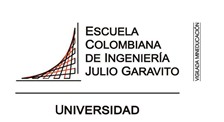
****

**Escuela Colombiana de Ingeniería Julio Garavito**

**Modelos y bases de datos 2024-1**

**Autoestudio V**

**Cristian Santiago Pedraza Rodríguez**

**Andersson David Sánchez Méndez**

**23 de abril de 2024**

# 

# MODELOS Y BASES DE DATOS

**PL/ SQL Básico 2024-1**

# Guía autoestudio 5/6

## OBJETIVO

1. Conocer herramientas que facilitan el trabajo del desarrollador de una base de datos específicamente la herramienta SQL Developer
2. Desarrollar competencias para definir e implementar los servicios que puede ofrecer una base de datos cumpliendo requisitos de concurrencia, seguridad y recuperación.

## TÓPICOS OBJETIVO

1. Transacciones para concurrencia y recuperación
2. Permisos como mecanismos de seguridad

# ENTREGA

Publicar las respuestas en el espacio correspondiente en un archivo .zip , el nombre de este archivo debe ser la concatenación en orden alfabético de los primeros apellidos de cada uno de los miembros.

## TRANSACCIONES

1. **Transacciones**
   1. ¿Cómo se define el comienzo y fin de una transacción en ORACLE?

* El comienzo de una transacción se define mediante la sentencia **BEGIN TRANSACTION** o simplemente iniciando una operación DML (Data Manipulation Language) como **INSERT, UPDATE** o **DELETE**.
* El fin de una transacción se define mediante la sentencia COMMIT para confirmar los cambios realizados, o mediante la sentencia **ROLLBACK** para deshacer los cambios y revertir la base de datos a su estado anterior.
  1. ¿Cuáles son los diferentes tipos de aislamiento que soporta ORACLE? Para cada uno de ellos detalle, ¿cómo maneja los bloqueos? ¿qué problemas resuelve?
* Oracle admite varios niveles de aislamiento de transacciones, incluyendo:
  + Serializable: Evita que otras transacciones realicen cambios en los datos leídos por la transacción actual hasta que se completa la transacción actual.
  + Read Committed: Permite que las transacciones vean solo los datos confirmados por otras transacciones.
  + Read Uncommitted: Permite que una transacción vea los datos que están siendo modificados por otras transacciones, aunque aún no se hayan confirmado.

Para cada tipo de aislamiento, Oracle maneja los bloqueos de diferentes maneras para garantizar la coherencia de los datos y evitar problemas como la lectura sucia, la lectura fantasma y la escritura fantasma.

* 1. ¿Cuál es el tipo de aislamiento por defecto en ORACLE?
* El tipo de aislamiento por defecto en Oracle es el nivel de aislamiento "Read Committed". En este nivel, las transacciones ven solo los datos confirmados por otras transacciones.

## Vistas

* 1. ¿Cuáles son los mecanismos para la creación y borrado de vistas en ORACLE?
* Creación de vistas: Las vistas se crean utilizando la sentencia **CREATE VIEW**. Por ejemplo: **CREATE** **VIEW** nombre\_vista **AS SELECT** columna1, columna2 **FROM** tabla **WHERE** condicion;.
* Borrado de vistas: Las vistas se eliminan utilizando la sentencia DROP VIEW. Por ejemplo: DROP VIEW nombre\_vista;.
  1. ¿Cuáles son las restricciones de las vistas en ORACLE?
* Las vistas en Oracle tienen algunas restricciones:

1. No se pueden modificar datos a través de una vista si la vista involucra funciones de grupo (**GROUP BY**, **HAVING**) o funciones de conjunto (**UNION**, **MINUS**, **INTERSECT**).
2. Las columnas calculadas o las expresiones de columna no pueden actualizarse.
3. No se pueden establecer restricciones de clave externa en una vista.
4. No se pueden indexar vistas en Oracle antes de la versión 12c.

## Modularidad Paquetes

* 1. ¿Para qué sirve un paquete?

