

UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS



LABORATORIO 02:

“SQL DML en Oracle Database.”

ASIGNATURA: Administración de Base de Datos.

DOCENTE: Chávez Soto, Jorge Luis.

ALUMNA: Angeles Seclen, Juana Agripina – 23200004.

Semestre 2025-II

Lima, Perú

ÍNDICE

SQL DML en Oracle Database.....	3
Construcción de la Aplicación.....	3
Enunciado.....	3
Tareas Para Realizar.....	3
Tablespaces de datos y temporales.....	6
Actividad 01. Creación de objetos de la BD.....	7
Actividad 02. Restricción de sexo en EMPLEADOS.....	11
Actividad 03. Restricciones de unicidad en nombres en DEPARTAMENTOS y TRABAJOS.....	11
Actividad 04. Evitar solapamientos de salarios y trabajos.....	12
Actividad 05. Declaración de claves primarias.....	14
Actividad 06. Agregar teléfono y celular en EMPLEADOS.....	14
Actividad 07. Inserción de filas de ejemplo.....	14
Actividad 08. Validación de supervisor inexistente.....	19
Actividad 09. Borrar una universidad y su efecto en ESTUDIOS.....	19
Actividad 10. Forzar COD_POSTAL NOT NULL cuando CIUDAD esté rellena....	20
Actividad 11. Añadir VALORACION con valor por defecto 5 (1–10).....	21
Actividad 12. Eliminar la restricción NOT NULL en EMPLEADOS.NOMBRE.....	21
Actividad 13. Modificar DIRECC1 a VARCHAR2(40).....	22
Actividad 14. Conversión de FECHA_NAC a cadena.....	22
Actividad 15. Cambiar la clave primaria de EMPLEADOS al NOMBRE y los dos APELLIDOS.....	23
Actividad 16. Crear tabla INFORMACION UNIVERSITARIA.....	25
Actividad 17. Crear vista NOMBRE_EMPLEADOS para empleados de Málaga....	26
Actividad 18. Crear vista INFORMACION_EMPLEADOS con edad.....	27
Actividad 19. Crear vista INFORMACION_ACTUAL con salario.....	27
Actividad 20. Borrar todas las tablas.....	28

SQL DML en Oracle Database.

Construcción de la Aplicación

Enunciado

Se desea tener una base de datos que almacene la información sobre los empleados de una empresa, los departamentos en los que trabajan y los estudios de que disponen. Guardaremos el historial laboral y salarial de todos los empleados. Para ello contamos con las siguientes tablas:

EMPLEADOS		DEPARTAMENTOS	
Column Name	Data Type	Column Name	Data Type
-----	-----	-----	-----
DNI	NUMBER(8)	DPTO_COD	NUMBER(5)
NOMBRE	VARCHAR(10)	NOMBRE_DPTO	VARCHAR(30)
APELLIDO1	VARCHAR(15)	DPTO_PADRE	NUMBER(5)
APELLIDO2	VARCHAR(15)	PRESUPUESTO	NUMBER
DIRECC1	VARCHAR(25)	PRES_ACTUAL	NUMBER
DIRECC2	VARCHAR(20)		
CIUDAD	VARCHAR(20)	ESTUDIOS	
PROVINCIA	VARCHAR(20)	Column Name	Data Type
COD_POSTAL	VARCHAR(5)	-----	-----
SEXO	VARCHAR(1)	EMPLEADO_DNI	NUMBER(8)
FECHA_NAC	DATE	UNIVERSIDAD	NUMBER(5)
		AÑO	NUMBER
		GRADO	VARCHAR(3)
		ESPECIALIDAD	VARCHAR(20)
HISTORIAL_LABORAL		UNIVERSIDADES	
Column Name	Data Type	Column Name	Data Type
-----	-----	-----	-----
EMPLEADO_DNI	NUMBER(8)	UNIV_COD	NUMBER(5)
TRABAJO_COD	NUMBER(5)	NOMBRE_UNIV	VARCHAR(25)
FECHA_INICIO	DATE	CIUDAD	VARCHAR(20)
FECHA_FIN	DATE	MUNICIPIO	VARCHAR(2)
DPTO_COD	NUMBER(5)	COD_POSTAL	VARCHAR(5)
SUPERVISOR_DNI	NUMBER(8)		
HISTORIAL_SALARIAL		TRABAJOS	
Column Name	Data Type	Column Name	Data Type
-----	-----	-----	-----
EMPLEADO_DNI	NUMBER(8)	TRABAJO_COD	NUMBER(5)
SALARIO	NUMBER	NOMBRE TRAB	VARCHAR(20)
FECHA_COMIENZO	DATE	SALARIO_MIN	NUMBER(2)
FECHA_FIN	DATE	SALARIO_MAX	NUMBER(2)

Tareas Para Realizar

Estas son las entregables de la tarea a realizar:

- **Actividad 01:** Los siguientes atributos son obligatorios:

- NOMBRE (en todas las tablas)
- APELLIDO1 en EMPLEADOS
- PRESUPUESTO en DEPARTAMENTOS
- SALARIO en HISTORIAL_SALARIAL
- SALARIO_MIN y SALARIO_MAX en TRABAJOS.

Al crear las tablas correspondientes especificar la opción NOT NULL.

Si la tabla ya estuviese creada:

ALTER TABLE nombre_tabla ADD nombre_campo TIPO NOT NULL;

- **Actividad 02:** El atributo SEXO en EMPLEADOS sólo puede tomar los valores H para hombre y M para mujer.
- **Actividad 03:** Dos DEPARTAMENTOS no se llaman igual. Dos TRABAJOS tampoco.
- **Actividad 04:** Cada empleado tiene un solo salario en cada momento. También, cada empleado tendrá asignado un solo trabajo en cada momento.
- **Actividad 05:** Se ha de mantener la regla de integridad de referencia y pensar una clave primaria para cada tabla.
- **Actividad 06:** Agregue a la tabla empleados los campos de teléfono y celular para tener como ubicar rápidamente al empleado.
- **Actividad 07:** Inserte las siguientes filas (las columnas que no aparecen se considerarán nulas).

Empleados				
NOMBRE	APELLIDO1	APELLIDO2	DNI	SEXO
Sergio	Palma	Entrena	111222	H
Lucia	Ortega	Plus	222333	M

Historial_Laboral					
EM- PLEADO_DNI	TRAB_C OD	FE- CHA_INICIO	FE- CHA_FIN	DPTO_C OD	SUPERVI- SOR_DNI
111222		16/06/96		222333	

- **Actividad 08:** ¿Qué ocurre si se modifica esta última fila de historial_laboral asignándole al empleado 111222 un supervisor que no existe en la tabla de empleados?

