

CURSOS
INTERSEMESTRALES



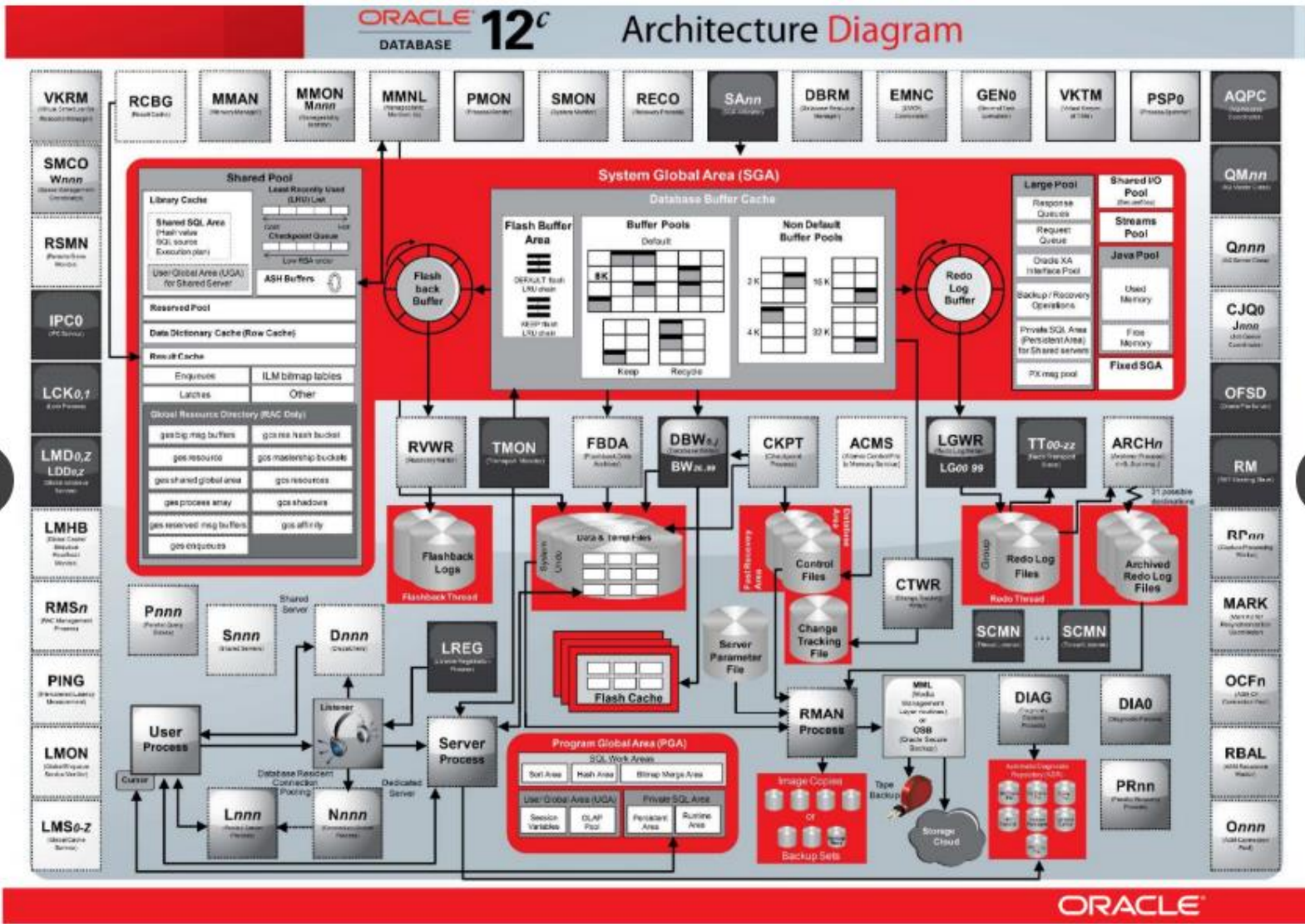
PROTECO

Temas de Administración

SQL 2

24 / Enero / 2020

Arquitectura Completa de una Base de Datos Tipo "Oracle"

