TEMA 04
Ejercicio práctico 02
Administración de las estructuras de Memoria

NOMBRE: **GRUPO:**

FECHA DE ENTREGA: **CALIFICACION:**

1.1. OBJETIVO

Comprender y revisar las principales vistas del diccionario de datos que muestran información acerca de la cantidad de memoria que se le asigna al db buffer cache, en particular al default pool. A partir de una serie de actividades, comprender y analizar la forma en la que se leen buffers en del default pool: lecturas físicas (de los data files), lecturas del caché y lecturas consistentes. Comprender y configurar JMeter como herramienta principal para simular operaciones simultáneas realizadas por diversos usuarios.

1.2. DB BUFFER CACHE.

1.2.1. Principales vistas asociadas con el DB Buffer cache.

- Crear un usuario `<nombre>0402` en caso de no existir. Otorgarle los privilegios necesarios para que este pueda crear tablas, sesiones, secuencias y cuota ilimitada en `users`.

Generar un script `s-01-crea-objetos.sql` que contenga la definición de los siguientes objetos:

- Crear una tabla `t03_db_buffer_cache`. La tabla deberá contener la siguiente información acerca del default pool del DB Buffer cache. Tip: revisar `v$buffer_pool`.

`block_size`: tamaño del bloque de datos por default configurado durante la creación de la BD.
`current_size`: Tamaño actual del default pool
`buffers`: Total de buffers en el pool.
`target_buffers`: Número nuevo de buffers que tendrá el pool después de una operación de crecimiento o reducción.
`prev_size`: Tamaño anterior en MB que tenía el pool antes de la última operación.
`prev_buffers`: Número de buffers anterior que tenía el pool antes de la última operación.
`default_pool_size`: Esta columna no pertenece a la vista `v$buffer_pool`. En esta columna se deberá almacenar el valor del parámetro de la BD que guarda la cantidad de memoria mínima que se le puede asignar al default pool.

C1. Incluir en el reporte el código DDL y los datos de esta consulta. Contestar las siguientes preguntas:

A. ¿Qué implica que los valores `current_size, target_size` y los valores para buffers, `target_buffers` tengan el mismo valor?
B. ¿Qué implica que los valores de los atributos `prev_size y pref_buffers` sea cero?
C. ¿Qué implica que el valor de la columna `default_pool_size` sea cero?

- Crear una tabla `t04_db_buffer_sysstats` Esta tabla contendrá 3 principales estadísticas que se recolectan del default pool. La tabla deberá contener la siguiente estructura:

DB_BLOCKS_GETS_FROM_CACHE	CONSISTENT_GETS_FROM_CACHE	PHYSICAL_READS_CACHE
2360194	7163294	231255

- Las 3 columnas que se muestran a continuación corresponden a las 3 estadísticas que se obtienen de la vista `v$sysstat`. Cada estadística es un renglón de dicha vista:

- Número de buffers que son obtenidos del default pool para satisfacer una sentencia SQL solicitada por un usuario.

```
select value from v$sysstat where name ='db block gets from cache' as  db_blocks_gets_from_cache;
```

- Número de buffers que son obtenidos del default pool en modo consistente para satisfacer una sentencia SQL solicitada por un usuario.

```
select value from v$sysstat where name ='consistent gets from cache' as consistent_gets_from_cache;
```

- Número de bloques de datos leídos de data files y almacenados en el default pool para satisfacer una sentencia SQL solicitada por un usuario.

```
select value from v$sysstat where name ='physical reads cache' as physical_reads_cache;
```

- Existe una métrica llamada `cache_hit_radio`. Este valor permite visualizar la relación entre lecturas realizadas al default pool VS lecturas físicas. Dicha relación permite visualizar si el db buffer caché cuenta con la memoria suficiente para minimizar las lecturas físicas. Su fórmula se calcula de la siguiente forma:

$$radio = 1 - \frac{physical\_reads\_cache}{db\_blocks\_gets\_from\_cache + consistent\_gets\_from\_cache}$$

- Agregar una cuarta columna a la tabla anterior llamada `cache_hit_radio` que contenga el cálculo de la fórmula anterior. Truncar a 6 decimales.
- **C2. Incluir en el reporte** el DDL de esta tabla y el resultado de ejecutar `select * from t04_db_buffer_sysstats` . Contestar la siguientes preguntas:

A. ¿Qué valores debería tener `cache_hit_radio` para considerar que el db buffer cache está correctamente configurado, es decir, las lecturas físicas se minimizan?
B. ¿Qué valores debería tener `cache_hit_radio` para considerar que el db buffer cache requiere un aumento de memoria debido a un incremento de lecturas físicas.

