database and so they consume no disk space. They got their value from an expression which can include

- Columns from the same Table
- Some constants
- Any SQL functions
- A User-defined PL/SQL functions

You cannot explicitly write the data to the virtual column. Virtual columns can be used in queries, DML and DDL statement. You can index as well as collect statistics on them.

This feature will help to lots of application which are already in production but need enhancement with little development efforts. Some of these will be covered in examples later on in this paper

SYNTAX FOR TABLE VIRTUAL COLUMN

```
Create table Test11g
(
      Col1  datatype,          → Standard Column
      Vcol1 datatype GENERATED ALWAYS AS
      ( expression)
      VIRTUAL (Constraint)    → Virtual Column
)

Where
      Vcol1                → Name of the Virtual Column
      Datatype             → All Oracle Datatype except LOB or LONG RAW
                             If not specified, then Column expression will
                             determines its datatype.
      GENERATED ALWAYS     → Indicates that the column is generated on demand
                             and not stored on disk
      AS (col_expression)  → Col Expression can be    - Simple Expression
                                                      - Complex Expression
                                                      - Function Expression
                                                      - Expression List
                             Expression can only use
                                   - Columns of the same table
                                   - Constants (strings or Numbers)
                                   - built in SQL functions except Aggregate
                                   - user-defined functions
      VIRTUAL              → Showing that column is Virtual. It is optional
      Constraint           → Regular Constraint Clause
```

VIRTUAL COLUMN EXAMPLES

Prior to Oracle 11g, we can only create partition on existing physical column of the table. With 11g, we can use expression as partitioning key utilizing one or more existing columns of a table. This kind of virtual column based partitioning support all types of partitions and add lots of flexibility.

Note that virtual column is a column whose value is the result of an expression and it is not stored on disk and only exist as metadata.

```
Create a table with a Virtual Column GRADE whose datatype is detemined by Case statement
SQL> create table Student_Marks
  2  (
  3  std_Id number,
  4  std_Name varchar2(20),
  5  std_Mark number,
  6  Grade
  7  generated always as
  8  (
  9  case
 10  when std_Mark > 80 then 'Excellent'
 11  when std_Mark > 60 then 'Very Good'
 12  when std_Mark > 50 then 'Good'
 13  when std_Mark > 40 then 'Satisfactory'
 14  else 'Failed'
 15  end
 16  ) virtual
 17  )
 18  /
Table created.

SQL> desc student_marks
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 STD_ID                                             NUMBER
 STD_NAME                                           VARCHAR2(20)
 STD_MARK                                           NUMBER
 GRADE                                              VARCHAR2(9)

Add Data to the Table
SQL> insert into student_marks(std_Id,std_Name,std_mark) values(&id,'&name',&m1)
  2  /
Or
SQL> insert into student_marks values(&id,'&name',&m1,default)
  2  /
Enter value for id: 1
Enter value for name: Inderpal
Enter value for m1: 90
1 row created.
```

```
SQL> /
Enter value for id: 2
Enter value for name: Inder
Enter value for m1: 70
1 row created.

SQL> /
Enter value for id: 3
Enter value for name: Indy
Enter value for m1: 51
1 row created.

SQL> /
Enter value for id: 4
Enter value for name: IPS
Enter value for m1: 41
1 row created.

SQL> /
Enter value for id: 5
Enter value for name: Noway
Enter value for m1: 30
1 row created.

SQL> commit;
Commit complete.

SQL> select * from student_marks;
    STD_ID STD_NAME               STD_MARK GRADE
---------- -------------------- ---------- ---------
         1 Inderpal                     90 Excellent
         2 Inder                        70 Very Good
         3 Indy                         51 Good
         4 IPS                          41 Satisfactory
         5 Noway                        30 Failed
```

You can create an Index on Virtual column

```
SQL> create index text_vidx on student_marks(grade);
Index created.

SQL> select index_type from user_indexes where index_name = 'TEXT_VIDX';
INDEX_TYPE
---------------------------
FUNCTION-BASED NORMAL

SQL> select column_expression from user_ind_expressions where index_name = 'TEXT_VIDX';
COLUMN_EXPRESSION
--------------------------------------------------------------------------------
CASE  WHEN "STD_MARK">80 THEN 'Excellent' WHEN "STD_MARK">60 THEN 'Very Good' WH
EN "STD_MARK">50 THEN 'Good' WHEN "STD_MARK">40 THEN 'Poor' ELSE 'Failed' END
```

