Universidad Nacional Autónoma de México

Facultad de ingeniería, Ingeniería en computación.

Bases de Datos.

Ejercicio 3.7

PROFESOR: ING. FERNANDO ARREOLA FRANCO

Semestre 2024-2

Grupo: 1

Fecha de elaboración: 08/04/2024

Alumno: Soto Huerta Gustavo Isaac

Número de Cuenta: 318201458

1) Separar nombre y apellido de la tabla asesor (si aplica)

```
SELECT
   SPLIT_PART(NOMBRE_ASESOR, ' ', 1) AS Nombre,
   SPLIT_PART(NOMBRE_ASESOR, ' ', 2) AS Apellido
FROM ASESOR;
```

2) Alumno U Asesor (sólo en el atributo nombre)

```
(SELECT NOMBRE_ALUMNO AS Nombre FROM ALUMNO)
UNION
(SELECT NOMBRE_ASESOR AS Nombre FROM ASESOR);
```

3) Alumno INTERSECT Asesor (sólo en el atributo nombre)

```
(SELECT NOMBRE_ALUMNO AS Nombre FROM ALUMNO)
INTERSECT
(SELECT NOMBRE_ASESOR AS Nombre FROM ASESOR);
```

4) Alumno - Asesor y Asesor - Alumno

```
-- Mostrar relaciones entre alumnos y asesores en ambas direcciones
WITH Alumno_Asesor AS (
   SELECT
      AL.NOMBRE_ALUMNO,
      COALESCE(ASE.NOMBRE_ASESOR, 'Sin Asesor') AS NOMBRE_ASESOR
   FROM
      ALUMNO AS AL
   LEFT JOIN
      ASESOR AS ASE ON AL.ID_ASESOR = ASE.ID_ASESOR
), Asesor_Alumno AS (
   SELECT
      COALESCE(ASE.NOMBRE_ASESOR, 'Sin Asesor') AS NOMBRE_ASESOR,
      COALESCE(AL.NOMBRE_ALUMNO, 'Sin Alumno') AS NOMBRE_ALUMNO
   FROM
      ASESOR AS ASE
   LEFT JOIN
      ALUMNO AS AL ON ASE.ID_ASESOR = AL.ID_ASESOR
)
SELECT * FROM Alumno_Asesor
UNION ALL
SELECT * FROM Asesor_Alumno;
```

5)Alumno natural join Asesor y Alumno cross join asesor

```
-- Natural Join entre Alumno y Asesor
SELECT
   NOMBRE_ALUMNO,
   NOMBRE_ASESOR
FROM
   ALUMNO
NATURAL JOIN
   ASESOR;

-- Cross Join entre Alumno y Asesor
SELECT
   AL.NOMBRE_ALUMNO,
   ASE.NOMBRE_ASESOR
FROM
   ALUMNO AS AL
CROSS JOIN
   ASESOR AS ASE;
```

6) Modificar la tabla Alumno, agregue los siguientes atributos: carrera varchar(40) y edad smallint

```
-- Modificar la tabla ALUMNO para agregar los campos CARRERA y EDAD
ALTER TABLE ALUMNO
ADD COLUMN CARRERA VARCHAR(40),
ADD COLUMN EDAD SMALLINT;

-- Actualizar registros existentes con valores predeterminados o específicos
UPDATE ALUMNO
SET CARRERA = 'Ingeniería', EDAD = 22
WHERE ID_ALUMNO = 1;

UPDATE ALUMNO
SET CARRERA = 'Matemáticas', EDAD = 23
WHERE ID_ALUMNO = 2;

UPDATE ALUMNO
SET CARRERA = 'Física', EDAD = 21
WHERE ID_ALUMNO = 3;

UPDATE ALUMNO
SET CARRERA = 'Química', EDAD = 24
WHERE ID_ALUMNO = 4;

UPDATE ALUMNO
SET CARRERA = 'Biología', EDAD = 22
```

```
WHERE ID_ALUMNO = 5;
```