- Tip: Estas preguntas podrían responderse fácilmente al realizar el siguiente ejercicio.

1.2.2. Carga y consulta de datos.

En este ejercicio se realizará la carga de 80,000 registros en una tabla que contiene datos aleatorios de diversos tipos de datos. Posterior a la carga y al reinicio de la instancia, se simulará la creación de 50 usuarios que lanzarán consultas hacia esta misma tabla de forma simultánea durante 4 minutos. Durante este periodo de tiempo se realizará una recolección manual y simple de las 3 métricas y del `cache_hit_radio` para observar la forma en la que se comporta el db buffer cache y comprobar los conceptos vistos en teoría.

Para implementar este plan de pruebas se hará uso de JMeter. Esta es una herramienta escrita en Java que permite realizar diversos tipos de pruebas, en particular, ofrece la configuración de diversos hilos de ejecución para simular pruebas de volumen, concurrencia, etc. Este ejercicio se divide en las siguientes etapas:

- Carga de datos
- Consulta de datos y monitoreo del DB Buffer Cache.

1.2.3. Etapa 1 Carga de datos.

- Instalar Java (JDK), configurar las variables `JAVA_HOME` y `PATH`. Para mayores detalles ver el siguiente video.
- Descargar e iniciar JMeter. Para una mejor comprensión del proceso de instalación se proporciona el siguiente video.
- Carga de datos. La inserción de los 80,000 registros se deberá realizar en una tabla con la siguiente estructura.
- Generar un script `s-02-monitor-default-pool.sql` que realice las siguientes acciones:

- Crear la siguiente tabla para almacenar los datos aleatorios.

```
create table t01_random_data(
  id number(18,0),
  r_varchar varchar2(1024),
  r_char char(18),
  r_integer number(10,0),
  r_double number(20,10),
  r_date date,
  r_timestamp timestamp
);
```

- Crear una secuencia llamada `seq_t01_random_data`

La tabla anterior contiene atributos de diversos tipos de datos cuya finalidad es almacenar datos aleatorios.

A. Antes de comenzar con la carga de datos, asegurarse de haber ejecutado el script `s-02-monitor-default-pool.sql`. Revisar este video para comenzar con el proceso de carga de datos.
B. Al terminar la prueba, comprobar la existencia de los 80,000 registros en la BD.

1.2.4. Consultas y recopilación de resultados.

1.2.4.1. Recopilación de resultados.

En este ejercicio se realizará una simulación en la que 50 usuarios concurrentes consulten datos aleatorios de la tabla `t01_random_data`. A nivel general la idea de este ejercicio es la siguiente:

1. Reiniciar la instancia para que el db buffer cache no contenga datos previamente cargados.
2. Realizar consultas cada segundo durante 4 minutos hacia la vista `v$sysstat` para obtener los valores de las 4 columnas que contiene la tabla `t04_db_buffer_sysstats` Importante: Las consultas se deben realizar sobre `v$sysstat`, no sobre `t04_db_buffer_sysstats` Para el caso de la columna `cache_hit_radio`, truncar a 6 decimales.
3. El resultado de cada consulta deberá ser almacenado en la siguiente tabla. Incluir su definición en el archivo `s-02-monitor-default-pool.sql`

```
create table t02_db_cache_stats(
  sample date date,
  db_blocks_gets_from_cache number,
  consistent_gets_from_cache number,
  physical reads cache number,
  cache_hit_radio number
);
```

En general este muestreo de datos permitirá monitorear al DB buffer caché durante 4 minutos. Revisar el siguiente video para configurar esta tarea en JMeter. La columna `sample_date` contiene la fecha en la que se realiza la consulta**. C3. Incluir en el reporte** la pantalla de JMeter donde se muestra la configuración de la sentencia SQL que realiza la inserción de datos en `t02_db_cache_stats`.

