Misceláneo de Nuevas Funcionalidades de Oracle Database 12c (parte 2)

Por Francisco Riccio

Introducción

A continuación se detallará una lista de nuevas funcionalidades disponibles en Oracle Database 12c, que están orientadas a optimizar nuestras implementaciones y actividades de desarrollo y administración.

Implementación

1. Campos cuyo valor por default referencian a secuencias

En las versiones anteriores si deseábamos que un campo tenga valores que provengan de una secuencia; se realizaba mediante triggers que se dispararían antes de la operación INSERT y obtenían el NEXTVAL de la secuencia. En esta nueva versión, Oracle Database 12c nos permite de manera sencilla indicar que el valor por default de un campo será el NEXTVAL de una secuencia sin la necesidad de crear ningún trigger.

En el ejemplo se crea una secuencia con su configuración por default (inicia en 1 e incrementa en 1 en 1).

```
SQL> create sequence seq_contador;
Sequence created.
```

Creamos una tabla llamada PRODUCTO cuyo campo CONTADOR estará asociado al NEXTVAL de la secuencia SEQ_CONTADOR en caso no se le especifique en la operación de INSERT un valor.

```
SQL> create table producto (cod number, contador number default seq_contador.nextval);
Table created.
```

Ingresamos un registro en la tabla PRODUCTO sin especificar un valor en el campo CONTADOR y vemos que por default ha sido poblado con el valor NEXTVAL de la secuencia.

```
SQL> insert into producto(cod) values (1);
1 row created.
SQL> commit;
Commit complete.
SQL> select * from producto;
       COD   CONTADOR
---------- ----------
         1          1
```

2. Columnas de tipo VARCHAR/NVARCHAR2/RAW con soporte a 32 KB

Oracle Database 12c permite crear columnas de tipo VARCHAR/NVARCHAR2/RAW con soporte a 32 KB. Las versiones anteriores se tenía un límite de 4000 bytes para los tipos de datos VARCHAR/NVARCHAR2 y 2000 bytes para el tipo de dato RAW.

Para habilitar esta opción que nos ofrece Oracle en su nueva versión debemos modificar el parámetro MAX_STRING_SIZE del valor STANDARD a EXTENDED.

En el siguiente ejemplo se crea la tabla ARTICULO con un campo llamado DESCRIPCION que puede albergar 32KB de información.

```
SQL> alter system set max_string_size='EXTENDED' scope=spfile;

System altered.

SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> startup
ORACLE instance started.

Total System Global Area 1068937216 bytes
Fixed Size                  2390048 bytes
Variable Size             750782432 bytes
Database Buffers          310378496 bytes
Redo Buffers                5386240 bytes
Database mounted.
Database opened.
SQL> create table ARTICULO (descripcion varchar(32000));

Table created.
```

Recordemos que el tipo de dato VARCHAR es más eficiente que los tipos de datos CLOB y XMLType.

3. Optimización al agregar una nueva columna en una tabla

Cuando agregamos una columna en una tabla existente y está columna la definimos con un valor por default; Oracle Database 12c hará que esta columna no ocupe espacio, esto simplemente se almacenará como una entrada en el diccionario de datos.

Validación:

Hemos creado una tabla llamada DEMO en una base de datos Oracle Database 11g y 12c ambas con la misma estructura y la misma cantidad de filas.

Oracle Database 11g:

```
SQL> select bytes from dba_segments where segment_name='DEMO' and owner='SYS';

     BYTES
----------
    262144

SQL> alter table DEMO add campo2 char(2000) default 'VALOR';

Table altered.

SQL> execute dbms_stats.gather_table_stats('SYS','DEMO');

PL/SQL procedure successfully completed.

SQL>
SQL> select bytes from dba_segments where segment_name='DEMO' and owner='SYS';

     BYTES
----------
  55574528
```

Podemos apreciar que la tabla DEMO ha incrementado su tamaño de 262144 bytes a 55574528 bytes al agregar una nueva columna.

La misma prueba realizada en una base de datos Oracle Database 12c entregó los siguientes resultados:

```
SQL> select bytes from dba_segments where segment_name='DEMO' and owner='SYS';

     BYTES
----------
    262144

SQL> alter table DEMO add campo2 char(2000) default 'VALOR';

Table altered.

SQL> execute dbms_stats.gather_table_stats('SYS','DEMO');

PL/SQL procedure successfully completed.

SQL>
SQL> select bytes from dba_segments where segment_name='DEMO' and owner='SYS';

     BYTES
----------
    262144
```

Podemos ver que la tabla mantuvo su mismo tamaño de 262144 bytes.

El campo recién ocupará espacio en la tabla cuando se ejecuten operaciones UPDATE sobre el campo.

4. Movimiento online de particiones y subparticiones de una tabla

Oracle Database 12c nos permite mover una partición y subpartición de una tabla particionada a otro tablespace o sobre el mismo de manera online permitiendo que las aplicaciones puedan seguir escribiendo sobre la tabla sin generar algún corte en el servicio.

La sintaxis es la siguiente:

ALTER TABLE <NOMBRE_TABLA> MOVE PARTITION|SUBPARTITION <NOMBRE_PARTICION_SUBPARTICION> *UPDATE INDEXES* ONLINE *TABLESPACE* *<NOMBRE_TABLESPACE>*

Ejemplo:

Creamos la tabla particionada VENTAS y moveremos su partición P2 al tablespace DATA de manera online.

```
SQL> create table VENTAS(
  2   id number(5),
  3   cliente varchar2(30) not null,
  4   total_venta number(8) not null,
  5   no_semana number(2) not null
  6  )
  7  partition by range(no_semana)
  8  (
  9   partition p1 values less than(12) tablespace USERS,
 10   partition p2 values less than(24) tablespace USERS,
 11   partition p3 values less than(36) tablespace USERS
 12  );

Table created.

SQL> alter table VENTAS move partition P2 online tablespace DATA;

Table altered.
```

Nota: Si la tabla particionada tiene índices locales y globales es recomendable utilizar la cláusula UPDATE INDEXES como se muestra a continuación:

```
SQL> alter table VENTAS move partition P2 update indexes online tablespace DATA;

Table altered.
```

UPDATE INDEXES permitirá que los índices asociados sobre la tabla no queden en estado UNUSABLE.

5. Personalización en la política de compresión de filas

Oracle Database 12c mantiene los mismos métodos de compresión que se venía trabajando en la versión 11gR2, los cuales se detallan:

Método de Compresión	Sintaxis
BASIC	COMPRESS BASIC
OLTP	COMPRESS FOR OLTP
WAREHOUSE	COMPRESS FOR QUERY LOW|HIGH
ARCHIVE	COMPRESS FOR ARCHIVE LOW|HIGH

Ahora es posible indicar basado en una regla que ciertas filas tendrán un método de compresión.

La sintaxis es:

ALTER TABLE <NOMBRE_TABLA> ILM ENABLE ACTIVITY TRACKING (WRITE TIME); (Habilitamos el Activity Tacking)

ALTER TABLE <NOMBRE_TABLA> ILM ADD <NOMBRE_REGLA> <METODO_COMPRESION> <POLITICA>; (Creamos la regla)

Ejemplo:

Crearemos una regla llamada REGLA_OLTP el cual comprimirá con el método OLTP para aquellas filas de la tabla ventas que no tengan ninguna modificación en 30 días.

```
SQL> create table ventas(
  2   id number(5),
  3   cliente varchar2(30) not null,
  4   total_venta number(8) not null,
  5   no_semana number(2) not null
  6  )
  7  partition by range(no_semana)
  8  (
  9   partition p1 values less than(4) tablespace users,
 10   partition p2 values less than(8) tablespace users,
 11   partition p3 values less than(12) tablespace users
 12  );

Table created.

SQL> alter table VENTAS ilm enable activity tracking (write time);

Table altered.

SQL> alter table VENTAS ilm add REGLA_OLTP
  2  compress for oltp
  3  row after 30 days of no modification;

Table altered.
```

Asimismo también es posible indicar que si una partición no ha tenido modificaciones se le configure un método de compresión.

La sintaxis es:

ALTER TABLE <NOMBRE_TABLA> ILM ENABLE ACTIVITY TRACKING SEGMENT ACCESS; (Habilitamos el Activity Tacking)

ALTER TABLE <NOMBRE_TABLA> ILM ADD <NOMBRE_REGLA> <METODO_COMPRESION> <POLITICA>; (Creamos la regla)

Ejemplo:

```
SQL> alter table ventas ilm enable activity tracking segment access;

Table altered.

SQL> alter table ventas ilm add REGLA_OLAP
  2  compress for query high
  3  segment after 90 days of no modification;

Table altered.
```

Conclusiones

Durante el transcurso de este material se ha mostrado algunas de las nuevas opciones que Oracle Database 12c tiene incorporado ayudándonos en muchas actividades que antes se realizaban con mayor esfuerzo y ahora vienen listas para usarse de manera sencilla.

Publicado por Ing. Francisco Riccio. Es un IT Oracle Specialist e instructor de cursos oficiales de certificación Oracle. Está reconocido por Oracle como un Oracle ACE y certificado en productos de Oracle Application / Oracle Database / Oracle Developer.

e-mail: francisco@friccio.com

web: www.friccio.com