# Arquitectura de Oracle

## Administración de Bases de Datos

Curso 2018-2019

**Mercedes G. Merayo, Yolanda García, Jesús Correas**

**Departamento de Sistemas Informáticos y Computación**
**Universidad Complutense de Madrid**

.

# Contenido

- Introducción. Propiedades **ACID.**
- Elementos principales de la arquitectura de un SGBD Oracle:
  - Procesos y estructuras en memoria.
  - Estructuras de almacenamiento en disco.
- Estructura lógica y física de almacenamiento.
- Estructura lógica de almacenamiento:
  - **Tablespaces, segments, extents, bloques** (lógicos).
- Estructura física de almacenamiento:
  - **Datafiles, controlfiles, ficheros de redo log.**
- Procesos y estructuras en memoria: **Instancia.**
  - Estructuras en memoria: **SGA, PGA.**
  - Procesos **background, server, client.**

# Arquitectura de una Base de datos - Introducción

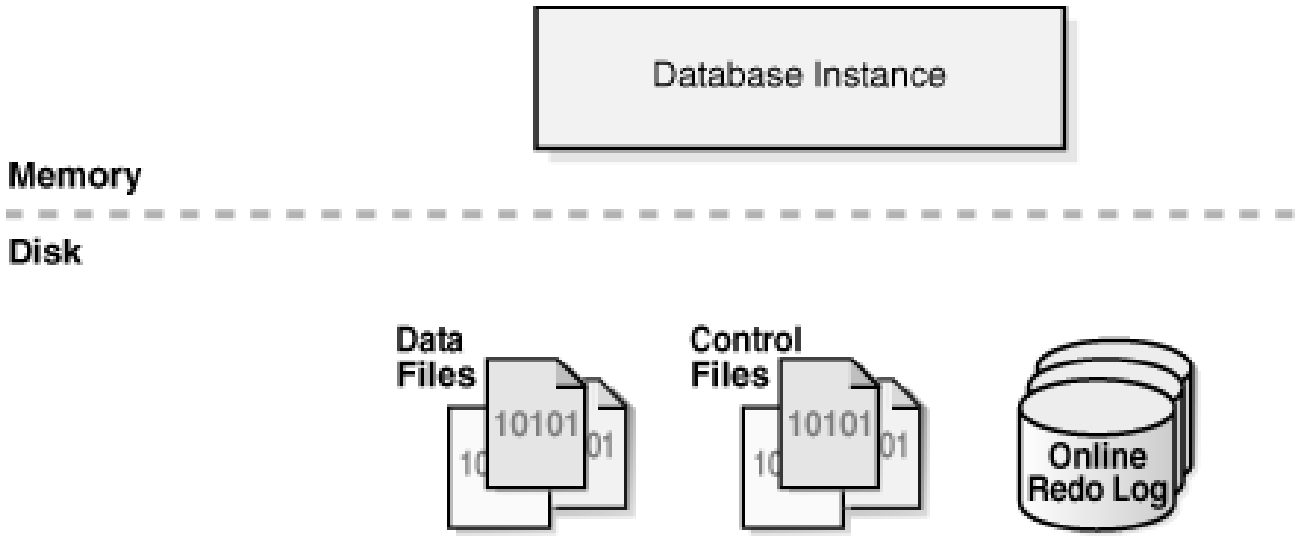
- La arquitectura de un SGBD es clave para tratar adecuadamente la información contenida en una BD.
- Debe poder tratar situaciones muy complejas en gran cantidad de escenarios:
  - Descripción autocontenida de las BD instaladas en el SGBD.
  - Independencia de los dispositivos.
  - Múltiples usuarios que operan sobre la BD concurrentemente.
  - Seguridad de accesos a múltiples BD operando en el mismo SGBD.
  - Seguridad frente a fallos del sistema.
  - Copia de seguridad y recuperación.
  - Instalaciones distribuidas / multiprocesador.
  - Información estadística sobre accesos, eficiencia.
  - Acceso eficiente a los datos.

