



**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 Orteiz 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

## INDICE

Introducción .....	3
Planteamiento del Problema .....	4
Objetivos .....	5
Objetivo General .....	5
Objetivos Específicos .....	5
El lenguaje SQL .....	6
• Creación de Tablas: .....	6
• Creación de Triggers: .....	6
• Inserción de Datos: .....	6
• Indexación de Columnas: .....	6
• Consultas de Datos: .....	6
• Adición de Columnas: .....	7
• Consulta del Diccionario de Datos de Oracle: .....	7
• Actualización de Datos: .....	7
• Consultas Agregadas: .....	7
• Compresión de Datos: .....	7
• Eliminación de Datos y Tablas: .....	7
• Restauración de Tablas Eliminadas: .....	8
Conclusiones .....	9
Bibliografía .....	10
Anexos .....	11
Link presentación .....	21

## **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/>

## Anexos

```
5 CREATE TABLE empleados (  
6     empleado_id NUMBER PRIMARY KEY,  
7     nombre VARCHAR2(50) NOT NULL,  
8     apellido VARCHAR2(50) NOT NULL,  
9     fecha_contratacion DATE NOT NULL,  
10    salario NUMBER(10,2) NOT NULL,  
11    departamento_id NUMBER,  
12    cargo VARCHAR2(50),  
13    CONSTRAINT chk_salario CHECK (salario > 0)  
14 );  
15  
16 -- Tabla de Departamentos
```

Resultados	Explicar	Describir	SQL Guardado	Historial
Tabla creada.				

```
16 -- Tabla de Departamentos  
17 CREATE TABLE departamentos (  
18     departamento_id NUMBER PRIMARY KEY,  
19     nombre_departamento VARCHAR2(50) NOT NULL,  
20     ubicacion VARCHAR2(100),  
21     director_id NUMBER  
22 );  
23
```

Resultados	Explicar	Describir	SQL Guardado	Historial
Tabla creada.				
0,06 segundos				

```
63 -- Secuencia para generar IDs automáticos para historial de salarios  
64 CREATE SEQUENCE seq_historial_id  
65 START WITH 1  
66 INCREMENT BY 1  
67 NOCACHE  
68 NOCYCLE;  
69  
70 //////////////////////////////////////////////////
```

Resultados	Explicar	Describir	SQL Guardado	Historial
Secuencia creada.				
0,03 segundos				

```
29
30 -- Tabla para el historial de cambios de salario
31 CREATE TABLE historial_salarios (
32     historial_id NUMBER PRIMARY KEY,
33     empleado_id NUMBER,
34     salario_anterior NUMBER(10,2),
35     salario_nuevo NUMBER(10,2),
36     fecha_cambio DATE,
37     usuario VARCHAR2(50),
38     CONSTRAINT fk_empleado_historial FOREIGN KEY (empleado_id) REFERENCES empleados(empleado_id)
39 );
40
```

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.

```
65 ---9. Creación de Triggers (Creating Triggers)
66 CREATE OR REPLACE TRIGGER trg_actualiza_salario
67 AFTER UPDATE OF salario ON empleados
68 FOR EACH ROW
69 BEGIN
70     INSERT INTO historial_salarios (
71         historial_id,
72         empleado_id,
73         salario_anterior,
74         salario_nuevo,
75         fecha_cambio,
76         usuario
77     ) VALUES (

```

Resultados   Explicar   Describir   SQL Guardado   Historial

Disparador creado.

0,0

```
356 ---10. Compresión de Datos (Compressing Data)
357 -- Crear una tabla de historial con compresión
358 CREATE TABLE historial_extendido (
359     id NUMBER PRIMARY KEY,
360     empleado_id NUMBER,
361     evento VARCHAR2(100),
362     descripcion VARCHAR2(4000),
363     fecha TIMESTAMP,
364     usuario VARCHAR2(50)
365 ) COMPRESS FOR OLTP;
366
367 Insertar algunos datos de ejemplo en la tabla comprimida
```

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:

```
23
24 -- Añadir la restricción de clave foránea
25 ALTER TABLE empleados
26 ADD CONSTRAINT fk_departamento
27 FOREIGN KEY (departamento_id)
28 REFERENCES departamentos(departamento_id);
29
30 -- Tabla para el historial de cambios de salario
31 CREATE TABLE historial_salarios (
```

Resultados	Explicar	Describir	SQL Guardado	Historial
Tabla modificada.				
0,08 segundos				

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.

```
106 -- Insertar departamentos
107 INSERT INTO departamentos (departamento_id, nombre_departamento, ubicacion)
108 VALUES (seq_departamento_id.NEXTVAL, 'Recursos Humanos', 'Piso 1');
109
110 INSERT INTO departamentos (departamento_id, nombre_departamento, ubicacion)
111 VALUES (seq_departamento_id.NEXTVAL, 'Tecnología', 'Piso 2');
112
113 INSERT INTO departamentos (departamento_id, nombre_departamento, ubicacion)
114 VALUES (seq_departamento_id.NEXTVAL, 'Finanzas', 'Piso 3');
115
116 INSERT INTO departamentos (departamento_id, nombre_departamento, ubicacion)
117 VALUES (seq_departamento_id.NEXTVAL, 'Marketing', 'Piso 2');
118
119 -- 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.

