TEMA 07 parte 2
Ejercicio práctico 10
Administración de datos Undo

NOMBRE: **GRUPO:**

FECHA DE ENTREGA: **CALIFICACION:**

1.1. OBJETIVO

Comprender e ilustrar los efectos que produce un tablespace undo sin espacio durante la operación de la base de datos.

1.2. ADMINISTRACIÓN BÁSICA DE DATOS UNDO.

A. Generar una sentencia SQL que muestre el tablespace undo que actualmente está en uso.
B. Generar una sentencia SQL que genere un nuevo tablespace undo cuya característica principal es contar con un espacio muy limitado y por lo tanto la base de datos estará en riesgo de generar errores al requerir más espacio undo del disponible. Nombrar al tablespace `undotbs2`, asignarle un solo data file ubicado en `'/u01/app/oracle/oradata/<ORACLE_SID>/undotbs_2.dbf'` con una capacidad únicamente de 30 MB. El data file no deberá auto extenderse. La administración de sus extensiones debe ser local.
C. Generar la sentencia SQL necesaria para que la instancia ahora haga uso del nuevo tablespace en lugar el actual. Aplicar el cambio únicamente mientras la instancia esté iniciada.
D. Crear una nueva sesión con el usuario SYS, ejecutar nuevamente la sentencia del inciso A para verificar que la instancia ahora está haciendo uso del tablespace `undotbs2`.
E. Considerar la vista `v$undostat`. Generar la siguiente consulta. Mostrar la fecha inicio y fin del periodo de la muestra hasta nivel de segundos, el id del tablespace que se está empleando, total de bloques undo empleados, total de transacciones, id de la consulta que tardó el mayor tiempo en ejecutarse, tiempo en segundos de la consulta que más tiempo tardó en ejecutarse durante el periodo de la muestra, número de bloques undo activos, no expirados y expirados; el valor del parámetro `undo_retention` calculado por la instancia. Ordenar los registros con base a la fecha de inicio de forma descendente. Mostrar únicamente los primeros 20 registros.
F. Considerar el periodo de muestreo más reciente de la consulta realizada en el inciso E. ¿Cuantos bloques podrían ser sobrescritos sin causar mayores inconvenientes?, ¿Cuántos bloques NO pueden ser sobrescritos aun?

G. Observar los valores de la columna `undotsn` de la consulta del inciso E. Confirmar que a partir del cambio del tablespace undo, el valor de esta columna cambia. Generar una sentencia SQL que permita mostrar los nombres de los tablespaces asociados a los valores de esta columna. Tip: auxiliarse de `v$tablespace`. En la consulta incluir únicamente el periodo de la muestra, id del tablespace y su nombre.
H. Generar una consulta que muestre las siguientes columnas: Nombre del tablespace undo que fue creado en pasos anteriores, total de bloques que contiene, total de bloques libres, porcentaje de bloques libres. Confirmar que el porcentaje de bloques libres es bajo, menor al 20%. Esta condición pone en riesgo la operación correcta que requieren hacer uso de los datos Undo.
I. Empleando el usuario creado en temas anteriores, generar una tabla con la siguiente estructura:

```
create table <iniciales>_cadena_2 (
  id number constraint jrc_cadena_2_pk primary key,
  cadena varchar2(1024)
) nologging;
```

- Observar la cláusula `nologging` que se agrega al final. La cláusula evita la generación de datos de Redo.
- En esta tabla se estarán realizando diversas operaciones DML para verificar el funcionamiento de los datos undo. Dichas instrucciones provocarán buena cantidad de datos Redo, y por lo tanto su archivado. Para este ejercicio no es necesario conservarlos. Por esta razón, se usa la cláusula `nologging`.
- Generar una secuencia `sec_<iniciales>_cadena_2`.
- Crear un programa PL/SQL que realice la inserción de 50,000 registros. Auxiliarse de la función `dbms_random.string`. Hacer uso de la secuencia para generar los valores de la PK. Ejecutar el programa y hacer `commit`;

J. Replicación del error al ejecutar sentencias DML.

Ejecutar nuevamente la consulta del inciso E. Esperar unos minutos a que el número de bloques undo utilizados sea cercano a cero. Posteriormente, Iniciar con el borrado de registros de los datos insertados. Debido a que se tiene poco espacio Undo, el resultado esperado es un error ya que las sentencias `delete` requieren guardar datos Undo y los bloques disponibles no serán suficientes.

Realizar el proceso de borrado por rangos de 5,000 registros. Posteriormente realizar nuevamente la consulta del inciso E y considerar únicamente al registro más reciente, agregar un renglón a la siguiente tabla. No hacer `commit` ni `rollback` en cada intento. Detener la actividad hasta ocurrir alguno de los 2 eventos: error (incluir el detalle), o al terminar de realizar el borrado. Al terminar sin importar el resultado, realizar `rollback`. Revisar los datos de la tabla y generar una conclusión al respecto.