Colección de procedimientos almacenados, funciones y otros objetos de base de datos que están agrupados juntos. Algunas características son:

* **Organización y modularidad**: Facilitan la gestión y mantenimiento del código, ya que las funciones y procedimientos relacionados se agrupan en un solo lugar.
* **Reutilización de código**: Reducen la duplicación de código y mejora la coherencia y la eficiencia del desarrollo.
* **Seguridad**: Los permisos se pueden asignar al paquete en su conjunto, lo que permite un mejor control sobre quién puede ejecutar y modificar los objetos contenidos en él.
* **Rendimiento**: Reducen el tiempo de compilación y ejecución de las consultas.
* **Transacciones atómicas**: Conjunto de operaciones dentro de un paquete se ejecutará completamente o no se ejecutará en absoluto. Esto garantiza la consistencia de los datos y ayuda a mantener la integridad de la base de datos.
* **Procedimientos y funciones encapsulados**: Ayudan a ocultar la complejidad y proporciona una interfaz clara para interactuar con la base de datos.
* **Variables locales**: Visibles solo dentro del paquete para evitar colisiones de nombres y ayudar a mantener el estado del paquete.
* **Especificación y cuerpo**: Declara los procedimientos y funciones disponibles (especificación), y la implementación (cuerpo).
* **Sobrecarga de procedimientos**: Varios procedimientos o funciones con el mismo nombre pero con diferentes listas de parámetros.
  1. ¿Cuales son los mecanismos para la creación, invocación, modificación y borrado de paquetes en ORACLE?

En Oracle, los paquetes se crean, invocan, modifican y eliminan utilizando diversos comandos y procedimientos.

* **Creación de paquetes**:

CREATE PACKAGE: Crear la especificación del paquete, donde se definen los procedimientos, funciones, variables y tipos de datos que formarán parte del paquete.

CREATE PACKAGE BODY: Después de crear la especificación del paquete, se utiliza este comando para crear el cuerpo del paquete, donde se implementan los procedimientos y funciones definidos en la especificación.

Ejemplo:

CREATE OR REPLACE PACKAGE mi\_paquete AS

PROCEDURE procedimiento\_1;

FUNCTION funcion\_1 RETURN NUMBER;

END mi\_paquete;

CREATE OR REPLACE PACKAGE BODY mi\_paquete AS

PROCEDURE procedimiento\_1 IS

BEGIN

-- Código del procedimiento

END procedimiento\_1;

FUNCTION funcion\_1 RETURN NUMBER IS

BEGIN

-- Código de la función

RETURN 0;

END funcion\_1;

END mi\_paquete;

* **Invocación de paquetes**:

Los procedimientos y funciones dentro de un paquete se pueden invocar como cualquier otro procedimiento o función de base de datos, especificando el nombre del paquete y el nombre del procedimiento o función.

Ejemplo:

BEGIN

mi\_paquete.procedimiento\_1;

DBMS\_OUTPUT.PUT\_LINE('Resultado: ' || mi\_paquete.funcion\_1);

END;

* **Modificación de paquetes**:

Para modificar un paquete existente, se utiliza el comando `ALTER PACKAGE`. Sin embargo, Oracle no permite modificar directamente la especificación de un paquete existente. Toca eliminar y volver a crear el paquete o usar la cláusula `ALTER PACKAGE` para agregar o eliminar elementos solo en el cuerpo del paquete.

Ejemplo:

ALTER PACKAGE mi\_paquete ADD PROCEDURE nuevo\_procedimiento;

* **Borrado de paquetes**:

Para eliminar un paquete, se utiliza el comando `DROP PACKAGE`.

Ejemplo:

DROP PACKAGE mi\_paquete;

## SYS\_REFCURSOR

* 1. ¿Qué es un SYS\_REFCURSOR?¿Para qué sirve?

Es un tipo de dato especial en Oracle que se utiliza para devolver conjuntos de resultados desde una función o procedimiento almacenado. Es una especie de puntero a un conjunto de filas de una tabla que puede ser manipulado en programas PL/SQL.