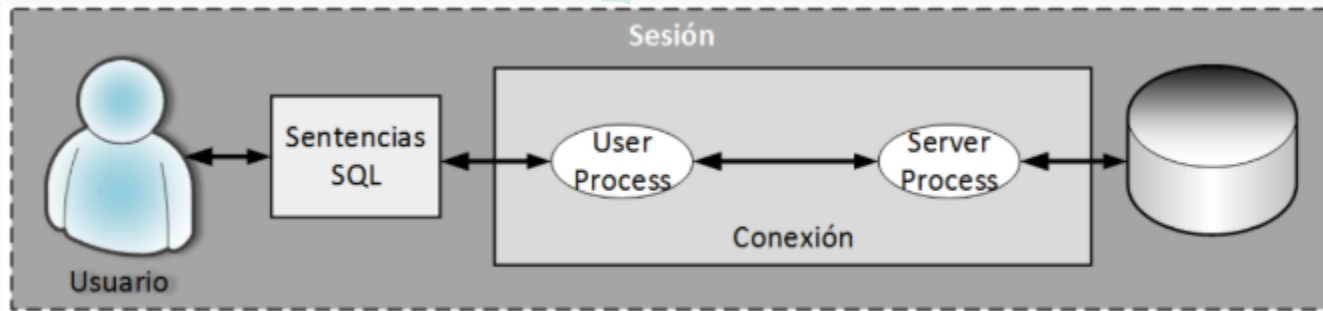


[https://www.oracle.com/webfolder/technetw
ork/tutorials/obe/db/12c/r1/poster/OUTPUT_
poster/poster.html#tab_1](https://www.oracle.com/webfolder/technetw
ork/tutorials/obe/db/12c/r1/poster/OUTPUT_
poster/poster.html#tab_1)