Registros eliminados	# bloques utilizados	# transacciones ejecutadas	#bloques activos	Retención en segundos calculado.
1-5000				
…				

K. ¿Qué acciones se pueden realizar para corregir el error adicional a aumentar el tamaño del tablespace undo ?

L. Replicación del error ***snapshoot too old.***

En la siguiente actividad se provocará el segundo error: ***Snapshoot too old***. El escenario es el siguiente: Suponer la creación de una transacción T1-RC que tiene un nivel de aislamiento de *lecturas repetibles*. Esto significa que si una sentencia SQL consulta datos a la tabla creada anteriormente, la base de datos deberá mostrar siempre la misma información a pesar de otras transacciones hagan `commit`. Este nivel de aislamiento obliga a que la base de datos haga uso de los datos undo para poder brindar el nivel de aislamiento solicitado (read consistency). Sin embargo, ¿qué sucedería si otras transacciones comienzan a realizar operaciones DML? Debido a que el espacio undo es limitado, la base de datos comenzará a reescribir los segmentos undo para tratar de atender exitosamente a las transacciones DML. La sobrescritura de estos segmentos undo provocará que los datos consultados por T1-RC ya no estén disponibles en el segmento undo, y por lo tanto, la base de datos ya no podrá mostrar los datos con el nivel de aislamiento solicitado, provocando así un error *snapshoot too old.* La forma de entender este error es:

La consulta inició hace cierto tiempo. Al ser una transacción de larga duración, los datos undo que requiere para garantizar consistencia ya fueron reciclados, por lo tanto la "versión (snapshot)" que requiere esta transacción es demasiada vieja y ¡ya no se encontró en los segmentos undo! Las acciones para generar este error son las siguientes:

1. Abrir una sesión en sqlplus con el usuario dueño de la tabla creada anteriormente. Iniciar una transacción con nivel de aislamiento que permita lecturas repetibles:

```
set transaction isolation level serializable name 'T1-RC';
```

2. Ejecutar las siguientes consultas las cuales sirven como referencia. Incluir resultados.

```
select count(*) from <iniciales>_cadena_2;
select count(*)
from <iniciales>_cadena_2
where cadena  like 'A%'
or cadena like 'Z%'
or cadena like 'M%';
```

Mientras esta transacción no haga `commit` o `rollback`, el nivel de aislamiento deberá mostrar exactamente los mismos resultados aunque otras transacciones hagan cambios sobre los registros de la tabla.

3. Abrir otra sesión empleando el mismo usuario. Este usuario intentará borrar los registros de la tabla. La instancia tratará atender exitosamente a esta transacción pero provocará error en la sesión 1. Para ello, repetir las actividades del inciso J. Ejecutar operaciones `delete` en rangos de 5000 registros. Ya no será necesario llenar la tabla. Al terminar de ejecutar la sentencia `delete` hacer `commit` y ejecutar nuevamente las consultas de T1-RC. Observar que se muestran exactamente los mismos resultados a pesar de que otras transacciones han borrado datos y han hecho `commit`. Esto se debe gracias a que la transacción fue configurada con el nivel de aislamiento `serializable` y permite lecturas repetibles. La BD está tomando los datos de los segmentos undo que aún no son sobrescritos. Repetir estas acciones hasta producir el error *snapshoot too old*. Incluir las sentencias y la salida de su ejecución. ¿Cuántas instrucciones `delete` se tuvieron que ejecutar para provocar el error *snapshppt too old*?

M. Provocar ahora el comportamiento inverso: Hacer que la base de datos le de preferencia a consultas que requieran hacer uso de datos undo en lugar de dar preferencia a las sentencias SQL:

1. Incluir la sentencia SQL que permite modificar este comportamiento, ejecutarla.

2. Posterior a su ejecución, nuevamente crear una transacción T1-RC y ejecutar las mismas consultas. De forma similar, en otra sesión intentar eliminar registros de la tabla. Incluir las sentencias SQL empleadas, incluir los errores generados, explicar el comportamiento obtenido. ¿Qué sucede al ejecutar repetidamente las consultas de T1-RC?

1.3. VALIDADOR.

- Sin validador para este ejercicio.

```
sqlplus /nolog
start s-05-validador-main.sql
```

1.4. CONTENIDO DE LA ENTREGA.

- Incluir las sentencias SQL y la salida obtenida en cada uno de los puntos de este ejercicio.
- Elementos generales indicados en la rúbrica general de ejercicios prácticos (datos generales, conclusiones y comentarios).
- Entrega individual