```
127 -- Insertar empleados
128 ✓ INSERT INTO empleados (
129     empleado_id,
130     nombre,
131     apellido,
132     fecha_contratacion,
133     salario,
134     departamento_id,
135     cargo
136 ✓ ) VALUES (
137     seq_empleado_id.NEXTVAL,
138     'Juan',
139     'Pérez',
140     '2023-01-01',
141     5000,
142     1,
143     'Analista de Datos'
144 )
```

Resultados	Explicar	Describir	SQL Guardado	Historial
1 fila(s) insertada(s).				

```

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:

```

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

```

214 ---4. Indexación de Columnas (Indexing Columns)
215 -- Índice para búsqueda por apellido
216 CREATE INDEX idx_empleados_apellido ON empleados(apellido);
217
218 -- Índice para búsqueda por departamento
219 CREATE INDEX idx_empleados_departamento ON empleados(departamento_id);
220
221 -- Índice compuesto para búsquedas combinadas
222 CREATE INDEX idx_empleados_nombre_apellido ON empleados(nombre, apellido);
223
224 -- Índice para búsquedas de salario
225 CREATE INDEX idx_empleados_salario ON empleados(salario);

```

**Resultados**

Explicar

Describir

SQL Guardado

Historial

Índice creado.

0,04 segundos

```

228 ---5. Consultas de Datos (Querying Data)
229 -- Consultar todos los empleados
230 SELECT * FROM empleados;
231

```

**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](#)

```

232 -- Consultar empleados con salario mayor a 3700
233 SELECT empleado_id, nombre, apellido, salario
234 FROM empleados
235 WHERE salario > 3700;
236

```

**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](#)

```

237 -- Consultar empleados por departamento
238 SELECT e.empleado_id, e.nombre, e.apellido, d.nombre_departamento
239 FROM empleados e
240 JOIN departamentos d ON e.departamento_id = d.departamento_id
241 ORDER BY d.nombre_departamento, e.apellido;
242
243 -- Consultar información del empleado y su director

```

**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](#)



```

254  -----6. Adición de Columnas (Adding Columns)
255  -- Añadir columna para correo electrónico
256  ALTER TABLE empleados
257  ADD correo_electronico VARCHAR2(100);
258
259  -- Añadir columna para número de teléfono
260  ALTER TABLE empleados
261  ADD telefono VARCHAR2(20);
262
263  -- Añadir columna para dirección
264  ALTER TABLE empleados
265  ADD direccion VARCHAR2(200);
266

```

**Resultados**

Explicar

Describir

SQL Guardado

Historial

Tabla modificada.

```

266
267  -- Actualizar datos para las nuevas columnas
268  UPDATE empleados
269  SET correo_electronico = LOWER(nombre || '.' || apellido || '@empresa.com')
270  WHERE empleado_id > 0;
271  //////////
272  //////////

```

**Resultados**

Explicar

Describir

SQL Guardado

Historial

4 fila(s) actualizada(s).

```

274  -- Consultar todas las tablas del esquema actual
275  SELECT table_name
276  FROM user_tables
277  ORDER BY table_name;
278

```

**Resultados**

Explicar

Describir

SQL Guardado

Historial

TABLE_NAME
DEPARTAMENTOS
EMPLEADOS
HISTORIAL_SALARIOS

3 filas devueltas en 0,15 segundos

[Descargar](#)

```

295 -- Consultar triggers asociados a la tabla
296 SELECT trigger_name, trigger_type, triggering_event
297 FROM user.triggers
298 WHERE table_name = 'EMPLEADOS';
299 //////////////////////////////////////////////////
300 //////////////////////////////////////////////////

```

TRIGGER_NAME	TRIGGER_TYPE	TRIGGERING_EVENT
TRG_VALIDAR_EMPLEADO	BEFORE EACH ROW	INSERT OR UPDATE

1 filas devueltas en 0,11 segundos [Descargar](#)

```

301 ---8. Actualización de Datos (Updating Data)
302 -- Actualizar el salario de un empleado (activará el trigger)
303 UPDATE empleados
304 SET salario = 3800
305 WHERE empleado_id = 1;
306
307 -- Actualizar el departamento de un empleado
308 UPDATE empleados
309 SET departamento_id = 2
310 WHERE empleado_id = 3;
311
312 -- Actualizar múltiples columnas
313 UPDATE empleados
314 SET telefono = '555-123-4567',

```

Resultados	Explicar	Describir	SQL Guardado	Historial
------------	----------	-----------	--------------	-----------

1 fila(s) actualizada(s).