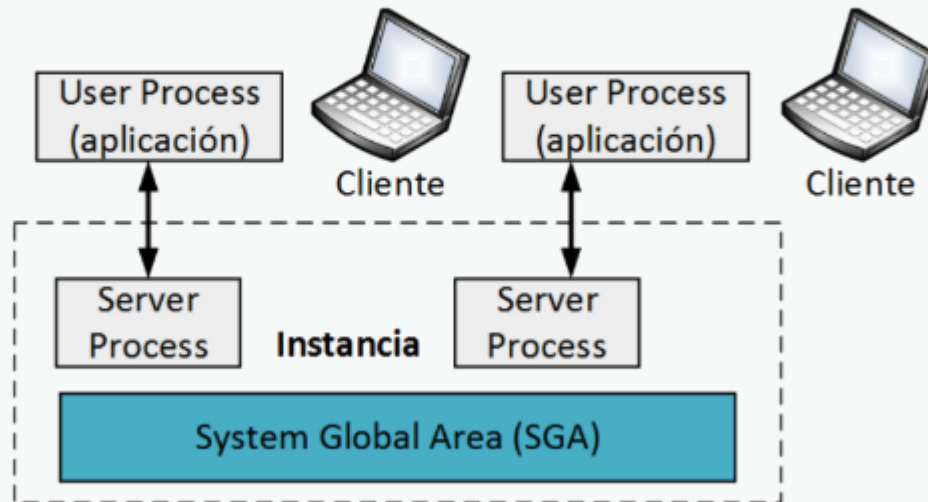
PROTECO

Conexiones hacia una Instancia



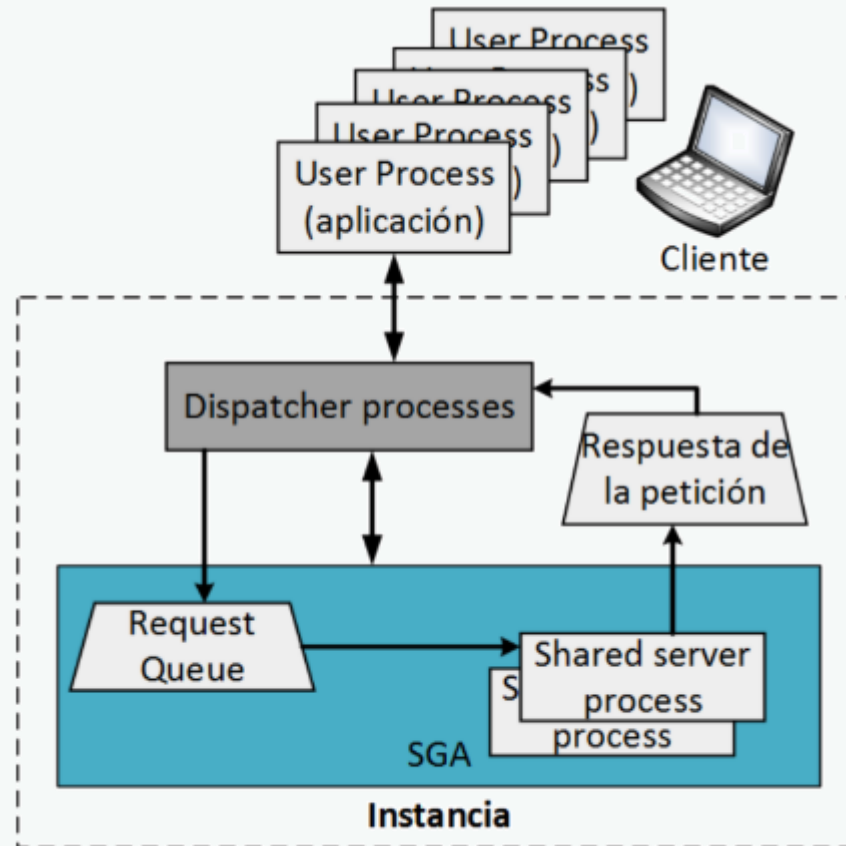
4.4.1. Dedicated Server process

- Se crea un nuevo proceso por cada usuario conectado a la base de datos.



Conexiones hacia una Base

Shared Server process



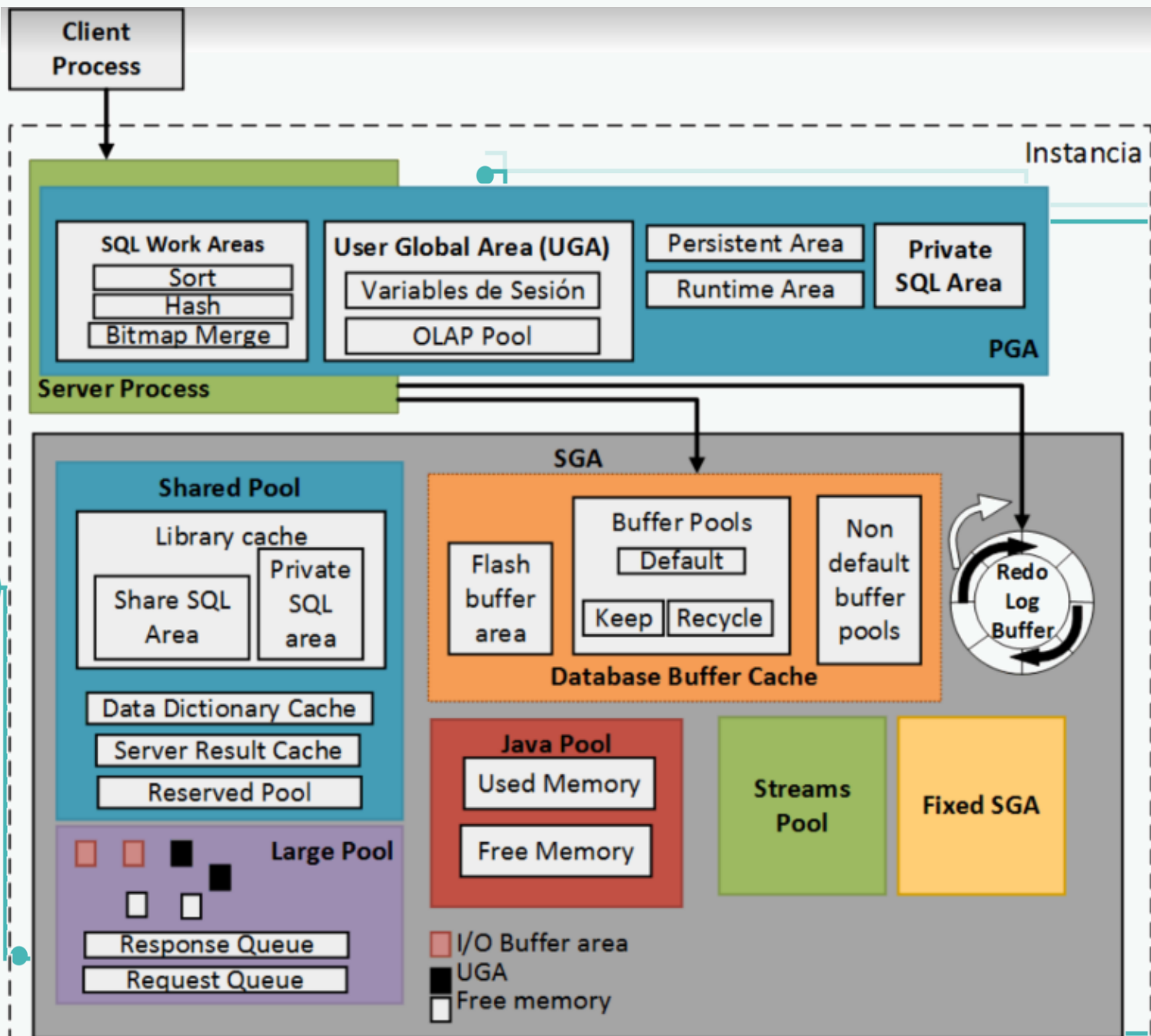
Arquitectura interna de la memoria

Al momento de iniciar una instancia, se crean diversas áreas de memoria, así como diversos procesos de background.