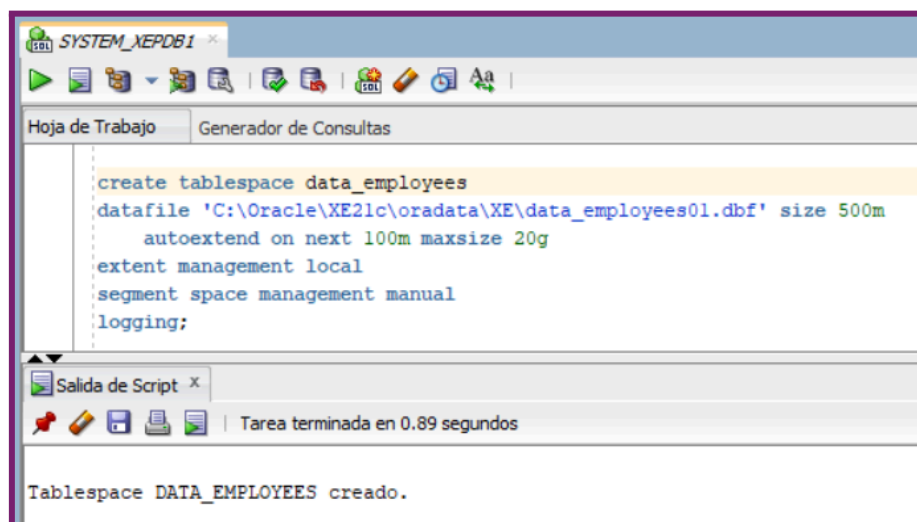
- **Actividad 09:** Borre una universidad de la tabla de UNIVERSIDADES ¿Qué le sucede a la restricción de clave ajena de la tabla ESTUDIOS? Altere la definición de la tabla para que se mantenga la restricción, aunque se borre una universidad.
- **Actividad 10:** Añada una restricción que obligue a que las personas que hayan introducido la CIUDAD deban tener el campo COD_POSTAL a NOT NULL. ¿Qué ocurre con las filas ya introducidas?
- **Actividad 11:** Añada un nuevo atributo VALORACIÓN en la tabla de EMPLEADOS que indique de 1 a 10 la valoración que obtuvo el empleado en su entrevista de trabajo al iniciar su andadura en la empresa. Ponga el valor por defecto 5 para ese campo.
- **Actividad 12:** Elimine la restricción de que el atributo NOMBRE de la tabla EMPLEADOS no puede ser nulo.
- **Actividad 13:** Modificar el tipo de datos de DIREC1 de la tabla EMPLEADOS a cadena de caracteres de 40 como máximo.
- **Actividad 14:** ¿Podría modificar el tipo de datos del atributo FECHA_NAC de la tabla EMPLEADOS Y convertirla a tipo cadena?
- **Actividad 15:** Cambiar la clave primaria de EMPLEADOS al NOMBRE y los dos APELLIDOS.
- **Actividad 16:** Crear una nueva tabla llamada INFORMACION UNIVERSITARIA que tenga el NOMBRE y los dos APELLIDOS (en un solo atributo) de todos los EMPLEADOS junto con la UNIVERSIDAD donde estudiaron. Cárguela con los datos correspondientes.
- **Actividad 17:** Crear una vista llamada NOMBRE_EMPLEADOS con el NOMBRE y los dos APELLIDOS (en un solo atributo) de todos los EMPLEADOS que son de Málaga.
- **Actividad 18:** Crear otra vista llamada INFORMACION_EMPLEADOS con el NOMBRE y los dos APELLIDOS (en un solo atributo) y EDAD (no fecha de nacimiento) de todos los EMPLEADOS.
- **Actividad 19:** Crear otra vista sobre la anterior llamada INFORMACION_ACTUAL que dispone de toda la información de INFORMACION_EMPLEADOS junto con el SALARIO que está cobrando en este momento.
- **Actividad 20:** Borrar todas las tablas. ¿Hay que tener en cuenta las claves ajenas a la hora de borrar las tablas?

Tablespaces de datos y temporales

- **Tablespace de datos (data_employees):**

Permite almacenar los datos de la base de datos con suficiente espacio para inserciones masivas y crecimiento académico.

```
create tablespace data_employees  
  
datafile  
'C:\Oracle\XE21c\oradata\XE\data_employees01.dbf' size  
500m  
  
autoextend on next 100m maxsize 20g  
  
extent management local  
  
segment space management manual  
  
logging;
```

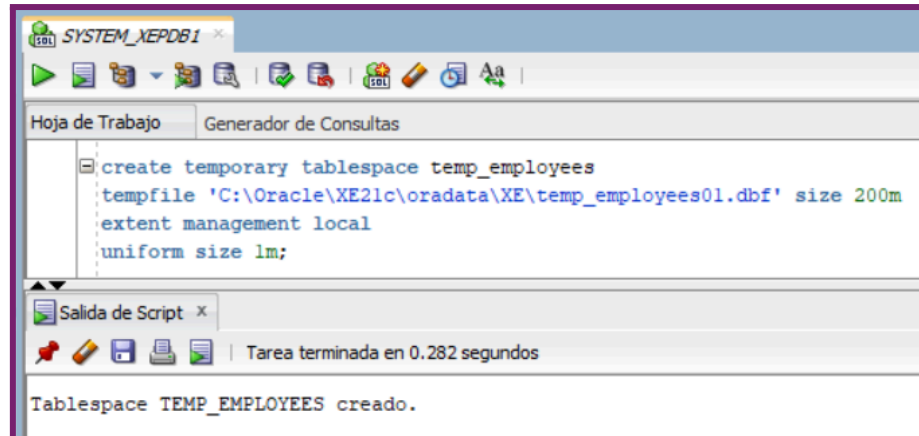


- **Tablespace temporal (temp_employees):**

Almacena datos temporales utilizados durante las operaciones de la base de datos.

```
create temporary tablespace temp_employees
```

```
tempfile  
'C:\Oracle\XE21c\oradata\XE\temp_employees01.dbf' size  
200m  
  
extent management local  
  
uniform size 1m;
```



Actividad 01. Creación de objetos de la BD.

Se crean las tablas básicas de la base de datos: empleados, departamentos, trabajos, universidades, historial laboral, historial salarial y estudios.

- **Tabla Empleados:**

```
create table empleados (  
  
    DNI number(8) primary key,  
  
    NOMBRE varchar2(10) not null,  
  
    APELLIDO1 varchar2(15) not null,  
  
    APELLIDO2 varchar2(15),  
  
    DIRECC1 varchar2(25),  
  
    DIRECC2 varchar2(20),
```

```
        CIUDAD varchar2(20),

        PROVINCIA varchar2(20),

        COD_POSTAL varchar2(5),

        SEXO varchar2(1) check (SEXO in ('H', 'M')),

        FECHA_NAC date

    ) tablespace data_employees;
```

- **Tabla Departamentos:**

```
create table departamentos (

    DPTO_COD number(5) primary key,

    NOMBRE_DPTO varchar2(30) not null,

    DPTO_PADRE number(5),

    PRESUPUESTO number not null,

    PRES_ACTUAL number,

    constraint uq_departamentos_nombre unique
(NOMBRE_DPTO),

    constraint fk_departamentos_padre foreign key
(DPTO_PADRE) references departamentos(DPTO_COD)

) tablespace data_employees;
```

- **Tabla Trabajos:**

```
create table trabajos (

    TRABAJO_COD number(5) primary key,

    NOMBRE_TRAB varchar2(20) not null,
```



```
SALARIO_MIN number not null,  
  
SALARIO_MAX number not null,  
  
constraint uq_trabajos_nombre unique (NOMBRE_TRAB)  
  
) tablespace data_employees;
```

- **Tabla Universidades:**

```
create table universidades (  
  
    UNIV_COD number(5) primary key,  
  
    NOMBRE_UNIV varchar2(25) not null,  
  
    CIUDAD varchar2(20),  
  
    MUNICIPIO varchar2(20),  
  
    COD_POSTAL varchar2(5)  
  
) tablespace data_employees;
```

- **Tabla Historial_Laboral:**

```
create table historial_laboral (  
  
    EMPLEADO_DNI number(8),  
  
    TRABAJO_COD number(5),  
  
    FECHA_INICIO date,  
  
    FECHA_FIN date,  
  
    DPTO_COD number(5),  
  
    SUPERVISOR_DNI number(8),  
  
    constraint pk_historial_laboral primary key
```

```

(EMPLEADO_DNI, TRABAJO_COD, FECHA_INICIO),

        constraint  fk_histlab_empleado  foreign  key
(EMPLEADO_DNI) references empleados(DNI),

        constraint  fk_histlab_trabajo  foreign  key
(TRABAJO_COD) references trabajos(TRABAJO_COD),

        constraint  fk_histlab_departamento  foreign  key
(DPTO_COD) references departamentos(DPTO_COD),

        constraint  fk_histlab_supervisor  foreign  key
(SUPERVISOR_DNI) references empleados(DNI)

) tablespace data_employees;

```

- **Tabla Historial_Salarial:**

```

create table historial_salarial (

        EMPLEADO_DNI number(8),

        SALARIO number not null,

        FECHA_COMIENZO date,

        FECHA_FIN date,

        constraint  pk_historial_salarial  primary  key
(EMPLEADO_DNI, FECHA_COMIENZO),

        constraint  fk_histsal_empleado  foreign  key
(EMPLEADO_DNI) references empleados(DNI)

) tablespace data_employees;

```

- **Tabla Estudios:**

```

create table estudios (

```

```

EMPLEADO_DNI number(8),

UNIVERSIDAD number(5),

AÑO number,

GRADO varchar2(3),

ESPECIALIDAD varchar2(20),

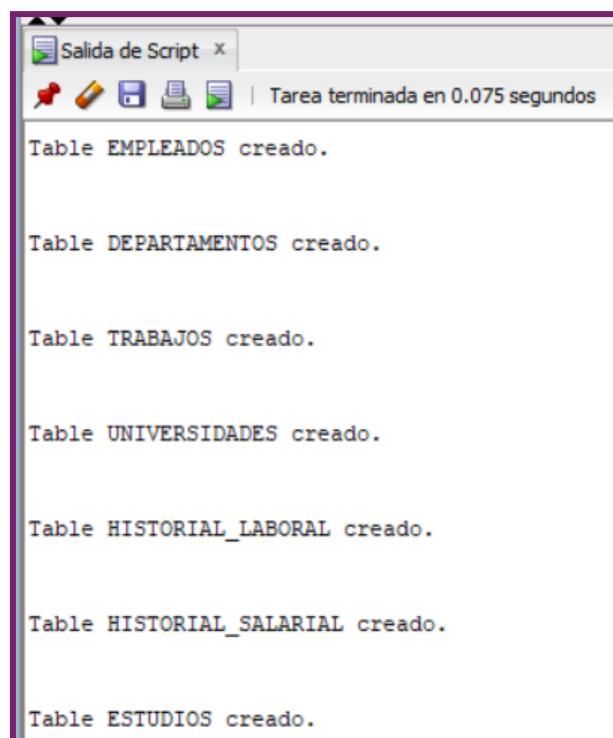
    constraint pk_estudios primary key (EMPLEADO_DNI,
UNIVERSIDAD),

    constraint fk_estudios_empleado foreign key
(EMPLEADO_DNI) references empleados(DNI),

    constraint fk_estudios_univ foreign key
(UNIVERSIDAD) references universidades(UNIV_COD)

) tablespace data_employees;

```



Actividad 02. Restricción de sexo en EMPLEADOS.

Se restringió el atributo SEXO a los valores 'H' o 'M' mediante CHECK en EMPLEADOS.

Actividad 03. Restricciones de unicidad en nombres en DEPARTAMENTOS y TRABAJOS.

Se asegura que no existan dos departamentos ni dos trabajos con el mismo nombre mediante UNIQUE en DEPARTAMENTOS y TRABAJOS.

Actividad 04. Evitar solapamientos de salarios y trabajos.

Se crean triggers para impedir que un empleado tenga salarios o trabajos solapados en fechas.