```

324 ---9. Consultas Agregadas (Aggregate Queries)
325 -- Calcular el salario promedio de todos los empleados
326 SELECT AVG(salario) AS salario_promedio
327 FROM empleados;
328
329 -- Calcular el salario promedio por departamento
330 SELECT d.nombre_departamento, AVG(e.salario) AS salario_promedio
331 FROM empleados e
332 JOIN departamentos d ON e.departamento_id = d.departamento_id
333 GROUP BY d.nombre_departamento;
334
335 -- Contar empleados por departamento

```

Resultados	Explicar	Describir	SQL Guardado	Historial
------------	----------	-----------	--------------	-----------

SALARIO_PROMEDIO
3950

```

335 -- Contar empleados por departamento
336 SELECT d.nombre_departamento, COUNT(e.empleado_id) AS total_empleados
337 FROM empleados e
338 JOIN departamentos d ON e.departamento_id = d.departamento_id
339 GROUP BY d.nombre_departamento;
340

```

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](#)

Oracle APEX 24.2.2 interface showing the 'HISTORIAL\_EXTENDIDO' table. The table has columns: ID, EMPLEADO\_ID, EVENTO, DESCRIPCION, FECHA, and USUARIO. The 'Datos' tab is selected, showing a list of events.

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

Oracle APEX 24.2.2 interface showing the 'EMPLEADOS' table. The table has columns: EMPLEADO\_ID, NOMBRE, APELLIDO, FECHA\_CONTRATACION, SALARIO, DEPARTAMENTO\_ID, and CARGO. The 'Datos' tab is selected, showing a list of employees.

EMPLEADO_ID	NOMBRE	APELLIDO	FECHA_CONTRATACION	SALARIO	DEPARTAMENTO_ID	CARGO
2	Maria	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

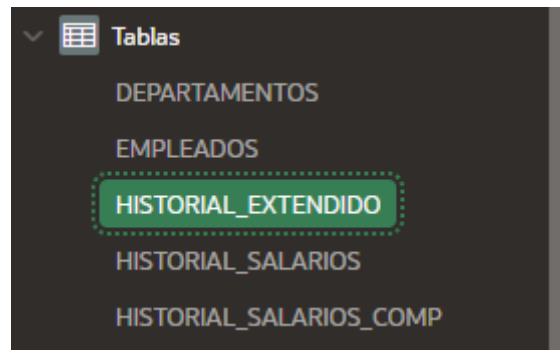
Oracle APEX 24.2.2 interface showing the execution of a SQL query to delete an employee. The query is: `DELETE FROM empleados WHERE empleado_id = 4;`. The results show 1 row(s) deleted.

```
413 -- Ahora elimina el empleado
414 DELETE FROM empleados
415 WHERE empleado_id = 4;
416
```

Resultados: 1 fila(s) suprimida(s). 0,02 segundos.

EMPLEADOS table view:

EMPLEADO_ID	NOMBRE	APELLIDO	FECHA_CONTRATACION	SALARIO	DEPARTAMENTO_ID	CARGO	CORREO ELECTRONICO	TELEFONO
2	Maria	González	22/03/19	4410	2	Desarrollador ...	maria.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...	

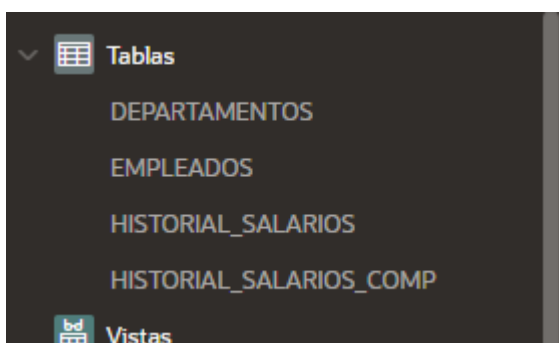


Parece que mencionas "tablas borradas". ¿Te refieres a tablas de bases de datos que fueron eliminadas o estás hablando de otro tipo de tablas? Si es sobre bases de datos, ¿en qué contexto te gustaría recuperar o trabajar con ellas?

```
440 -----12. Eliminación de Tablas (Dropping Tables)
441 -- Eliminar tabla de historial extendido
442 DROP TABLE historial_extendido;
443
```

Resultados	Explicar	Describir	SQL Guardado	Historial
Tabla borrada.				
0,80 segundos				

```
456 ----13. Restauración de Tablas Eliminadas (Un-dropping Tables)
457 -- Consultar la papelera de reciclaje
458 SELECT * FROM recyclebin;
459
460 -- Restaurar una tabla desde la papelera
461 FLASHBACK TABLE historial_extendido TO BEFORE DROP;
462
463 -- Purgar una tabla específica de la papelera
464 PURGE TABLE historial_extendido;
465
466 -- Purgar toda la papelera de reciclaje
467 PURGE RECYCLEBIN;
468 ////////////////////////////////// --
```



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.

## **Link presentación**

[https://www.canva.com/design/DAGjsUwMPuU/qZSLbGDZ0K0gALAyvy5xDg/edit?utm\\_content=DAGjsUwMPuU&utm\\_campaign=designshare&utm\\_medium=link2&utm\\_source=sharebutton](https://www.canva.com/design/DAGjsUwMPuU/qZSLbGDZ0K0gALAyvy5xDg/edit?utm_content=DAGjsUwMPuU&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton)