Asignatura:

Teoría de bases de datos

Catedrático(a):

Ricardo Enrique Lagos Mendoza

Proyecto

Alumnos:

Jose Noe Leon Perez 116450005
Maby Esther Santos 120450100
Dania Maritza Gomez Sanchez 121250001
Denis Ortez Carranza- 221020007
Jose Wilberto Espinoza 119160010
Alex Fernando Maldonado Cerrato 121630033
Adolfo Jose Zuniga Santos 120450280
Jessy Geraldina Gonzales Reyes 122290064
Angie Carolina Hernandez 121070003

Fecha:

05/04/2025

ÍNDICE

Introducción

Las bases de datos juegan un papel fundamental en la organización, almacenamiento y gestión de la información en distintos sectores. SQL (Structured Query Language) es el lenguaje estándar para la manipulación y gestión de bases de datos relacionales, permitiendo realizar operaciones como la creación, modificación, consulta y eliminación de datos. Este informe tiene como objetivo desarrollar los conceptos esenciales de SQL, su aplicación en bases de datos y la optimización del rendimiento mediante diversas técnicas.

Planteamiento del Problema

En el manejo de bases de datos, es crucial comprender la estructura y la funcionalidad de SQL para garantizar una gestión eficiente de la información. Muchas organizaciones requieren que los datos sean fácilmente accesibles, seguros y organizados de manera óptima. Sin embargo, la falta de conocimientos sobre SQL y sus funcionalidades avanzadas puede generar problemas de integridad, rendimiento y seguridad en la administración de bases de datos.

Objetivos

Objetivo General

- Comprender los fundamentos del lenguaje SQL y su aplicación en la gestión de bases de datos relacionales.

Objetivos Específicos

- Explicar la creación, modificación y eliminación de datos en bases de datos.
- Analizar la importancia de índices, claves primarias y foráneas en la optimización de consultas.
- Explorar las funciones de agregación y la manipulación avanzada de datos en SQL.
- Evaluar los métodos de compresión, eliminación y restauración de datos en SQL.

El lenguaje SQL

Es la herramienta principal en la gestión de bases de datos relacionales. Se basa en un conjunto de sentencias estructuradas que permiten manipular y gestionar la información de manera eficiente. Entre sus principales funcionalidades se encuentran:

- **Creación de Tablas:**

  Las bases de datos están compuestas por tablas que almacenan registros en filas y columnas. Cada tabla debe tener una clave primaria para garantizar la unicidad de los datos y evitar redundancias. Además, se pueden definir claves foráneas para establecer relaciones entre distintas tablas.

- **Creación de Triggers:**

  Los triggers son procedimientos almacenados que se ejecutan automáticamente ante ciertos eventos en una tabla, como inserciones, actualizaciones o eliminaciones. Estos garantizan la integridad y coherencia de los datos.

- **Inserción de Datos:**

  SQL permite poblar la base de datos con información relevante y posteriormente recuperarla mediante la sentencia SELECT, que incluye filtros avanzados, ordenamientos y agrupaciones.

- **Indexación de Columnas:**

  Mejora el rendimiento de las consultas al acelerar la búsqueda de datos. Un índice es una estructura que almacena referencias a los registros de una tabla, permitiendo una recuperación más rápida de la información.

- **Consultas de Datos:**

  Se utilizan para recuperar información almacenada en la base de datos mediante la sentencia SELECT, que puede incluir condiciones, agrupaciones y ordenamientos para obtener resultados específicos.

- **Adición de Columnas:**

  Es posible modificar la estructura de una tabla agregando nuevas columnas sin perder los datos existentes, lo que permite ampliar la base de datos según las necesidades del sistema.

- **Consulta del Diccionario de Datos de Oracle:**

  En sistemas como Oracle, se puede acceder al diccionario de datos, que almacena información sobre la estructura y los objetos de la base de datos, facilitando la administración y optimización de la misma.

- **Actualización de Datos:**

  SQL permite actualizar registros existentes sin necesidad de eliminarlos y volver a insertarlos, garantizando la coherencia de la información almacenada.