# Arquitectura de una Base de datos - Propiedades ACID

- Un SGBD de cualquier BD relacional debe cumplir cuatro propiedades denominadas **ACID**:
  - **A de atomicidad:** En una transacción, todas las operaciones se terminan, o bien no se realiza ninguna.
  - **C de consistencia:** Una transacción debe **preservar la consistencia** de la BD. Suele ser responsabilidad del usuario/programador.
    - ⋆ En particular, una consulta debe ser consistente con el estado de la base de datos **en el instante de inicio** de la ejecución de la consulta.
  - **I de aislamiento:** Una transacción no completada (con `COMMIT`) es invisible al resto del mundo.
  - **D de durabilidad:** Cuando se completa una transacción con `COMMIT`, entonces es imposible que la base de datos la pierda.
- Normalmente las propiedades ACID se aplican a las **transacciones** de la BD, pero también es aplicable a la integridad de los datos en general.

# Arquitectura de una BD Oracle - Elementos principales

- Un **servidor de BD** Oracle está formado por dos elementos principales:
  - Una serie de **ficheros en disco** que almacenan los datos de la BD (en la terminología de Oracle es la **database**).
  - Un conjunto de **estructuras en memoria** y de **procesos** que gestionan los ficheros en disco: la **instancia** de la BD.



# Arquitectura de una BD Oracle

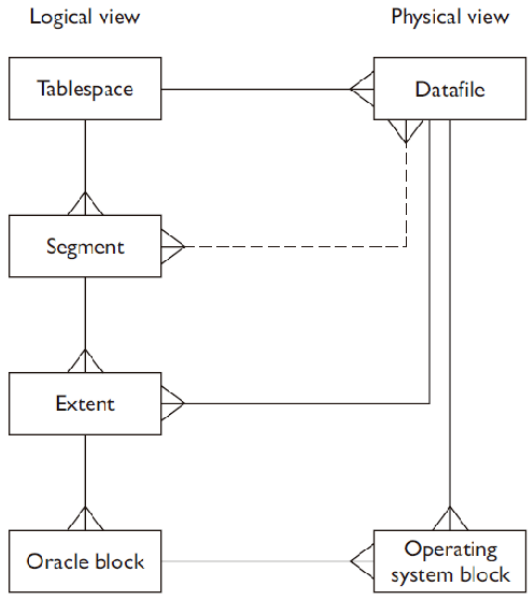
- Oracle proporciona una **abstracción total** del los componentes lógicos respecto a las estructuras y el almacenamiento físico en disco:
- No hay una correspondencia directa entre:
  - los **elementos lógicos:** tablas, vistas, etc.
  - y el **almacenamiento físico:** ficheros de datos (*datafiles*), ficheros de control, etc.
- Los **programadores** y **usuarios** solo tienen acceso a los elementos lógicos de la BD.
- Los **administradores del sistema operativo** gestionan los ficheros en disco.
- El **administrador de la BD** es el único que puede acceder a la **correspondencia entre elementos lógicos y almacenamiento físico.**

# Estructura lógica y física de almacenamiento

- **Desde el punto de vista lógico** del almacenamiento, el elemento fundamental es el **segmento.**
  - Un segmento representa el almacenamiento de un **objeto lógico** de datos en una BD: una **tabla** o un **índice**, pero también de otros tipos (**LOBs**, **partitions**, **undo**, etc.).
  - Otros objetos (procedimientos PL/SQL, vistas, secuencias) no almacenan datos, por lo que no están almacenados en segmentos.
- **Desde el punto de vista físico (del S.O.)**, el elemento fundamental es el **datafile.**
  - Son ficheros del sistema de ficheros del SO.
  - El SO reparte el contenido de los ficheros en **bloques.**

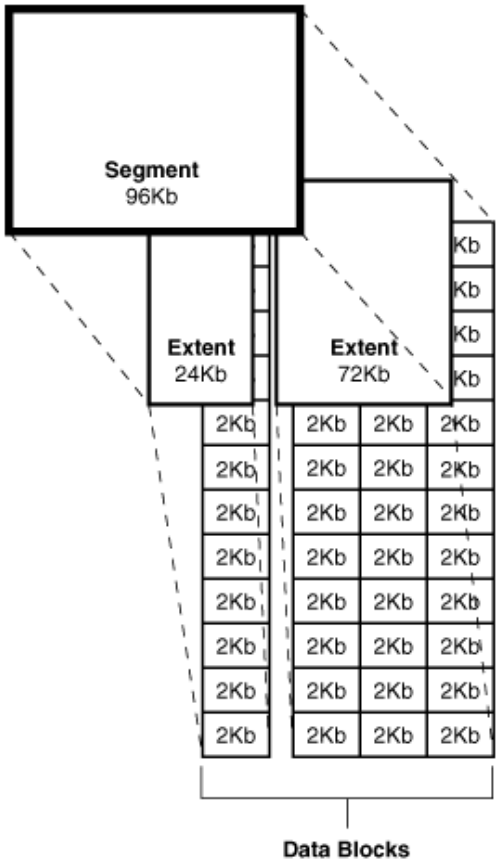
# Estructura lógica y física de almacenamiento

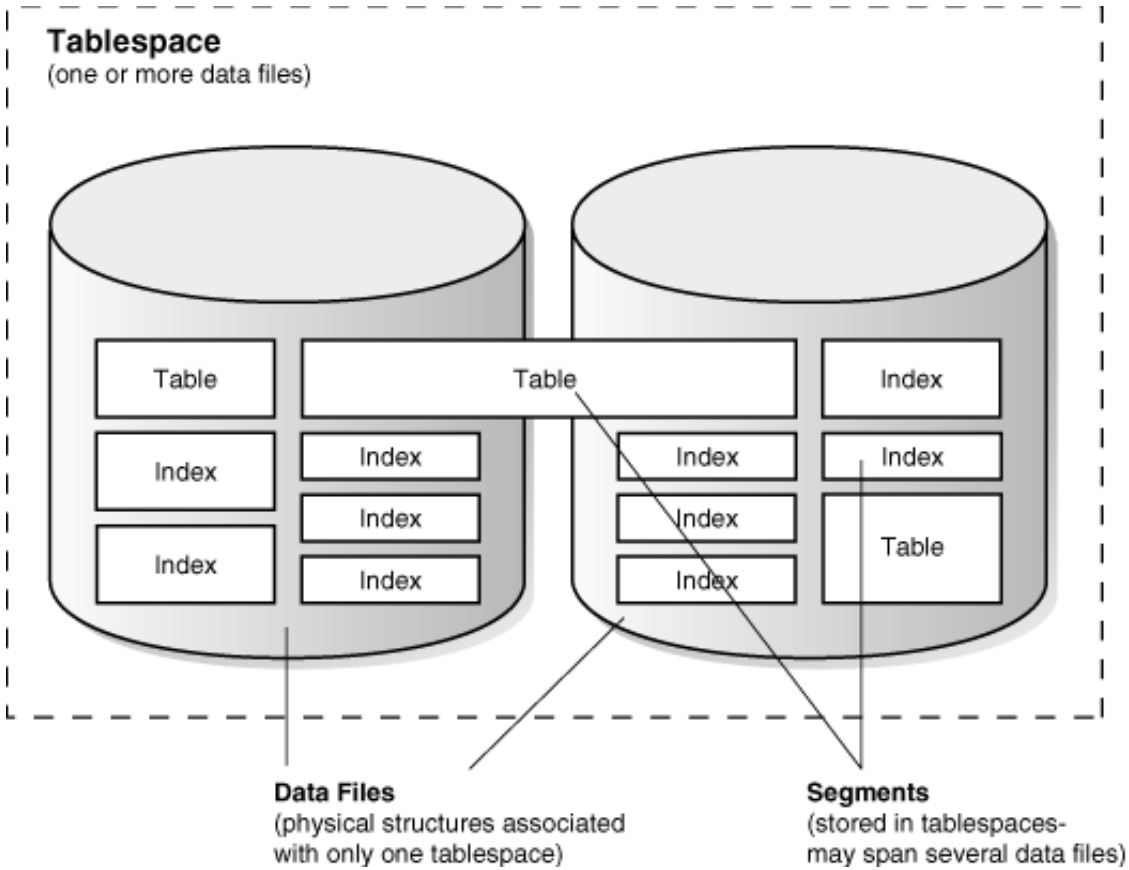
- Los segmentos están formados por una serie de **extents:** secuencias de **bloques consecutivos** en el datafile.
- Un **bloque** de un extent **no se corresponde con un bloque del SO.** Su tamaño es un **múltiplo** del tamaño del bloque del SO.




- **No existe una correspondencia directa** entre **segment** y **datafile**:
  - Un datafile puede contener extents de varios segments, y
  - un segment puede estar repartido entre varios datafiles.
- El **tablespace** es el elemento **lógico** que permite relacionar segments y datafiles.

# Estructura lógica de almacenamiento - Segments y Tablespaces



# Estructura lógica de almacenamiento - Tablespaces

Un **tablespace** es una colección **física** de uno o más datafiles y **lógica** de uno o más segments.

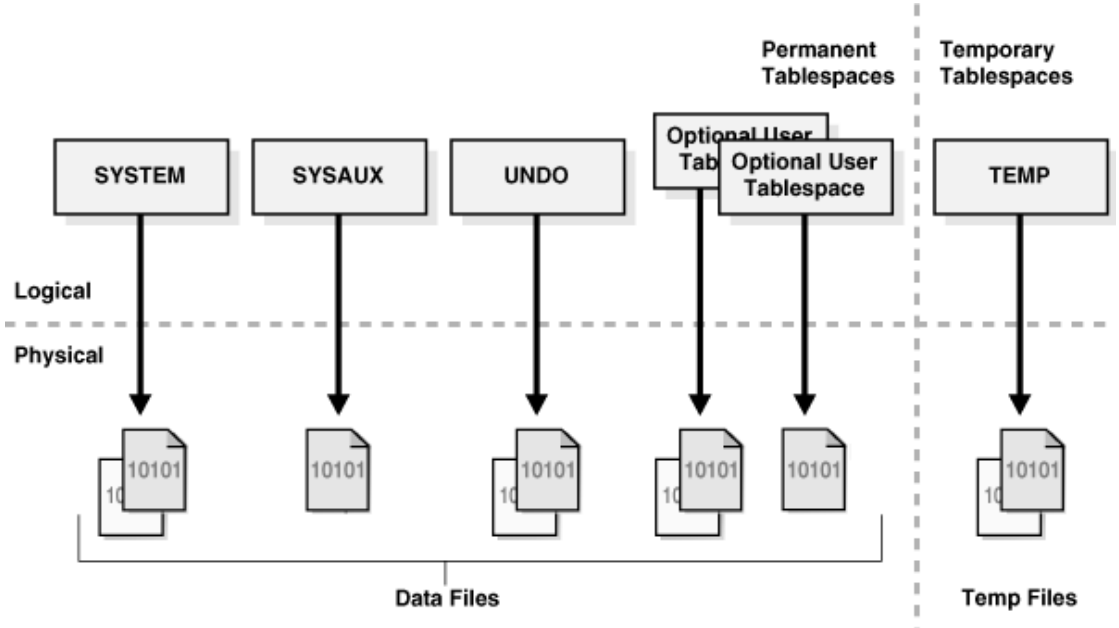
- Permite agrupar los datafiles (y los segments) en unidades lógicas que se tratan a la vez.
- Pueden estar en dos **modos:**
  - **Online** para poder acceder a los datos contenidos.
  - **Offline** para determinadas operaciones de administración y mantenimiento (copias de seguridad, recuperación, replicación, etc.).
- Toda instalación de Oracle debe tener un tablespace particular: **`SYSTEM`.**
  - Contiene el **diccionario de datos** y la información asociada a procedimientos almacenados PL/SQL.
  - Se crea automáticamente durante la creación de la **database.**

# El diccionario de datos

- La relación entre elementos lógicos y almacenamiento físico se encuentra en el **diccionario de datos.**
- Está formado por un conjunto de **tablas y vistas** con información de la propia BD:
  - Definición de los objetos de la BD.
  - Espacio reservado a y utilizado por estos objetos.
  - Valores por defecto para las columnas de las tablas.
  - Restricciones de integridad.
  - Usuarios, privilegios y roles.
  - Información de auditoría.
- La mayor parte de los datos del diccionario están encriptados y no puede accederse a ellos (son las **base tables**). Su propietario es **`SYS`.**
- **No se deben modificar estos datos directamente porque se puede corromper la integridad de la BD.**
- El administrador puede acceder a **vistas** **`DBA_`** (y sinónimos **`"V$"`**) para consultar información del diccionario.

# Estructura lógica de almacenamiento - Tablespaces

- Una instalación típica de Oracle contiene los siguientes tablespaces:




- **SYSAUX** lo utiliza Oracle para algunas tareas internas. Las operaciones básicas del sistema pueden continuar si SYSAUX está offline.

# Estructura lógica de almacenamiento - Tablespaces

- **Read-only tablespaces:**
  - Tablespaces para almacenar datos no modificables.
  - No es necesario gestionar modificaciones ni realizar copias de seguridad.
  - Para modificarlos deben convertirse a **read/write.**
- **Temporary tablespaces:**
  - Para realizar operaciones que no se pueden hacer completamente en memoria: ordenación, join de tablas muy grandes.
  - Se pueden crear tablespaces temporales específicos.
  - Hay un **tablespace temporal por defecto:** en algunos casos se puede utilizar `SYSTEM`, pero es habitual crear uno específico al instalar Oracle.
- **Transportable tablespaces:**
  - Permiten mover tablespaces de una instalación de Oracle a otra.
  - Es más rápido que exportar/importar datos: solo hay que copiar los datafiles y los índices ya están generados.

# Estructura lógica de almacenamiento - Tablespaces

- **Bigfile tablespaces:**
  - Están formados por un solo datafile, pero de tamaño hasta 1024 veces mayor que un datafile normal (*smallfile* –máx. 4M blocks, limitado por SO).
  - El tamaño máximo es 128TB (¡hay que considerar el sistema de backup!).
  - El número máximo de datafiles en una instalación es 64K.
  - Multiplica por 1024 la capacidad máxima de una instalación.
- **Undo tablespaces:**
  - Almacena la información para deshacer transacciones (al ejecutar `ROLLBACK` o recuperar una BD).
  - Mantiene la información previa a la transacción para proporcionar la propiedad ACID de **consistencia** de las consultas concurrentes.
  - Lo gestiona automáticamente Oracle.
  - Pueden ser *bigfile* y pueden crearse varios (uno al menos).

# Estructura lógica de almacenamiento - Segments, extents

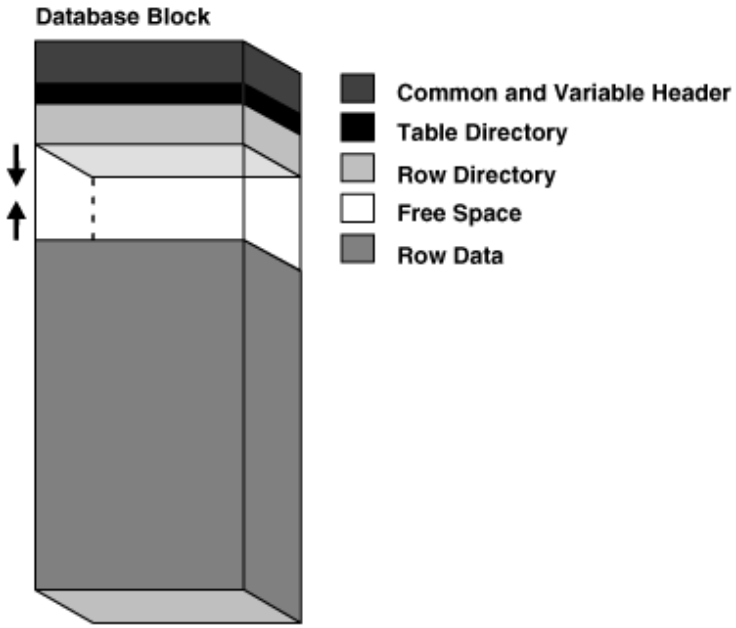
- **Segment:**
  - almacena un **objeto de la BD** (tabla, índice, etc.).
  - Pertenece a un tablespace, pero puede estar repartido en varios datafiles.
  - Se identifican por el nombre del objeto y el propietario.
  - Tipos: **`TABLE`, `INDEX`, `UNDO`, `ROLLBACK`, `TABLE PARTITION`, `INDEX PARTITION`, `LOBSEGMENT`, `LOBINDEX`, `LOBPARTITION`, `CLUSTER`, `NESTED TABLE`.**
  - Se pueden consultar en la vista **`DBA_SEGMENTS`.**
- **Extent:**
  - Un segmento está formado por uno o varios extents.
  - Cuando se llena el último extent de un segmento, se añade uno nuevo.
  - Un extent está formado por bloques contiguos en el mismo datafile.
  - Los extents de un segmento pueden estar en varios datafiles.
  - Se pueden consultar en la vista **`DBA_EXTENTS`.**