En las áreas de memoria se almacena información de diversos tipos:

- Código de programas
- Información acerca de cada sesión sin importar si está activa o inactiva.
- Información que muestra el status actual de la ejecución de un programa o de una sentencia SQL
- Información de los bloqueos existentes en la base de datos
- Datos en cache: bloques de datos, datos REDO.





Administración de Tablas

Una tabla representa a la unidad básica de almacenamiento en una base de datos. Los datos son almacenados en forma de registros y columnas

La siguiente tabla muestra los tipos de tablas básicas que pueden existir en una base de datos.

Tipo de tabla	Descripción
Heap Organized Table (ordinary)	Tabla básica de propósito general. Sus datos representan a una colección almacenada sin considerar algún orden en particular. Representa la configuración por default.
Clustered table	La tabla forma parte de un "Cluster". Un cluster está formado por un conjunto de tablas que comparte diversos bloques de datos debido a que comparten columnas con datos idénticos.
Index-Organized table	Los datos de la tabla se almacenan empleando una estructura de datos equivalente a un árbol B-Tree. En lugar de crear un índice <i>unique</i> para almacenar los valores de la PK, en cada nodo del árbol se almacenan los valores de todas las columnas.
Partitioned table	Permite dividir el contenido de una tabla en conjuntos de datos llamados particiones . Cada partición puede ser administrada de forma totalmente independiente, puede ser almacenada en su propio tablespace. Esto ofrece diversos beneficios adicionales: mejora en desempeño. Cada partición puede ser configurada de forma independiente, cada uno con sus propios parámetros.



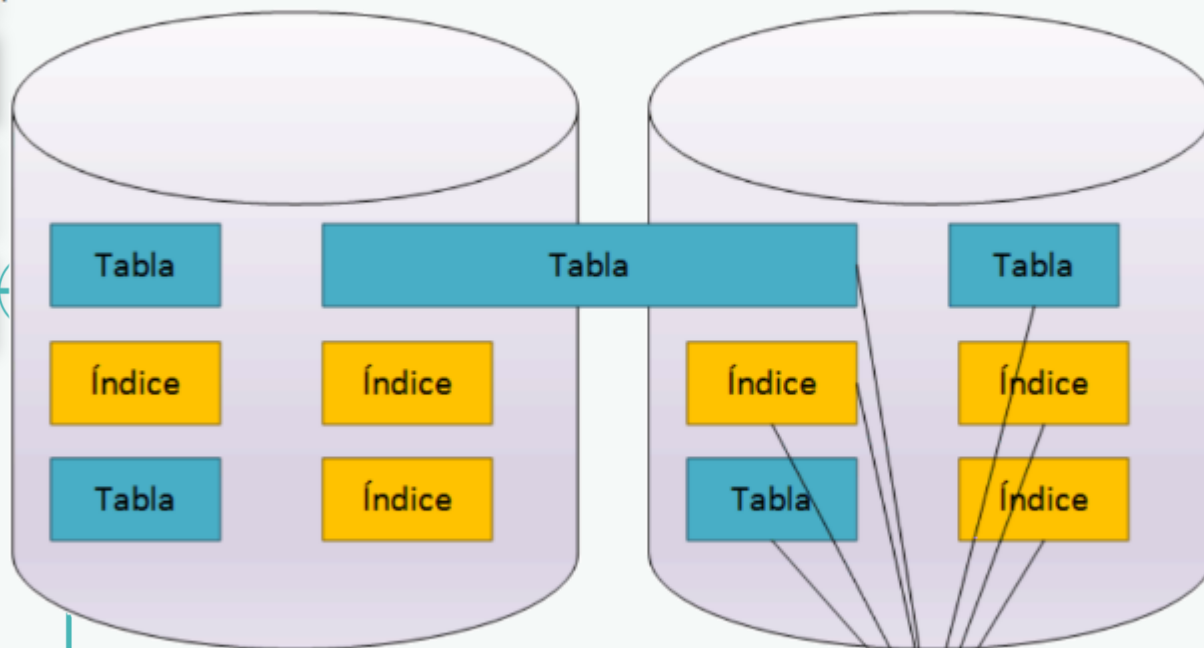
TableSpace

A nivel del sistema operativo, los datos se almacenan en archivos con extensión `dbf` llamados data files. Cada base de datos debe contar con al menos un data file.

- Los datos de cada objeto no particionado (p.e. una tabla) son almacenados en un **segmento**. Cada segmento pertenece a un solo **tablespace**.
- El concepto de *tablespace* y *data file* son similares pero con las siguientes diferencias:
 - Cada tablespace está formado por al menos un data file. Agrupa a un conjunto de data files.
 - Un segmento puede expandirse en varios data files, pero dentro de un único tablespace.
 - Toda base de datos debe contar mínimo con 2 tablespaces: `system` Y `sysaux`. Típicamente se crean de forma adicional los tablespaces `undo` y `temp`.



Tablespace: TS-01
Punto de montaje: /u01/data01
Capacidad: 50GB



Data File:
data01.dbf

Data File:
data02.dbf



PROTECO

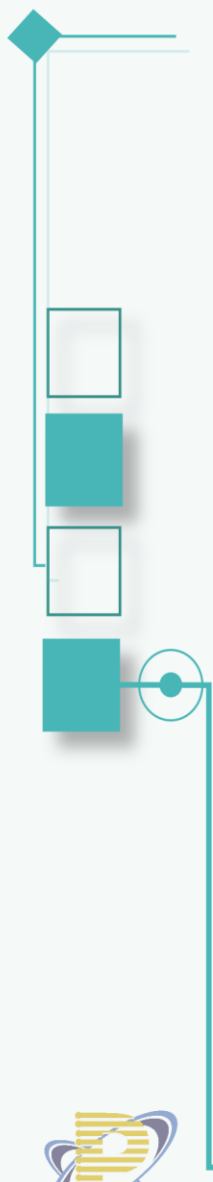
- La cláusula `tablespace` de la instrucción `create table` permite especificar al `tablespace` donde serán almacenados los datos.
- Para tablas particionadas, es posible seleccionar un `tablespace` diferente.
- En caso de no especificar esta cláusula, los datos se almacenarán en el `tablespace` por default definido. Por ejemplo, `users`.
- De no existir un `tablespace` por default asignado a un usuario, los datos se guardarán en el `tablespace system` -> Mala práctica, posibilidad de contención al querer acceder a los datos y a la vez al diccionario de datos, el cual está almacenado en el `tablespace system`.
- Conjuntos de tablas asociadas con una misma aplicación deben ser almacenadas en un mismo `tablespace`. Si se encuentran disgregadas en varios `tablespaces`, los tiempos requeridos para realizar tareas administrativas (backups, recovery) pueden aumentar considerablemente ya que se tiene que ejecutar estas tareas en cada uno de los `tablespaces`.



Al eliminar una tabla con la sentencia `drop table`, ocurre lo siguiente:

- La definición de la tabla se elimina del diccionario de datos.
- Todos los índices y triggers asociados con la tabla son eliminados.
- Todas las vistas y objetos PL/SQL que hacen referencia a la tabla eliminada permanecen pero son marcados como inválidos.
- Todos los sinónimos que apunten a la tabla eliminada permanecen, no son marcados como inválidos, pero fallarán al ser empleados.
- Todas las extensiones que fueron creadas son recuperadas y disponibles como “espacio libre” del tablespace y pueden ser reutilizadas por otro objeto.
- Si la tabla forma parte de un cluster, los registros correspondientes son eliminados del cluster.
- Para eliminar la tabla y sus respectivos constraints de referencia, emplear la cláusula `cascade constraints`.
- Cuando la tabla se elimina, la BD no la elimina de forma inmediata. En realidad la BD la renombra y la mueve a la llamada papelera de reciclaje ***recycle bin***.
- La tabla puede ser recuperada de la papelera empleando la sentencia `flashback table`.
- Para evitar el paso por la papelera, incluir la sentencia `purge` al final de la sentencia `drop table`.





Al proceso de recuperación de una tabla de la papelera de reciclaje se le conoce como ***flashback drop***.

- En realidad, la papelera de reciclaje es una tabla del diccionario de datos que contiene información de los objetos eliminados.
- Por lo anterior, los objetos eliminados siguen ocupando espacio.
- Cualquier usuario puede visualizar a los objetos eliminados en su papelera de reciclaje.