- **Consultas Agregadas:**

  Facilitan la realización de cálculos sobre un conjunto de datos, como promedios, sumatorias, conteos y valores máximos o mínimos.

- **Compresión de Datos:**

  Optimiza el almacenamiento de la información para mejorar el rendimiento del sistema. Esto es especialmente útil en bases de datos de gran tamaño donde la eficiencia en la administración del espacio es crucial.

- **Eliminación de Datos y Tablas:**

  Permite eliminar registros o tablas enteras de forma permanente cuando ya no son necesarias.

- **Restauración de Tablas Eliminadas:**

  Algunas bases de datos permiten recuperar registros o incluso tablas enteras eliminadas accidentalmente. Esto se logra a través de registros de transacciones o funciones de rollback.

Estas funcionalidades permiten que SQL sea una herramienta poderosa en la gestión de datos, asegurando eficiencia, seguridad e integridad en cualquier sistema de bases de datos relacional.

Conclusiones

El estudio de SQL es esencial para la gestión eficiente de bases de datos relacionales. A través de este informe, se ha explorado la creación, manipulación y optimización de datos mediante sentencias SQL, destacando la importancia de los índices, claves primarias y foráneas en la integridad y rendimiento de la base de datos. Además, se ha resaltado el papel de las consultas agregadas y la indexación en la optimización del acceso a la información. En conclusión, el dominio de SQL permite mejorar la administración y el rendimiento de bases de datos, contribuyendo a la eficiencia operativa en diversos sectores. La aplicación de estas prácticas en entornos reales facilita la toma de decisiones basada en datos confiables y organizados de manera óptima.

Bibliografía

- Oracle Live SQL Tutorial. (s.f.). *Introduction to SQL.* Recuperado de: https://livesql.oracle.com/next/worksheet?tutorial=introduction-to-sql-CDpng3
- Date, C. J. (2004). *An Introduction to Database Systems.* Pearson.
- Elmasri, R., & Navathe, S. B. (2015). *Fundamentals of Database Systems.* Pearson.
- O'Reilly Media. (s.f.). *SQL Performance Explained.* Recuperado de: https://use-the-index-luke.com/

```sql
CREATE TABLE empleados (
    empleado_id NUMBER PRIMARY KEY,
    nombre VARCHAR2(50) NOT NULL,
    apellido VARCHAR2(50) NOT NULL,
    fecha_contratacion DATE NOT NULL,
    salario NUMBER(10,2) NOT NULL,
    departamento_id NUMBER,
    cargo VARCHAR2(50),
    CONSTRAINT chk_salario CHECK (salario > 0)
);

-- Tabla de Departamentos
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
Tabla creada.
```

```sql
-- Tabla de Departamentos
CREATE TABLE departamentos (
    departamento_id NUMBER PRIMARY KEY,
    nombre_departamento VARCHAR2(50) NOT NULL,
    ubicacion VARCHAR2(100),
    director_id NUMBER
);
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
Tabla creada.

0,06 segundos
```

```sql
-- Secuencia para generar IDs automáticos para historial de salarios
CREATE SEQUENCE seq_historial_id
START WITH 1
INCREMENT BY 1
NOCACHE
NOCYCLE;

////////////////////////////
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
Secuencia creada.

0,03 segundos
```

```sql
-- Tabla para el historial de cambios de salario
CREATE TABLE historial_salarios (
    historial_id NUMBER PRIMARY KEY,
    empleado_id NUMBER,
    salario_anterior NUMBER(10,2),
    salario_nuevo NUMBER(10,2),
    fecha_cambio DATE,
    usuario VARCHAR2(50),
    CONSTRAINT fk_empleado_historial FOREIGN KEY (empleado_id) REFERENCES empleados(empleado_id)
);
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
Tabla creada.
```