1.2.4.2. Generación de consultas.

Durante los 4 minutos en donde se realiza el monitoreo del db buffer cache se realizarán consultas aleatorias simulando a 50 usuarios concurrentes. La generación de estas consultas se realizará con base a la siguiente estrategia:

1. Durante el primer minuto de la prueba no se realizarán consultas. La finalidad es monitorear el estado de db buffer cache durante un minuto previo al inicio de las consultas y así poder apreciar los cambios que sufre a partir de segundo minuto.
2. A partir del minuto 2, los 50 usuarios comenzarán a realizar consultas aleatorias. La sentencia SQL es la siguiente:

```
select sum(length(r_varchar)) from t01_random_data where id = sys.dbms_random.value(?,?);
```

- Observar que la sentencia anterior contiene 2 placeholders (sentencia preparada). Sus valores serán 1 y 80000. Es decir, cada consulta obtendrá un registro al azar y calculará la longitud del atributo `r_varchar`.

Revisar los siguientes videos para configurar esta tarea en JMeter.

- Video parte 4
- Video parte 5

3. Ejecutar la prueba.
4. Al concluir revisar el contenido de la tabla `t02_db_cache_stats`. Se obtendrán alrededor de 100 registros.
5. Para realizar el análisis del comportamiento del db buffer caché, generar las siguientes gráficas.

- `Fecha vs db_blocks_gets_from_cache`
- `Fecha vs consistent_gets_from_cache`
- `Fecha vs physical_reads_cache`
- Fecha vs las 3 columnas anteriores (en la misma gráfica)

C4. Incluir en el reporte las 4 gráficas y una explicación o análisis de los resultados obtenidos.

1.3. SIMULACIÓN DE LA MEMORIA FLASH PARA EL DB BUFFER CACHE.

En este ejercicio se realizarán las configuraciones necesarias para habilitar el uso de una memoria flash asociada al db buffer cache.

Crear un Shell script `s-02-flash-cache.sh` que realice las siguientes acciones:

A. Crear un directorio `/u04/db_flash_cache`. Esta carpeta será empleada para simular un área de memoria flash que puede ser empleada por el db buffer cache.

B. El directorio `db_flash_cache` deberá pertenecerle al usuario `oracle` y al grupo `oinstall`.
C. Ejecutar el script.

Crear un script `s-03-flash-cache.sql` que realice las siguientes acciones:

A. Generar un nuevo pfile que actúe como respaldo en caso de ocurrir alguna falla.
B. Realizar las configuraciones necesarias para destinar 50MB como tamaño máximo a un archivo de cache llamado `<nombre_instancia>_flash.cache`. La configuración deber ser permanente.
A. Ejecutar el script.
B. Para comprobar los resultados, reiniciar la instancia, verificar que el archivo de cache exista.

1.4. VALIDADOR.

- Copiar todos los scripts de validación en la misma carpeta donde se ubican los scripts del ejercicio. Ejecutar el validador empleando el usuario ordinario del sistema operativo.

```
sqlplus /nolog
start s-06-validador-main.sql
```

1.5. CONTENIDO DE LA ENTREGA.

- C1. Código DDL y resultado al consultar la tabla `t03_db_buffer_cache`
  - Respuesta pregunta A. Implicaciones valores en cero para `current_size, target_size, target_buffers`
  - Respuesta pregunta B. Implicaciones valores en cero para `prev_size y pref_buffers`
  - Respuesta pregunta C. Implicaciones valores en cero para `default_pool_size`
- C2. Código DDL y resultado al consultar la tabla `t04_db_buffer_sysstats`
  - Respuesta pregunta A. Valores de `cache_hit_radio` que indican una correcta cantidad de memoria para el default pool.
  - Respuesta pregunta B Valores de `cache_hit_radio` que indican una cantidad incorrecta de memoria para el default pool.
- C3. Pantalla JMeter
- C4. Gráficas y análisis de resultados.
- C5. Resultado del validador.
- Elementos generales indicados en la rúbrica general de ejercicios prácticos (datos generales, conclusiones y comentarios).
- Entrega individual