What will happen when we create new table from existing virtual Column table. The new Table will be regular table with no virtual column and can be updated.

```
SQL> create table std as select * from student_marks;
Table created.

SQL> insert into std values (11,'TEST',99);
insert into std values (11,'TEST',99)
            *
ERROR at line 1:
ORA-00947: not enough values
```

Insert is Allowed which is not possible for Virtual column as shown below with STUDENT_MARKS table

```
SQL> insert into std values (11,'TEST',99,'POOR');
1 row created.

SQL> select * from std;
    STD_ID STD_NAME              STD_MARK GRADE
---------- -------------------- ---------- ---------
         1 Inderpal                     90 Excellent
         2 Inder                        70 Very Good
         3 Indy                         50 Poor
         7 test51                       51 Good
         8 test41                       41 Poor
        11 TEST                         99 POOR

6 rows selected.
```

```
Direct Insert not allowed on virtual column as it is always derived
SQL> insert into student_marks values (11,'TEST',99,'POOR');
insert into student_marks values (11,'TEST',99,'POOR')
            *
ERROR at line 1:
ORA-54013: INSERT operation disallowed on virtual columns

Two ways to Insert Data into Virtual Column Table
SQL> insert into student_marks values(11,'test',99,default);
1 row created.

SQL> insert into student_marks(std_id,std_name,std_mark) values (12,'test1',99);
1 row created.
```

WHAT YOU CAN DO WITH VIRTUAL COLUMN

1. Create an index which is FUNCTION-BASED index – As shown above
2. Virtual column can be used in WHERE clause of DELETE statement

```
SQL> select * from student_marks;
    STD_ID STD_NAME              STD_MARK GRADE
---------- -------------------- ---------- ---------
         1 Inderpal                     90 Excellent
         2 Inder                        70 Very Good
         3 Indy                         50 Poor
         4 IPS                          40 Failed
         5 Noway                        30 Failed
         7 test51                       51 Good
         8 test41                       41 Poor
         9 test30                       30 Failed
8 rows selected.

SQL> delete from student_marks where grade='Failed';
3 rows deleted.
```

3. Virtual column are available for Oracle 11g new feature RESULT CACHING. I will discuss it in detail as separate topic very soon.
4. Virtual column can be used as partitioning key column – Will be covered in detail in Partitioning paper

WHAT YOU CAN'T DO WITH VIRTUAL COLUMN

1. External tables can't have Virtual column – External Tables are Read only and so no DML and Index are allowed on them and hence it can't have virtual column
2. Index-organized Table can't have virtual column
3. Virtual Columns are also not supported for Object, cluster and Temporary tables
4. Cannot directly update virtual column using `UPDATE .. SET` clause. It can be used in `WHERE` clause of the `UPDATE` command.

```
SQL> update student_marks set grade='PASS';
update student_marks set grade='PASS'
       *
ERROR at line 1:
ORA-54017: UPDATE operation disallowed on virtual columns


SQL> update student_marks set std_mark=std_mark+5 where grade='Poor';
2 rows updated.
```

5. Virtual column expression cannot refer to another virtual column by its name

```
Add new virtual column to the existing table
SQL> alter table student_marks add ( total_marks as (std_mark +10));
Table altered.

SQL> desc student_marks
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 STD_ID                                             NUMBER
 STD_NAME                                           VARCHAR2(20)
 STD_MARK                                           NUMBER
 GRADE                                              VARCHAR2(9)
 TOTAL_MARKS                                        NUMBER

You cannot use Virtual Column to create new Virtual columns
SQL> alter table student_marks add (Grace_marks as (total_marks +1));
alter table student_marks add (Grace_marks as (total_marks +1))
*
ERROR at line 1:
ORA-54012: virtual column is referenced in a column expression
```

6. Virtual column can only use regular column of the table itself and not from outside table.
7. Virtual columns output must be a scalar datatype value.