0,06 segundos

Crear tablas en SQL es un proceso fundamental en el diseño de bases de datos. Para crear una tabla en SQL, usamos la instrucción CREATE TABLE, especificando el nombre de la tabla y las columnas con sus tipos de datos.

```sql
---4. Creación de Triggers (Creating Triggers)
CREATE OR REPLACE TRIGGER trg_actualiza_salario
AFTER UPDATE OF salario ON empleados
FOR EACH ROW
BEGIN
    INSERT INTO historial_salarios (
        historial_id,
        empleado_id,
        salario_anterior,
        salario_nuevo,
        fecha_cambio,
        usuario
    ) VALUES (
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
Disparador creado.
```

0,09 segundos

```sql
---10. Compresión de Datos (Compressing Data)
-- Crear una tabla de historial con compresión
CREATE TABLE historial_extendido (
    id NUMBER PRIMARY KEY,
    empleado_id NUMBER,
    evento VARCHAR2(100),
    descripcion VARCHAR2(4000),
    fecha TIMESTAMP,
    usuario VARCHAR2(50)
) COMPRESS FOR OLTP;
```

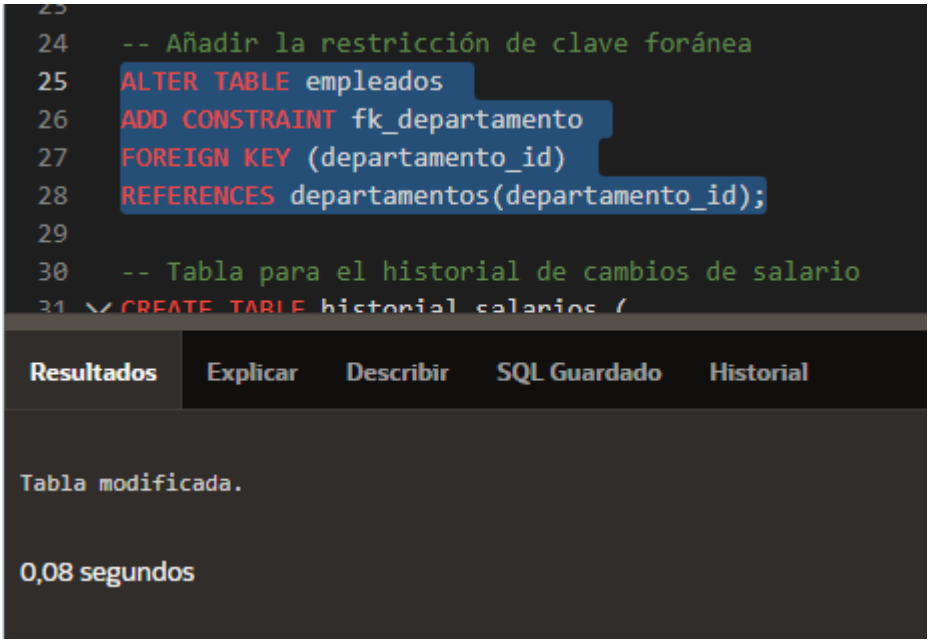
Resultados | Explicar | Describir | SQL Guardado | Historial

```
Tabla creada.
```

0,05 segundos

En SQL, para modificar la estructura de una tabla, usamos la instrucción ALTER TABLE. Esto te permite agregar, eliminar o modificar columnas, así como agregar restricciones a una tabla existente.

Aquí te explico los principales usos de ALTER TABLE:



en SQL es un tipo de procedimiento almacenado que se ejecuta automáticamente cuando ocurre un evento específico en una tabla o vista, como una inserción, actualización o eliminación de datos.