- **Triggers en Historial_Salarial:**

```
create or replace trigger trg_histsal_no_solape
before insert or update on historial_salarial
for each row
declare
    v_count integer;
begin
    select count(*) into v_count
    from historial_salarial
    where EMPLEADO_DNI = :new.EMPLEADO_DNI
        and (:new.FECHA_FIN is null or FECHA_COMIENZO <=
: new.FECHA_FIN)
        and (FECHA_FIN is null or :new.FECHA_COMIENZO <=
FECHA_FIN)
        and not (EMPLEADO_DNI = :new.EMPLEADO_DNI and
FECHA_COMIENZO = :new.FECHA_COMIENZO);
    if v_count > 0 then
        raise_application_error(-20001, 'Ya existe un
salario vigente en ese periodo.');
```

```
        end if;

    end;

/
```

- **Trigger en Historial_Laboral:**

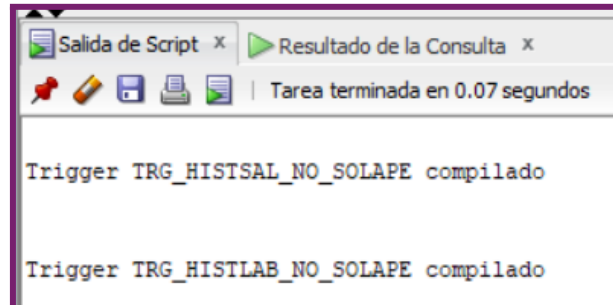
```
create or replace trigger trg_histlab_no_solape
before insert or update on historial_laboral
for each row
declare
    v_count integer;
begin
    select count(*) into v_count
    from historial_laboral
    where EMPLEADO_DNI = :new.EMPLEADO_DNI
        and (:new.FECHA_FIN is null or FECHA_INICIO <=
:new.FECHA_FIN)
        and (FECHA_FIN is null or :new.FECHA_INICIO <=
FECHA_FIN)
        and not (EMPLEADO_DNI = :new.EMPLEADO_DNI and
TRABAJO_COD = :new.TRABAJO_COD and FECHA_INICIO =
:new.FECHA_INICIO);

    if v_count > 0 then
        raise_application_error(-20002, 'Ya existe un
trabajo vigente en ese periodo.');
```

```
    end if;
```

```
end;
```

```
/
```



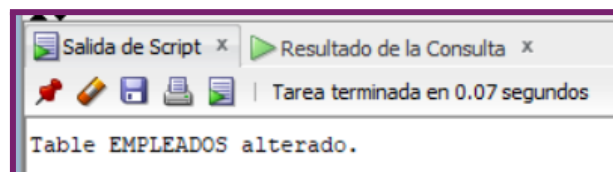
Actividad 05. Declaración de claves primarias.

Cada tabla cuenta con su clave primaria definida al momento de la creación, garantizando la unicidad de los registros.

Actividad 06. Agregar teléfono y celular en EMPLEADOS.

Se amplía la tabla EMPLEADOS permitiendo registrar números de teléfono fijo y celular de cada empleado.

```
alter table empleados  
  
add (TELEFONO varchar2(12),CELULAR varchar2(12));
```



Actividad 07. Inserción de filas de ejemplo.

Se insertan datos de prueba para departamentos, trabajos, universidades, empleados, historial laboral, historial salarial y estudios.

- **Departamentos:**

```
insert into departamentos (DPTO_COD, NOMBRE_DPTO,  
DPTO_PADRE, PRESUPUESTO, PRES_ACTUAL) values (10,  
'Administracion', null, 50000, 30000);  
  
insert into departamentos (DPTO_COD, NOMBRE_DPTO,  
DPTO_PADRE, PRESUPUESTO, PRES_ACTUAL) values (20,  
'Contabilidad', null, 40000, 20000);  
  
insert into departamentos (DPTO_COD, NOMBRE_DPTO,  
DPTO_PADRE, PRESUPUESTO, PRES_ACTUAL) values (30,  
'Recursos Humanos', null, 30000, 15000);  
  
insert into departamentos (DPTO_COD, NOMBRE_DPTO,  
DPTO_PADRE, PRESUPUESTO, PRES_ACTUAL) values (40,  
'Tecnologia', null, 80000, 60000);
```

- **Trabajos:**

```
insert into trabajos (TRABAJO_COD, NOMBRE_TRAB,  
SALARIO_MIN, SALARIO_MAX) values (100, 'Analista', 1000,  
5000);  
  
insert into trabajos (TRABAJO_COD, NOMBRE_TRAB,  
SALARIO_MIN, SALARIO_MAX) values (200, 'Cajero', 900,  
2500);  
  
insert into trabajos (TRABAJO_COD, NOMBRE_TRAB,  
SALARIO_MIN, SALARIO_MAX) values (300, 'Gerente', 5000,  
12000);  
  
insert into trabajos (TRABAJO_COD, NOMBRE_TRAB,  
SALARIO_MIN, SALARIO_MAX) values (400, 'Programador',  
1200, 6000);
```

- **Universidades:**

```
insert into universidades (UNIV_COD, NOMBRE_UNIV,
```

```

CIUDAD, MUNICIPIO, COD_POSTAL) values (1, 'UNMSM',
'Lima', 'Lima', '15001');

insert into universidades (UNIV_COD, NOMBRE_UNIV,
CIUDAD, MUNICIPIO, COD_POSTAL) values (2, 'PUCP',
'Lima', 'San Miguel', '15086');

insert into universidades (UNIV_COD, NOMBRE_UNIV,
CIUDAD, MUNICIPIO, COD_POSTAL) values (3, 'UNSA',
'Arequipa', 'Arequipa', '04001');

insert into universidades (UNIV_COD, NOMBRE_UNIV,
CIUDAD, MUNICIPIO, COD_POSTAL) values (4, 'UNSAAC',
'Cusco', 'Cusco', '08001');

```

- **Empleados:**

```

insert into empleados (DNI, NOMBRE, APELLIDO1,
APELLIDO2, DIRECC1, DIRECC2, CIUDAD, PROVINCIA,
COD_POSTAL, SEXO, FECHA_NAC, TELEFONO, CELULAR) values
(111222, 'Sergio', 'Palma', 'Entrena', null, null, null,
null, null, 'H', null, null, null);

insert into empleados (DNI, NOMBRE, APELLIDO1,
APELLIDO2, DIRECC1, DIRECC2, CIUDAD, PROVINCIA,
COD_POSTAL, SEXO, FECHA_NAC, TELEFONO, CELULAR) values
(222333, 'Lucia', 'Ortega', 'Plus', null, null, null,
null, null, 'M', null, null, null);

insert into empleados (DNI, NOMBRE, APELLIDO1,
APELLIDO2, DIRECC1, DIRECC2, CIUDAD, PROVINCIA,
COD_POSTAL, SEXO, FECHA_NAC, TELEFONO, CELULAR) values
(333444, 'Carlos', 'Fernandez', 'Lopez', 'Av. Ejercito
789', null, 'Arequipa', 'Arequipa', '04001', 'H',
to_date('1963-02-10', 'yyyy-mm-dd'), '054765432',
'986987654');

insert into empleados (DNI, NOMBRE, APELLIDO1,

```



```
APELLIDO2,    DIRECC1,    DIRECC2,    CIUDAD,    PROVINCIA,
COD_POSTAL, SEXO, FECHA_NAC, TELEFONO, CELULAR) values
(444555, 'Marina', 'Seclen', 'Diaz', 'Av. Grau 321',
null, 'Lima', 'Lima', '15002', 'M',
to_date('1962-04-08','yyyy-mm-dd'), '014444555',
'999888777');
```

- **Historial_Laboral:**

```
insert into historial_laboral (EMPLEADO_DNI,
TRABAJO_COD, FECHA_INICIO, FECHA_FIN, DPTO_COD,
SUPERVISOR_DNI) values (111222, 100,
to_date('1996-06-16','YYYY-MM-DD'), null, 10, null);

insert into historial_laboral (EMPLEADO_DNI,
TRABAJO_COD, FECHA_INICIO, FECHA_FIN, DPTO_COD,
SUPERVISOR_DNI) values (222333, 200,
to_date('1998-03-01','YYYY-MM-DD'), null, 20, 111222);

insert into historial_laboral (EMPLEADO_DNI,
TRABAJO_COD, FECHA_INICIO, FECHA_FIN, DPTO_COD,
SUPERVISOR_DNI) values (333444, 300,
to_date('1995-05-05','YYYY-MM-DD'), null, 30, null);

insert into historial_laboral (EMPLEADO_DNI,
TRABAJO_COD, FECHA_INICIO, FECHA_FIN, DPTO_COD,
SUPERVISOR_DNI) values (444555, 400,
to_date('1999-07-04','YYYY-MM-DD'), null, 40, null);
```

- **Historial_Salarial:**

```
insert into historial_salarial (EMPLEADO_DNI, SALARIO,
FECHA_COMIENZO, FECHA_FIN) values (111222, 2500,
to_date('1996-06-16','YYYY-MM-DD'), null);

insert into historial_salarial (EMPLEADO_DNI, SALARIO,
```

```
FECHA_COMIENZO,    FECHA_FIN)    values    (222333,    2000,
to_date('1998-03-01','YYYY-MM-DD'), null);

insert into historial_salarial (EMPLEADO_DNI, SALARIO,
FECHA_COMIENZO,    FECHA_FIN)    values    (333444,    6000,
to_date('1995-05-05','YYYY-MM-DD'), null);

insert into historial_salarial (EMPLEADO_DNI, SALARIO,
FECHA_COMIENZO,    FECHA_FIN)    values    (444555,    4800,
to_date('1999-07-04','YYYY-MM-DD'), null);
```

- **Estudios:**

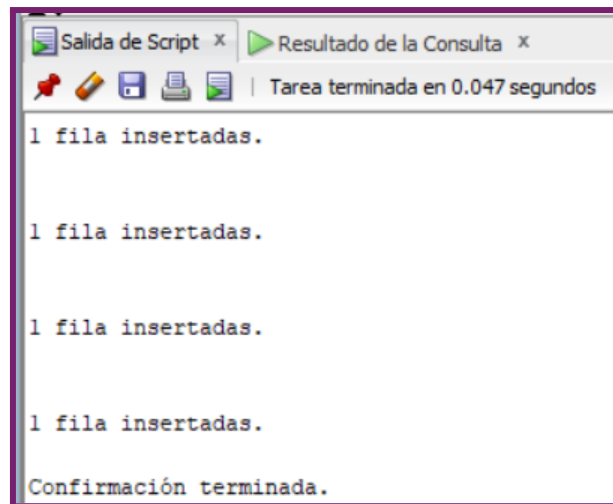
```
insert into estudios (EMPLEADO_DNI, UNIVERSIDAD, AÑO,
GRADO, ESPECIALIDAD) values (111222, 1, 1995, 'LIC',
'Administracion');

insert into estudios (EMPLEADO_DNI, UNIVERSIDAD, AÑO,
GRADO, ESPECIALIDAD) values (222333, 2, 1997, 'LIC',
'Administración');

insert into estudios (EMPLEADO_DNI, UNIVERSIDAD, AÑO,
GRADO, ESPECIALIDAD) values (333444, 2, 1987, 'ING',
'Industrial');

insert into estudios (EMPLEADO_DNI, UNIVERSIDAD, AÑO,
GRADO, ESPECIALIDAD) values (444555, 1, 1990, 'ING',
'Software');

commit;
```



Actividad 08. Validación de supervisor inexistente.

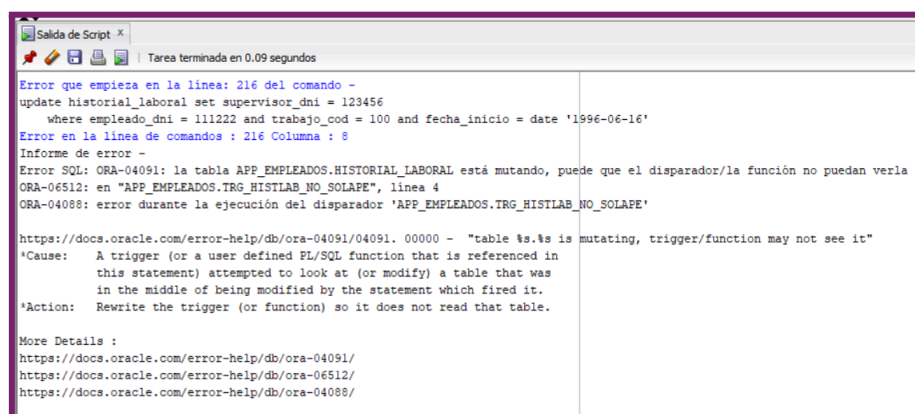
Se prueba la restricción de integridad referencial para evitar asignar un supervisor que no existe.

```

update historial_laboral set supervisor_dni = 123456

      where empleado_dni = 111222 and trabajo_cod = 100 and
fecha_inicio = date '1996-06-16';

```



Resultado esperado: Oracle rechaza la operación porque no existe un empleado con DNI = 123456.

Actividad 09. Borrar una universidad y su efecto en ESTUDIOS.

Si una universidad tiene registros en ESTUDIOS, no se puede borrar directamente.

```
delete from universidades where univ_cod = '1';

-- Generará error por FK
```

Para permitir el borrado de la universidad y sus estudios relacionados, se debe redefinir la FK con ON DELETE CASCADE:

```
-- 1) Ver el nombre actual de la constraint

select constraint_name from user_constraints

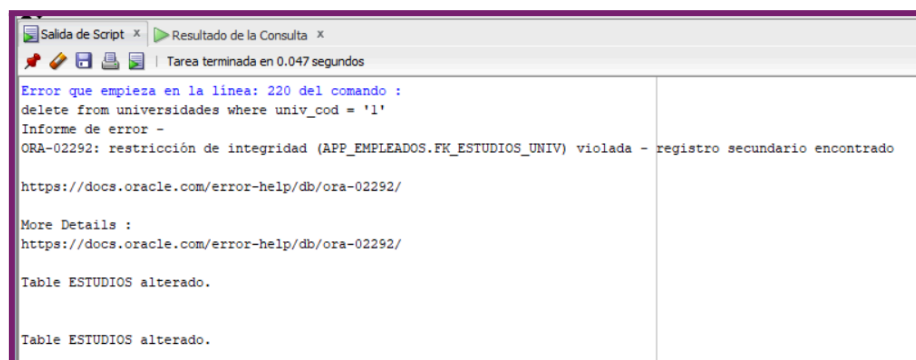
      where table_name = 'ESTUDIOS' and constraint_type = 'R';

-- 2) Borrar la constraint existente (fk_estudios_univ):

alter table ESTUDIOS drop constraint fk_estudios_univ;

-- 3) Crear la nueva FK con ON DELETE CASCADE:

alter table ESTUDIOS add constraint fk_estudios_univ foreign
key  (UNIVERSIDAD) references  UNIVERSIDADES  (UNIV_COD) on
delete cascade;
```

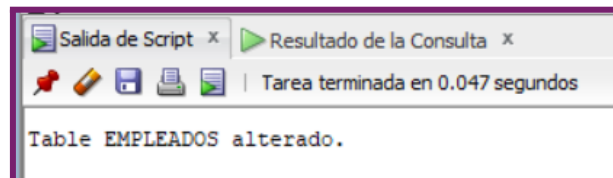


Actividad 10. Forzar COD_POSTAL NOT NULL cuando CIUDAD esté rellenada.

Se define un CHECK para garantizar que si se ingresa una ciudad, también se debe ingresar un código postal.

```
alter table empleados add constraint chk_ciudad_codpostal  
check (ciudad is null or cod_postal is not null);
```

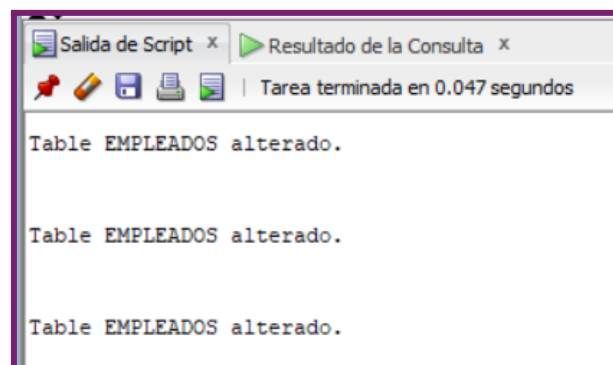
Si existen filas que violan esta condición, la sentencia fallará hasta que se corrijan los datos.



Actividad 11. Añadir VALORACION con valor por defecto 5 (1–10).

Se agrega una columna para registrar una valoración de cada empleado, con valor por defecto y restricción de rango.

```
-- Agregar columna con valor por defecto  
  
alter table empleados add (valoracion number(2) default 5);  
  
-- Forzar rango entre 1 y 10  
  
alter table empleados add constraint chk_valoracion_range  
check (valoracion between 1 and 10);  
  
-- Forzar que la columna sea not null  
  
alter table empleados modify (valoracion not null);
```

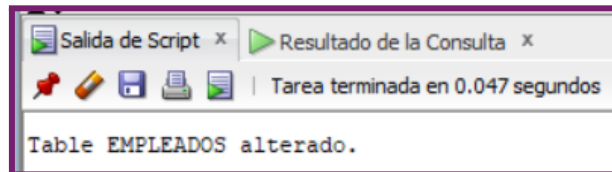


Esto garantiza que todos los empleados tengan un valor de valoración válido entre 1 y 10.

Actividad 12. Eliminar la restricción NOT NULL en EMPLEADOS.NOMBRE.

Se permite que el campo NOMBRE pueda quedar vacío si es necesario.

```
alter table empleados modify (nombre varchar2(20) null);
```

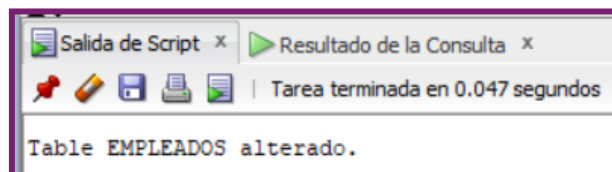


Esto puede ser útil en escenarios donde el nombre aún no esté disponible.

Actividad 13. Modificar DIRECC1 a VARCHAR2(40).

Se amplía la longitud de la dirección principal para permitir datos más largos.

```
alter table empleados modify (direcc1 varchar2(40));
```



Actividad 14. Conversión de FECHA_NAC a cadena.

No es recomendable convertir directamente la columna FECHA_NAC de tipo DATE a VARCHAR2, ya que se perderían las capacidades de comparación temporal y cálculos de edad o antigüedad.

En lugar de modificar la columna, se puede mostrar la fecha como texto utilizando TO_CHAR al consultar los datos:

```
-- No se recomienda:
```

```

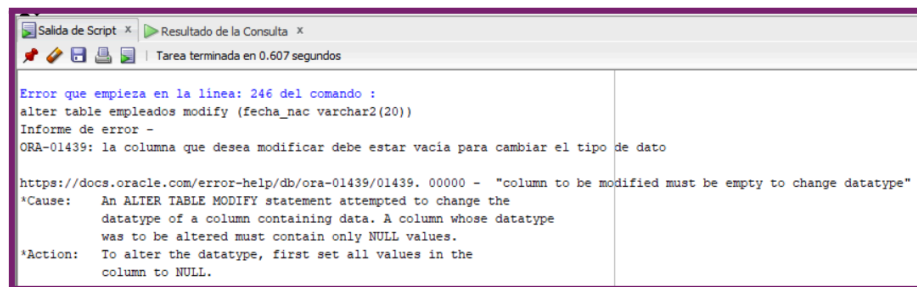
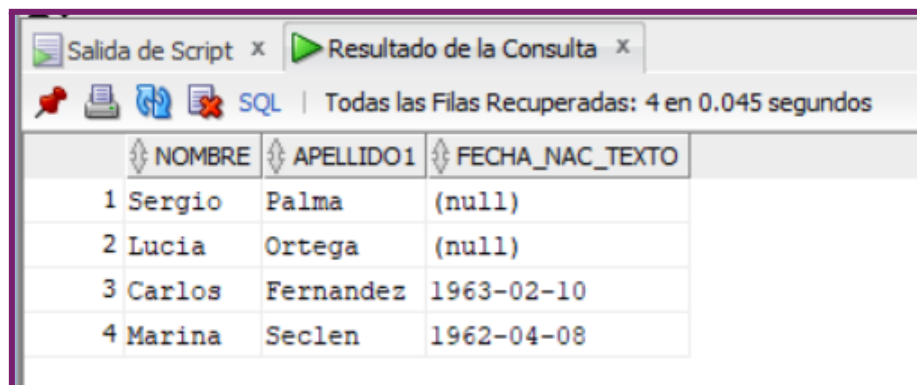
alter table empleados modify (fecha_nac varchar2(20));

-- Mejor usar TO_CHAR en consultas

select nombre, apellido1, to_char(fecha_nac, 'YYYY-MM-DD') as
fecha_nac_texto

from empleados;

```

Salida de Script x Resultado de la Consulta x

Todas las Filas Recuperadas: 4 en 0.045 segundos

	NOMBRE	APELLIDO1	FECHA_NAC_TEXTO
1	Sergio	Palma	(null)
2	Lucia	Ortega	(null)
3	Carlos	Fernandez	1963-02-10
4	Marina	Seclen	1962-04-08

Esto permite mostrar la fecha como texto sin comprometer la integridad de los cálculos y operaciones sobre fechas.

Actividad 15. Cambiar la clave primaria de EMPLEADOS al NOMBRE y los dos APELLIDOS.

Se reemplaza la PK basada en DNI por una PK compuesta de nombre y apellidos, actualizando todas las tablas dependientes.

```

-- 1: Eliminar las FOREIGN KEYS dependientes de EMPLEADOS

alter      table      historial_salarial      drop      constraint

```

```

fk_histsal_empleado;

alter      table      historial_laboral      drop      constraint
fk_histlab_empleado;

alter      table      historial_laboral      drop      constraint
fk_histlab_supervisor;

alter table estudios drop constraint fk_estudios_empleado;

-- 2: Eliminar la PRIMARY KEY actual (DNI)

alter table empleados drop constraint SYS_C008267;

-- 3: Crear la nueva PRIMARY KEY compuesta

alter table empleados add constraint pk_empleados_nom_ape
primary key (nombre, apellido1, apellido2);

-- Paso 4: Modificar las tablas dependientes para apuntar a la
nueva PK

-- HISTORIAL_SALARIAL

alter table historial_salarial add (

    nombre varchar2(10), apellido1 varchar2(15), apellido2
varchar2(15));

alter table historial_salarial add constraint fk_histsal_emp
foreign key (nombre, apellido1, apellido2)

    references empleados (nombre, apellido1, apellido2);

-- HISTORIAL_LABORAL

alter table historial_laboral add (

    nombre varchar2(10), apellido1 varchar2(15), apellido2
varchar2(15),

    nombre_sup varchar2(10), apellido1_sup varchar2(15),
apellido2_sup varchar2(15));

alter table historial_laboral add constraint fk_histlab_emp

```



```

foreign key (nombre, apellido1, apellido2)

references empleados (nombre, apellido1, apellido2);

alter table historial_laboral add constraint fk_histlab_sup
foreign key (nombre_sup, apellido1_sup, apellido2_sup)

references empleados (nombre, apellido1, apellido2);

-- ESTUDIOS

alter table estudios add (

nombre varchar2(10), apellido1 varchar2(15), apellido2
varchar2(15));

alter table estudios add constraint fk_estudios_emp foreign
key (nombre, apellido1, apellido2)

references empleados (nombre, apellido1, apellido2);

```

Actividad 16. Crear tabla INFORMACION UNIVERSITARIA.

Se crea una tabla para almacenar el nombre completo de los empleados junto con su universidad.

```

create table informacion_universitaria (

nombre_completo varchar2(50), universidad varchar2(25)

) tablespace data_employees;

```

- **Insertión de datos:**

```

insert into informacion_universitaria (nombre_completo,
universidad)

```

```
select

    e.nombre || ' ' || e.apellido1 || ' ' || e.apellido2
as nombre_completo,

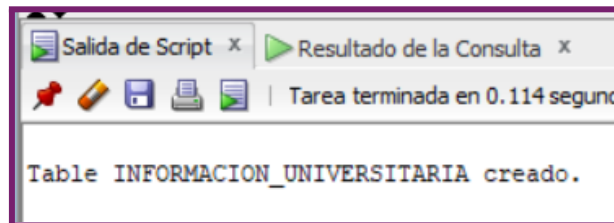
    u.nombre_univ as universidad

from empleados e

join estudios s on e.dni = s.empleado_dni

join universidades u on s.universidad = u.univ_cod;

commit;
```



Esto permite consultar fácilmente la relación entre empleados y universidades.

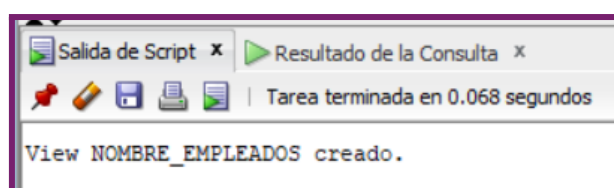
Actividad 17. Crear vista NOMBRE_EMPLEADOS para empleados de Málaga.

Se crea una vista que muestra los nombres completos de los empleados que residen en Málaga.

```
create view nombre_empleados as

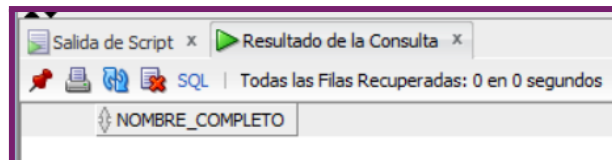
select nombre || ' ' || apellido1 || ' ' || apellido2 as
nombre_completo

from empleados where ciudad = 'Málaga';
```



Esta vista permite consultar rápidamente los nombres completos de los empleados de Málaga.

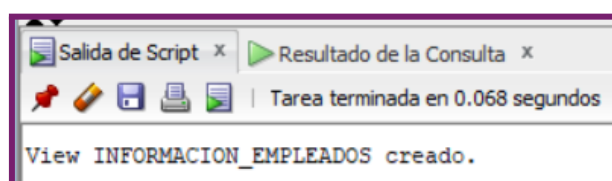
```
select * from nombre_empleados;
```



Actividad 18. Crear vista INFORMACION_EMPLEADOS con edad.

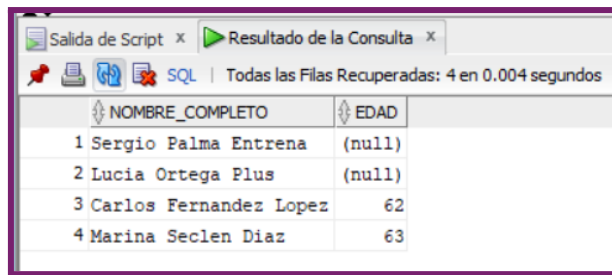
Se crea una vista que incluye el nombre completo de cada empleado y su edad calculada a partir de la fecha de nacimiento.

```
create view informacion_empleados as  
  
select nombre || ' ' || apellido1 || ' ' || apellido2 as  
nombre_completo,  
  
       floor(months_between(sysdate, fecha_nac)/12) as edad  
  
from empleados;
```



Esto facilita obtener información básica de los empleados junto con su edad sin necesidad de cálculos repetidos.

```
select * from informacion_empleados;
```



	NOMBRE_COMPLETO	EDAD
1	Sergio Palma Entrena	(null)
2	Lucia Ortega Plus	(null)
3	Carlos Fernandez Lopez	62
4	Marina Seclen Diaz	63

Actividad 19. Crear vista INFORMACION_ACTUAL con salario.

Se genera una vista combinando la información de informacion_empleados con los salarios vigentes.

```
create view informacion_actual as

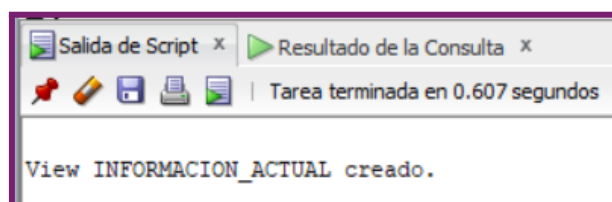
select ie.nombre_completo, ie.edad, hs.salario

from informacion_empleados ie

join historial_salarial hs

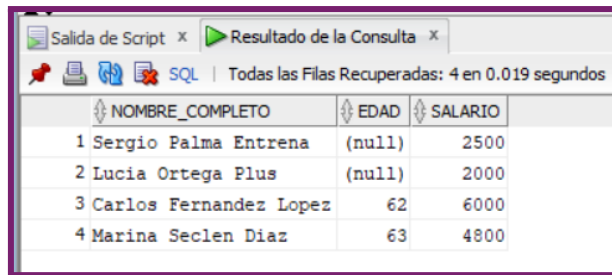
on ie.nombre_completo = (select nombre || ' ' || apellido1
|| ' ' || apellido2 from empleados e where e.dni =
hs.empleado_dni)

where hs.fecha_fin is null;
```



La vista permite consultar el nombre, edad y salario actual de cada empleado de forma rápida y sencilla.

```
select * from informacion_actual;
```



Salida de Script x Resultado de la Consulta x

Todas las Filas Recuperadas: 4 en 0.019 segundos

	NOMBRE_COMPLETO	EDAD	SALARIO
1	Sergio Palma Entrena	(null)	2500
2	Lucia Ortega Plus	(null)	2000
3	Carlos Fernandez Lopez	62	6000
4	Marina Seclen Diaz	63	4800

Actividad 20. Borrar todas las tablas.

Se eliminan todas las tablas de la base de datos, usando CASCADE CONSTRAINTS para evitar errores de integridad referencial.

```
drop table estudios cascade constraints;

drop table historial_salarial cascade constraints;

drop table historial_laboral cascade constraints;

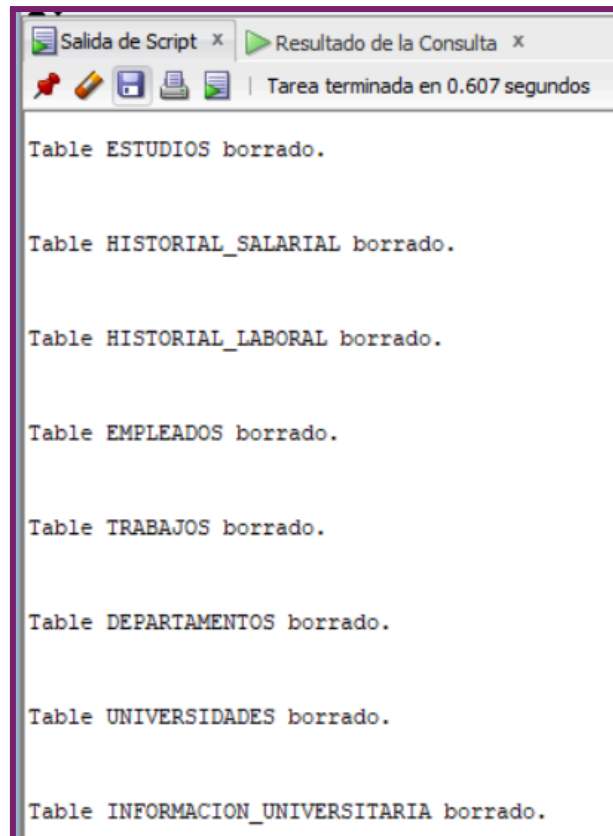
drop table empleados cascade constraints;

drop table trabajos cascade constraints;

drop table departamentos cascade constraints;

drop table universidades cascade constraints;

drop table informacion_universitaria cascade constraints;
```



```
Salida de Script x Resultado de la Consulta x
Tarea terminada en 0.607 segundos

Table ESTUDIOS borrado.

Table HISTORIAL_SALARIAL borrado.

Table HISTORIAL_LABORAL borrado.

Table EMPLEADOS borrado.

Table TRABAJOS borrado.

Table DEPARTAMENTOS borrado.

Table UNIVERSIDADES borrado.

Table INFORMACION_UNIVERSITARIA borrado.
```

Al borrar tablas con claves foráneas dependientes, CASCADE CONSTRAINTS elimina automáticamente las restricciones asociadas.