Se puede pensar como una especie de "caja" que contiene un conjunto de filas de una tabla, pero esta caja no contiene los datos directamente, sino un puntero que apunta a los datos. Esto permite que los datos sean recuperados y procesados de forma eficiente dentro de un procedimiento o función PL/SQL.

Para usar este tipo de dato, lo primero que hay que hacer es declararlo en el bloque PL/SQL y luego abrirlo para asociarlo con una consulta. Una vez abierto, se pueden recorrer sus resultados fila por fila y procesarlos según sea necesario. Después de haber usado el cursor, se debe cerrar para liberar los recursos.

* 1. ¿Cómo se define, se asigna y se retorna?

* **Definición**:

Primero, se define un `SYS\_REFCURSOR` utilizando la declaración `REFCURSOR`:

TYPE ref\_cursor IS REF CURSOR;

* **Asignación**:

Después de definir el tipo de dato, se puede usar para declarar una variable de tipo `SYS\_REFCURSOR` y asignarle un cursor abierto:

DECLARE

v\_cursor ref\_cursor;

BEGIN

OPEN v\_cursor FOR

SELECT columna1, columna2 FROM tabla WHERE condicion;

END;

* **Retorno**:

Para devolver un `SYS\_REFCURSOR` desde una función o procedimiento, se puede usar como parámetro de salida. Por ejemplo:

CREATE OR REPLACE FUNCTION obtener\_datos

RETURN ref\_cursor IS v\_cursor ref\_cursor;

BEGIN

OPEN v\_cursor FOR

SELECT columna1, columna2 FROM tabla;

RETURN v\_cursor;

END obtener\_datos;

También se puede usar un `SYS\_REFCURSOR` como parámetro de salida en un procedimiento:

CREATE OR REPLACE PROCEDURE mostrar\_datos (p\_cursor OUT ref\_cursor) IS

BEGIN

OPEN p\_cursor FOR

SELECT columna1, columna2 FROM tabla;

END mostrar\_datos;

Luego se puede llamar al procedimiento y trabajar con el cursor devuelto:

DECLARE

v\_cursor ref\_cursor;

v\_columna1 tabla.columna1%TYPE;

v\_columna2 tabla.columna2%TYPE;

BEGIN

mostrar\_datos(v\_cursor);

LOOP

FETCH v\_cursor INTO v\_columna1, v\_columna2;

EXIT WHEN v\_cursor%NOTFOUND;

-- Hacer algo con los datos

DBMS\_OUTPUT.PUT\_LINE('Columna1: ' || v\_columna1 || ', Columna2: ' || v\_columna2);

END LOOP;

CLOSE v\_cursor;

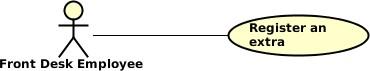
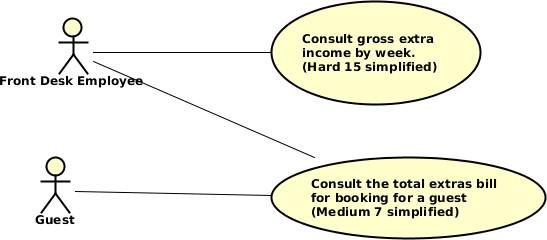
END;

El procedimiento `mostrar\_datos` abre un cursor y lo asigna al parámetro `p\_cursor`, que es un `SYS\_REFCURSOR`. Luego, en el bloque anónimo, el cursor se recorre para procesar los resultados.

Referencias

* Oracle Documentation: [Transactions](https://docs.oracle.com/en/database/oracle/oracle-database/19/sqlrf/Database-Concepts.html#GUID-55B6C5F2-53E2-4B19-AF91-88C25A6226FF), [SQL Statements for Transactions](https://docs.oracle.com/en/database/oracle/oracle-database/19/sqlrf/SQL-Statements-for-Transactions.html)
* Oracle Documentation: [Creating a View](https://docs.oracle.com/cd/E11882_01/server.112/e41084/statements_8004.htm#SQLRF01504), [Dropping a View](https://docs.oracle.com/cd/E11882_01/server.112/e41084/statements_8005.htm#SQLRF01505), [Restrictions on Updatable Views](https://docs.oracle.com/cd/E11882_01/server.112/e41084/statements_8004.htm#SQLRF01513)
* <https://www.ibm.com/docs/es/db2/11.1?topic=support-packages-plsql>

## PRACTICANDO.



El caso de uso Register an extra está definido en el autoestudio 4.