A continuación, te explico cómo crear un disparador (trigger) y algunos ejemplos comunes.

```sql
-- Insertar departamentos
INSERT INTO departamentos (departamento_id, nombre_departamento, ubicacion)
VALUES (seq_departamento_id.NEXTVAL, 'Recursos Humanos', 'Piso 1');

INSERT INTO departamentos (departamento_id, nombre_departamento, ubicacion)
VALUES (seq_departamento_id.NEXTVAL, 'Tecnología', 'Piso 2');

INSERT INTO departamentos (departamento_id, nombre_departamento, ubicacion)
VALUES (seq_departamento_id.NEXTVAL, 'Finanzas', 'Piso 3');

INSERT INTO departamentos (departamento_id, nombre_departamento, ubicacion)
VALUES (seq_departamento_id.NEXTVAL, 'Marketing', 'Piso 2');

-- Insertar empleados
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
1 fila(s) insertada(s).
```

Cuando se inserta un registro en una tabla en SQL, utilizamos la instrucción INSERT INTO. Este comando permite agregar uno o más registros a la tabla especificada.

```sql
-- Insertar empleados
INSERT INTO empleados (
    empleado_id,
    nombre,
    apellido,
    fecha_contratacion,
    salario,
    departamento_id,
    cargo
) VALUES (
    seq_empleado_id.NEXTVAL,
    'Juan',
    'Pérez',
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
1 fila(s) insertada(s).
```

```sql
200
201   -- Actualizar los directores de departamento
202   UPDATE departamentos SET director_id = 1 WHERE departamento_id = 1;
203   UPDATE departamentos SET director_id = 2 WHERE departamento_id = 2;
204   UPDATE departamentos SET director_id = 3 WHERE departamento_id = 3;
205   UPDATE departamentos SET director_id = 4 WHERE departamento_id = 4;
206
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
1 fila(s) actualizada(s).
```

0,01 segundos

La **modificación** y **actualización** de datos en una tabla de SQL se realizan mediante los comandos UPDATE y ALTER TABLE. A continuación, te explico cómo usar ambos:

```sql
207   -- Añadir clave foránea para director
208   ALTER TABLE departamentos
209   ADD CONSTRAINT fk_director
210   FOREIGN KEY (director_id)
211   REFERENCES empleados(empleado_id);
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
Tabla modificada.
```

0,06 segundos

```sql
---4. Indexación de Columnas (Indexing Columns)
-- Índice para búsqueda por apellido
CREATE INDEX idx_empleados_apellido ON empleados(apellido);

-- Índice para búsqueda por departamento
CREATE INDEX idx_empleados_departamento ON empleados(departamento_id);

-- Índice compuesto para búsquedas combinadas
CREATE INDEX idx_empleados_nombre_apellido ON empleados(nombre, apellido);

-- Índice para búsquedas de salario
CREATE INDEX idx_empleados_salario ON empleados(salario);
```

Resultados | Explicar | Describir | SQL Guardado | Historial

Índice creado.

0,04 segundos

```sql
---5. Consultas de Datos (Querying Data)
-- Consultar todos los empleados
SELECT * FROM empleados;
```

Resultados | Explicar | Describir | SQL Guardado | Historial

EMPLEADO_ID	NOMBRE	APELLIDO	FECHA_CONTRATACION	SALARIO	DEPARTAMENTO_ID	CARGO
2	María	González	22/03/19	4200	2	Desarrollador Senior
4	Laura	Martínez	05/09/20	3600	4	Especialista en Marketing Digital
1	Juan	Pérez	15/01/20	3500	1	Analista de RRHH
3	Carlos	Rodríguez	10/06/21	3800	3	Contador

4 filas devueltas en 0,02 segundos Descargar

```sql
-- Consultar empleados con salario mayor a 3700
SELECT empleado_id, nombre, apellido, salario
FROM empleados
WHERE salario > 3700;
```

Resultados | Explicar | Describir | SQL Guardado | Historial

EMPLEADO_ID	NOMBRE	APELLIDO	SALARIO
3	Carlos	Rodríguez	3800
2	María	González	4200

2 filas devueltas en 0,02 segundos Descargar

```sql
-- Consultar empleados por departamento
SELECT e.empleado_id, e.nombre, e.apellido, d.nombre_departamento
FROM empleados e
JOIN departamentos d ON e.departamento_id = d.departamento_id
ORDER BY d.nombre_departamento, e.apellido;
```