# Estructura lógica de almacenamiento - Bloques

- El **bloque** es la unidad mínima de almacenamiento lógico de Oracle.
- Un datafile está formado por bloques lógicos numerados consecutivamente.
- Un extent es una secuencia de bloques consecutivos. Los segmentos crecen asignándoles un nuevo extent.
- El tamaño del bloque es fijo para cada tablespace. No hay relación con el tamaño del bloque del SO (recomienda ser múltiplo).
- El parámetro `DB_BLOCK_SIZE` fija el valor por defecto para el servidor de BD (por defecto 8K).
- Cada bloque tiene una estructura interna para contener datos (filas de tablas, índices, etc.).

# Estructura lógica de almacenamiento - Bloques

- Un bloque puede contener información de varias tablas.
- Se puede almacenar un número variable de filas en cada bloque.




Parámetros de configuración de espacio libre:

- **PCTFREE:** **% libre mínimo** que se reserva para **actualizaciones** de filas (las filas son de longitud variable).
- **PCTUSED:** Cuando se llega al valor de `PCTFREE`, este es el **% de ocupación** al que debe bajar para permitir nuevas **inserciones** de filas.

- Además puede haber referencias (*chains*) a filas en otros bloques, migración de tablas a otros bloques, compresión, etc.
- Más información en *Oracle Database Concepts:*
  `https://docs.oracle.com/cd/E11882_01/server.112/e40540/`

# Estructura lógica de almacenamiento - Gestión de almacenamiento en tablespaces

- El servidor de BD debe mantener información sobre el espacio disponible en los datafiles.
  - Es necesaria cuando se necesita un nuevo extent para un segmento.
  - Se debe actualizar cuando se libera un extent.
- Esta información se puede mantener de dos formas:
  - **Dictionary-managed tablespaces:** Se mantiene en el diccionario (obsoleto).
  - **Locally managed tablespaces:** Cada datafile mantiene un *bitmap* con los bloques (o grupo de bloques) disponibles.
- Los extent pueden ser de tamaño fijo o de tamaño variable calculado por el sistema.
  - Cláusulas **`UNIFORM`** o **`AUTOALLOCATE`** de la instrucción `CREATE TABLESPACE`.

# Estructura física de almacenamiento - Datafiles

- el **datafile** es el elemento básico de **almacenamiento físico (del SO)** de una instalación de Oracle.
  - Son los ficheros físicos de datos observables desde el SO.
  - Inicialmente no contiene datos, pero Oracle reserva espacio para datos en el futuro.
  - El tamaño se fija al crearlo, pero se puede modificar posteriormente o especificarse como `AUTOEXTEND`.
  - Oracle también permite utilizar directamente discos sin pasar por el SO: *raw devices* o con su propio sistema de ficheros (ASM).
  - Además permite gestionar datafiles en múltiples máquinas, *clusters* de máquinas (RAC), etc.
- Hay otros dos tipos de **ficheros fundamentales:**
  - Los **controlfiles.**
  - Los ficheros de *redo log*.

# Estructura física de almacenamiento - Controlfiles

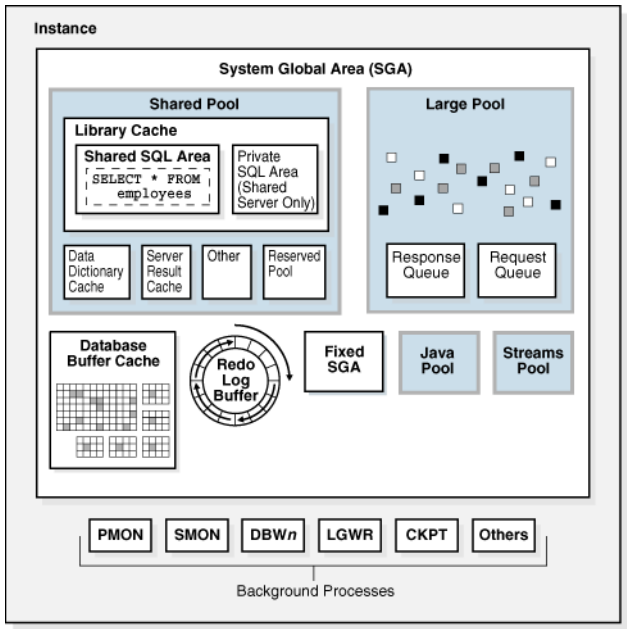
- El **controlfile** es un fichero binario que contiene información vital sobre la estructura de la instalación de Oracle:
  - Nombre e identificador del **servidor de BD.**
  - Ubicación de los demás ficheros de la instalación (datafiles, ficheros de *redo log*).
  - Información sobre los tablespaces.
  - Información de copia de seguridad y recuperación (*checkpoints*).
- Cada instalación tiene un controlfile, pero es tan importante que **se deben mantener varias copias** (hasta 8, en distintos dispositivos): están **multiplexados.**
- Las copias se mantienen sincronizadas automáticamente.

# Estructura física de almacenamiento - Ficheros de *redo log*

- Cuando se ejecutan sentencias SQL que modifican datos, los cambios **no se hacen en los datafiles,** sino que se modifican en copias en memoria.
- Las actualizaciones en disco se hacen en algún momento posterior por procesos especiales (**`DBWn`**).
- Si hay un fallo en el servidor, puede perderse gran cantidad de operaciones que no se han llegado a escribir en los datafiles.
- Para recuperar la información se utilizan los **ficheros de redo log.**
  - Son ficheros secuenciales en los que se escriben todas las operaciones de modificación de datos (`INSERT`, `UPDATE`, `DELETE`).
  - Como el controlfile, pueden estar **multiplexados:** varios ficheros de *redo log* se escriben con la misma información en distintos discos.
  - Además, distintos grupos de *redo log* van **rotando** para poder hacer copia de seguridad de los ficheros (si la BD está en modo *archivelog*).
  - El grupo que se está escribiendo se denomina **redo log online.**

# Procesos y estructuras en memoria - Instancia

- La **instancia** de un servidor de BD está formado por:
  - Un conjunto de estructuras de datos en memoria: la **System Global Area (SGA).**
  - Un conjunto de **procesos en background.**
  - Además, cada proceso tiene su propia **Program Global Area (PGA),** con estructuras de datos específicas.




- En instalaciones distribuidas (RAC) se utilizan múltiples instancias.

# Procesos y estructuras en memoria - SGA

- **Database Buffer Cache:**
  - Cuando se ejecuta una sentencia SQL se copian en *buffers* de este área los bloques afectados.
  - Las modificaciones de sentencias DML se realizan en los *buffers* en memoria.
  - No se escriben en disco mientras no se necesita el espacio en memoria. Los procesos `DBWn` realizan la escritura en disco.
  - **Dirty buffer** es un *buffer* modificado en memoria que no se ha guardado en disco.
- El tamaño de este área es **fundamental para la eficiencia:**
  - Si el tamaño no es suficiente, se incrementará el acceso a disco para escribir bloques recientes.
  - Si el tamaño es demasiado grande, habrá bloques en memoria que se acceden raramente.