La primera consulta corresponde a la consulta hard 15 incluyendo únicamente extras

La segunda consulta corresponde a la consulta medium 7 incluyendo únicamente extras

Las consultas retornan un CURSOR (ayuda: SYS\_REFCURSOR Ver moodle)

**NOTAS**

Consultar [especificaciones de entrega](https://campusvirtual.escuelaing.edu.co/moodle/mod/wiki/view.php?id=59242) Componente

## A. Ofreciendo servicios

1. Implemente los paquetes de componentes necesario para ofrecer las funciones básicas y consultas del ciclo actual del sistema (CRUD).

## PC\_EXTRAS[Consultar diseño al final]

(CRUDE (la especificación) , CRUDI (la implementación) )

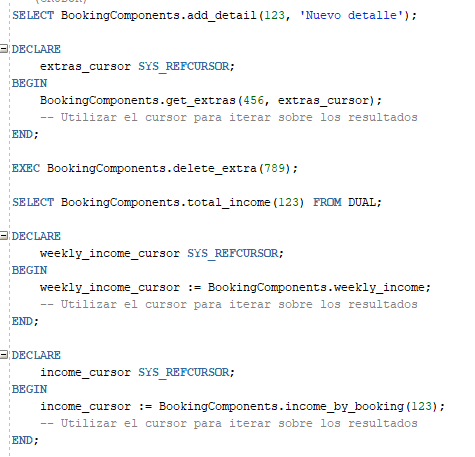








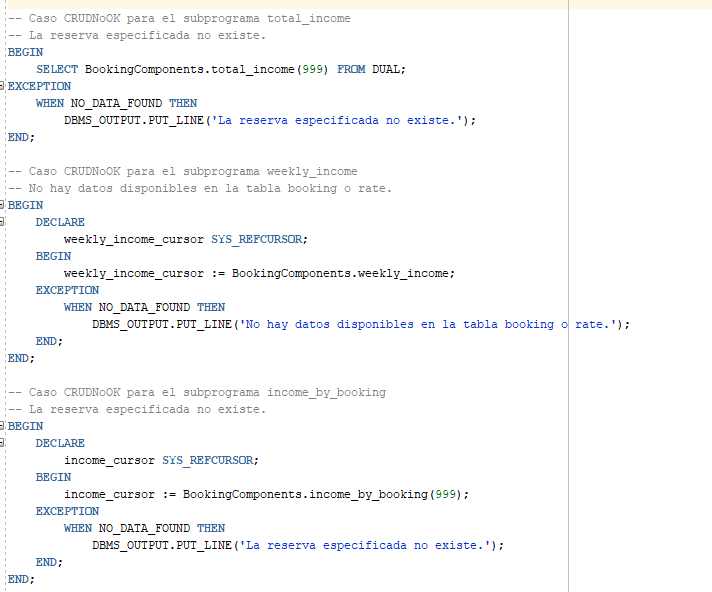
1. Proponga un caso de prueba exitoso por subprograma. (son seis) (CRUDOK)

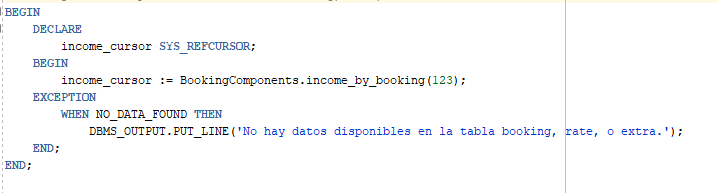


1. Proponga tres casos en los que el subprograma no se puede ejecutar. (CRUDNoOK)









1. Escriba las instrucciones necesarias para eliminar los paquetes. (CRUDX)