Resultados | Explicar | Describir | SQL Guardado | Historial

EMPLEADO_ID	NOMBRE	APELLIDO	NOMBRE_DEPARTAMENTO
3	Carlos	Rodríguez	Finanzas
4	Laura	Martínez	Marketing
1	Juan	Pérez	Recursos Humanos
2	María	González	Tecnología

4 filas devueltas en 0,06 segundos Descargar

```sql
------6. Adición de Columnas (Adding Columns)
-- Añadir columna para correo electrónico
ALTER TABLE empleados
ADD correo_electronico VARCHAR2(100);

-- Añadir columna para número de teléfono
ALTER TABLE empleados
ADD telefono VARCHAR2(20);

-- Añadir columna para dirección
ALTER TABLE empleados
ADD direccion VARCHAR2(200);
```

Resultados | Explicar | Describir | SQL Guardado | Historial

Tabla modificada.

```sql
-- Actualizar datos para las nuevas columnas
UPDATE empleados
SET correo_electronico = LOWER(nombre || '.' || apellido || '@empresa.com')
WHERE empleado_id > 0;
//////////
/////////
```

Resultados | Explicar | Describir | SQL Guardado | Historial

4 fila(s) actualizada(s).

```sql
-- Consultar todas las tablas del esquema actual
SELECT table_name
FROM user_tables
ORDER BY table_name;
```

Resultados | Explicar | Describir | SQL Guardado | Historial

TABLE_NAME
DEPARTAMENTOS
EMPLEADOS
HISTORIAL_SALARIOS

3 filas devueltas en 0,15 segundos Descargar

```sql
-- Consultar triggers asociados a la tabla
SELECT trigger_name, trigger_type, triggering_event
FROM user_triggers
WHERE table_name = 'EMPLEADOS';
//////////////////
//////////////////
```

Resultados | Explicar | Describir | SQL Guardado | Historial

TRIGGER_NAME	TRIGGER_TYPE	TRIGGERING_EVENT
TRG_VALIDAR_EMPLEADO	BEFORE EACH ROW	INSERT OR UPDATE

1 filas devueltas en 0,11 segundos Descargar

```sql
---8. Actualización de Datos (Updating Data)
-- Actualizar el salario de un empleado (activará el trigger)
UPDATE empleados
SET salario = 3800
WHERE empleado_id = 1;

-- Actualizar el departamento de un empleado
UPDATE empleados
SET departamento_id = 2
WHERE empleado_id = 3;

-- Actualizar múltiples columnas
UPDATE empleados
SET telefono = '555-123-4567',
```

Resultados | Explicar | Describir | SQL Guardado | Historial

```
1 fila(s) actualizada(s).
```

```sql
---9. Consultas Agregadas (Aggregate Queries)
-- Calcular el salario promedio de todos los empleados
SELECT AVG(salario) AS salario_promedio
FROM empleados;

-- Calcular el salario promedio por departamento
SELECT d.nombre_departamento, AVG(e.salario) AS salario_promedio
FROM empleados e
JOIN departamentos d ON e.departamento_id = d.departamento_id
GROUP BY d.nombre_departamento;

-- Contar empleados por departamento
```