# Procesos y estructuras en memoria - SGA

- **Shared pool:** Es la estructura más compleja. Contiene diversas subestructuras:
  - **Library cache:** Almacena las instrucciones SQL preparadas para acelerar las consultas que se repiten. Ahorra accesos al diccionario de datos.
  - **Data dictionary cache:** Almacena las últimas definiciones utilizadas: tablas, índices, privilegios, usuarios, etc.
  - **PL/SQL area:** objetos PL/SQL que se leen del diccionario de datos para evitar repetir accesos.
  - **SQL Query** y **PL/SQL Function Result Cache:** Resultados de las últimas ejecuciones de SQL y PL/SQL:
    - ⋆ Si no se han modificado los datos o parámetros de los procedimientos, reutiliza los resultados de consultas y llamadas.

# Procesos y estructuras en memoria - SGA

- **Log Buffer:**
  - Es un *buffer* circular en el que se guardan los datos que se van escribir en los ficheros de *redo log*.
  - El proceso **LGWR** escribe en disco el contenido de este buffer por lotes y cuando se ejecuta una sentencia `COMMIT`.
  - Se escribe simultáneamente en todos los ficheros *redo log* del grupo online.
- **Large Buffer:**
  - Se utiliza para determinadas operaciones en memoria demasiado grandes para el *shared buffer*.
  - Es opcional, pero facilita operaciones como la copia de seguridad con RMAN.
- **Java Buffer:**
  - Almacena datos y código de los procesos Java que se ejecutan en el servidor de BD. Por ejemplo, el código de las clases Java, compartido entre todos los procesos.

# Procesos y estructuras en memoria - Procesos

- Las aplicaciones de usuario habitualmente se ejecutan en procesos de **máquinas distintas** del servidor de BD:
  - **Client processes:** procesos que ejecutan el código de aplicaciones que hacen consultas al servidor de BD.
  - Por ejemplo, SQL*Plus o SQL Developer ejecutan en procesos cliente.
- La instancia se ejecuta en el servidor de BD y está Formada por gran número de procesos ejecutando concurrentemente:
  - **Background processes:** realizan tareas internas de la BD: inician la instancia, escriben datos en datafiles, ficheros de *redo log*, recuperación de datos, etc.
  - **Server processes:** cada proceso cliente se conecta con uno de estos en la máquina del servidor para ejecutar las operaciones en la BD. (Oracle se puede configurar para tener un solo *shared server process* al que se conectan todos los clientes).
- Todos los procesos del servidor acceden a la SGA y pueden enviarse notificaciones (*signals*) entre sí.
- Algunos procesos en *background* relevantes:

# Procesos y estructuras en memoria - *Background processes*

- **PMON** (Process Monitor): Monitoriza los demás procesos *background* y realiza la recuperación cuando un proceso servidor termina de forma anormal, liberando recursos y bloqueos y limpiando la SGA.
- **SMON** (System Monitor): Tareas diversas:
  - Si es necesario, recuperación de la instancia en el arranque.
  - Recuperación de transacciones terminadas al recuperar la instancia.
  - Limpieza de segmentos temporales, desfragmentación de extents, etc.
- **DBWn** (DB writer): Escribe los *buffers* de la BD modificados en memoria (*dirty*) en los datafiles. Se pueden crear múltiples procesos `DBWn` (sólo si multiprocesador).
- **LGWR** Escribe en los ficheros de *redo log* online. Cuando se llenan:
  - Rota los ficheros al siguiente grupo,
  - Pone los anteriores offline y notifica a los procesos `ARCn`.
- **ARCn** Archivan los ficheros de *redo log* offline.
- Otros procesos: **CKPT** (*checkpoints*), **FBDA** y **RVWR** (*flashback*), **MMON** y **MMNL** (AWR), **CJQ0** y **Jnnn** (*jobs*), etc.

# Procesos y estructuras en memoria - Resumen