Resultados | Explicar | Describir | SQL Guardado | Historial

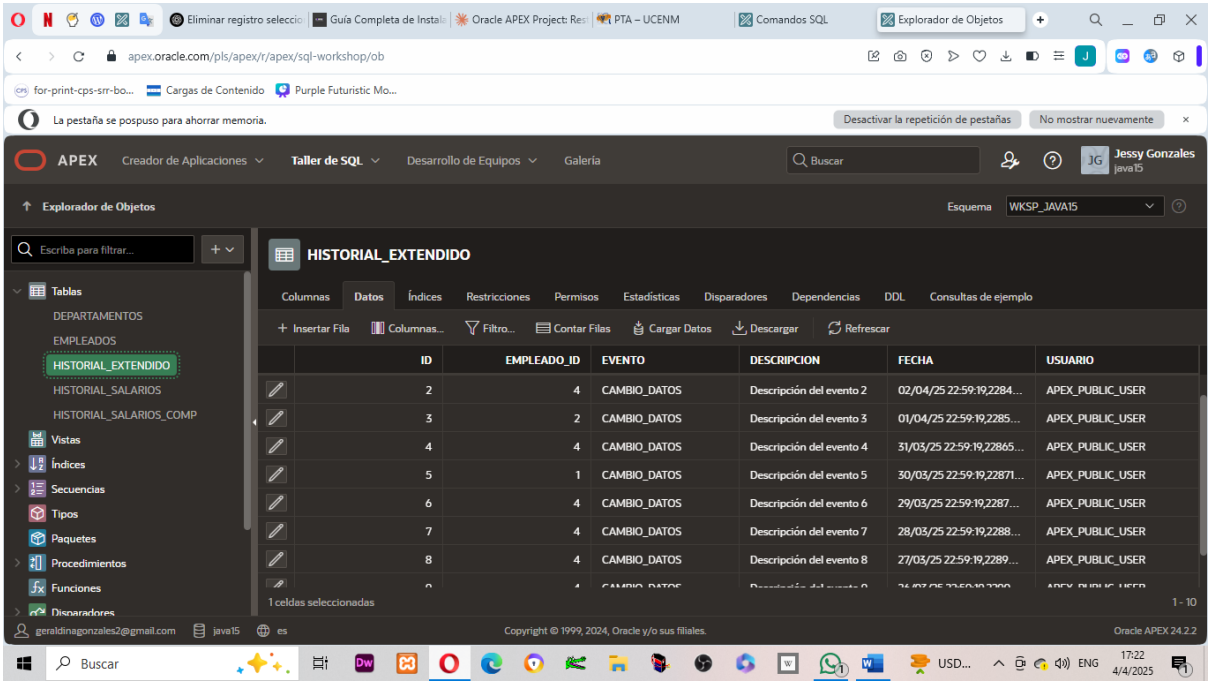
SALARIO_PROMEDIO
3950

```sql
-- Contar empleados por departamento
SELECT d.nombre_departamento, COUNT(e.empleado_id) AS total_empleados
FROM empleados e
JOIN departamentos d ON e.departamento_id = d.departamento_id
GROUP BY d.nombre_departamento;
```

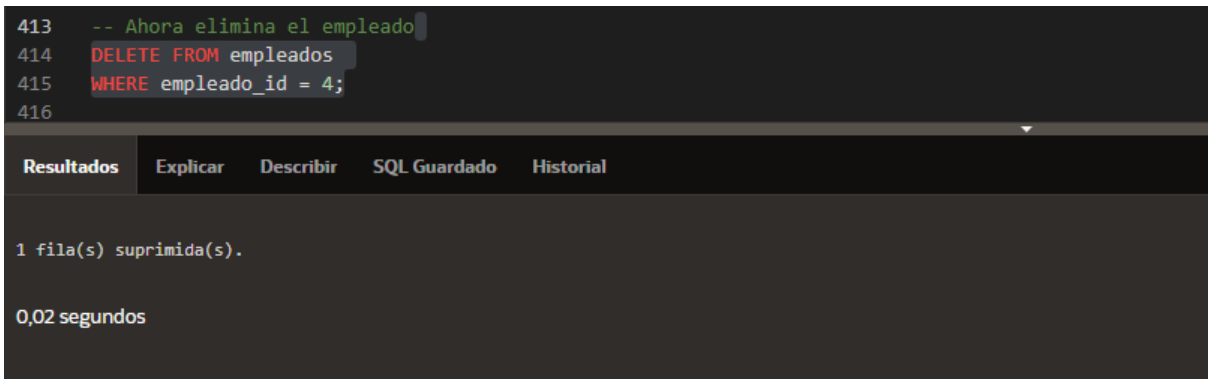
Resultados | Explicar | Describir | SQL Guardado | Historial

NOMBRE_DEPARTAMENTO	TOTAL_EMPLEADOS
Marketing	1
Tecnología	2
Recursos Humanos	1

3 filas devueltas en 0,02 segundos Descargar

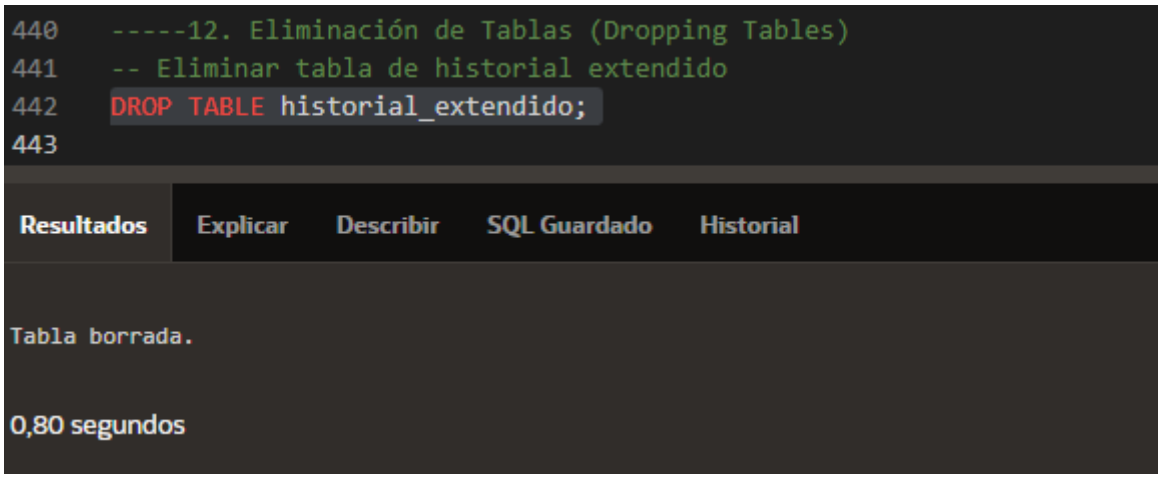

APEX
Taller de SQL
Explorador de Objetos
HISTORIAL_EXTENDIDO
Columnas
Datos
Índices
Restricciones
Permisos
Estadísticas
Disparadores
Dependencias
DDL
Consultas de ejemplo
ID
EMPLEADO_ID
EVENTO
DESCRIPCION
FECHA
USUARIO
2 4 CAMBIO_DATOS Descripción del evento 2 02/04/25 22:59:19,2284... APEX_PUBLIC_USER
3 2 CAMBIO_DATOS Descripción del evento 3 01/04/25 22:59:19,2285... APEX_PUBLIC_USER
4 4 CAMBIO_DATOS Descripción del evento 4 31/03/25 22:59:19,22865... APEX_PUBLIC_USER
5 1 CAMBIO_DATOS Descripción del evento 5 30/03/25 22:59:19,22871... APEX_PUBLIC_USER
6 4 CAMBIO_DATOS Descripción del evento 6 29/03/25 22:59:19,2287... APEX_PUBLIC_USER
7 4 CAMBIO_DATOS Descripción del evento 7 28/03/25 22:59:19,2288... APEX_PUBLIC_USER
8 4 CAMBIO_DATOS Descripción del evento 8 27/03/25 22:59:19,2289... APEX_PUBLIC_USER

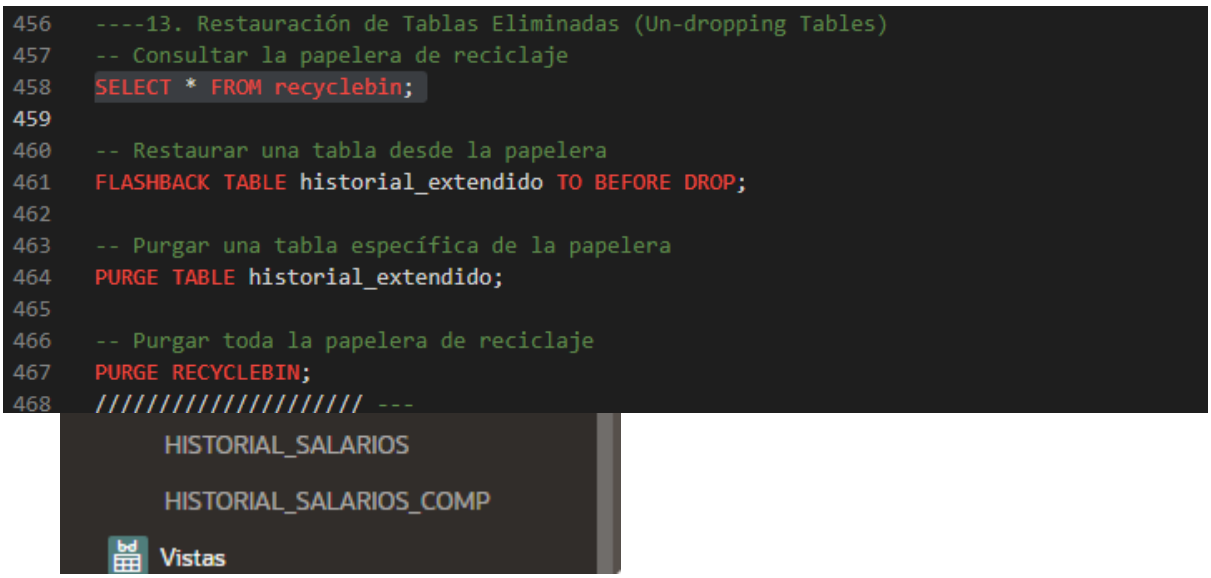

EMPLEADOS
Columnas
Datos
Índices
Restricciones
Permisos
Estadísticas
Disparadores
Dependencias
DDL
Consultas de ejemplo
Insertar Fila
Columnas...
Filtro...
Contar Filas
Cargar Datos
Descargar
Refrescar
EMPLEADO_ID NOMBRE APELLIDO FECHA_CONTRATACIOI SALARIO DEPARTAMENTO_ID CARGO
2 María González 22/03/19 4410 2 Desarrollador Senior
4 Laura Martínez 05/09/20 3600 4 Especialista en Marke...
1 Juan Pérez 15/01/20 3800 1 Analista de RRHH
3 Carlos Rodríguez 10/06/21 3990 2 Contador


413 -- Ahora elimina el empleado
414 DELETE FROM empleados
415 WHERE empleado_id = 4;
416
Resultados
Explicar
Describir
SQL Guardado
Historial
1 fila(s) suprimida(s).
0,02 segundos


EMPLEADOS
Columnas
Datos
Índices
Restricciones
Permisos
Estadísticas
Disparadores
Dependencias
DDL
Consultas de ejemplo
Insertar Fila
Columnas...
Filtro...
Contar Filas
Cargar Datos
Descargar
Refrescar
EMPLEADO_ID NOMBRE APELLIDO FECHA_CONTRATA SALARIO DEPARTAMENTO_I CARGO CORREO_ELECTRO TELEFONO
2 María González 22/03/19 4410 2 Desarrollador ... maría.gonzález... 555-123-45
1 Juan Pérez 15/01/20 3800 1 Analista de RR... juan.pérez@e...
3 Carlos Rodríguez 10/06/21 3990 2 Contador carlos.rodrígue...

Parece que mencionas "tablas borradas". ¿Te refieres a tablas de bases de datos que fueron eliminadas o estás hablando de otro tipo de tablas? Si es sobre bases de datos, ¿en qué contexto te gustaría recuperar o trabajar con ellas?





Parece que estás trabajando con una base de datos que tiene tablas relacionadas con "departamentos", "historial" y "empleados". Si estas tablas han sido borradas y deseas recuperarlas o restaurarlas, depende de varios factores, como si tienes un respaldo o si trabajas con un sistema de gestión de bases de datos que tenga características de recuperación de datos.