7) Insertar los siguientes 5 registros:
id_Alumno,nombre,ap_paterno, carrera,edad,id_asesor
6,Isaac,Lemus,Petrolera,30,as-1
7,Gabriela,Suarezs,Industrial,24,as-3
8,Pablo,Gonzalez,Computacion,23,as-2
9,David,Rivera,Industrial,25,as-1
10,Dayana,Plata,Computacion,24,as-4

```
-- Insertar nuevos registros en la tabla ALUMNO
INSERT INTO ALUMNO (ID_ALUMNO, NOMBRE_ALUMNO, CARRERA, EDAD,
ID_ASESOR)
VALUES
    (6, 'Isaac Lemus', 'Petrolera', 30, 'as-1'),
    (7, 'Gabriela Suarezs', 'Industrial', 24, 'as-3'),
    (8, 'Pablo Gonzalez', 'Computacion', 23, 'as-2'),
    (9, 'David Rivera', 'Industrial', 25, 'as-1'),
    (10, 'Dayana Plata', 'Computacion', 24, 'as-4');
```

8) Actualizar los 5 registros iniciales para asignar valores en los atributos agregados
1,Petrolera,27
2,Telecomunicaciones,24
3,Computacion,27
4,Industrial,25
5,Computacion,19

```
-- Actualizar los registros iniciales para asignar valores en los atributos agregados
UPDATE ALUMNO
SET CARRERA = 'Petrolera', EDAD = 27
WHERE ID_ALUMNO = 1;

UPDATE ALUMNO
SET CARRERA = 'Telecomunicaciones', EDAD = 24
WHERE ID_ALUMNO = 2;

UPDATE ALUMNO
SET CARRERA = 'Computacion', EDAD = 27
WHERE ID_ALUMNO = 3;

UPDATE ALUMNO
SET CARRERA = 'Industrial', EDAD = 25
WHERE ID_ALUMNO = 4;

UPDATE ALUMNO
SET CARRERA = 'Computacion', EDAD = 19
```

```
WHERE ID_ALUMNO = 5;
```

9) Validar que los datos coincidan con los esperado

```
-- Validar los datos en la tabla ALUMNO
SELECT
    AL.ID_ALUMNO,
    AL.NOMBRE_ALUMNO,
    AL.CARRERA,
    AL.EDAD,
    COALESCE(ASE.NOMBRE_ASESOR, 'Sin Asesor') AS NOMBRE_ASESOR
FROM
    ALUMNO AS AL
LEFT JOIN
    ASESOR AS ASE ON AL.ID_ASESOR = ASE.ID_ASESOR;
```

10) Nombre completo del alumno de mayor edad

```
-- Obtener el nombre completo del alumno de mayor edad
SELECT
    NOMBRE_ALUMNO,
    CARRERA,
    EDAD
FROM
    ALUMNO
ORDER BY
    EDAD DESC
LIMIT 1;
```

11) Nombre completo del alumno de menor edad

```
-- Obtener el nombre completo del alumno de menor edad
SELECT
    NOMBRE_ALUMNO,
    CARRERA,
    EDAD
FROM
    ALUMNO
ORDER BY
    EDAD ASC
LIMIT 1
```

12) cantidad de alumnos por carrera

```
-- Obtener la cantidad de alumnos por carrera
SELECT
```

```
    CARRERA,
    COUNT(*) AS CANTIDAD_ALUMNOS
FROM
    ALUMNO
GROUP BY
    CARRERA
ORDER BY
    CARRERA;
```

13) Nombre de la carrera que tiene a la persona más joven. Agrupar los datos.

```
-- Obtener el nombre de la carrera que tiene a la persona más joven
SELECT
    CARRERA
FROM
    ALUMNO
WHERE
    EDAD = (SELECT MIN(EDAD) FROM ALUMNO)
GROUP BY
    CARRERA;
```

14) Nombre de la carrera que tiene a la persona más grande. Agrupar los datos.

```
-- Obtener el nombre de la carrera que tiene a la persona más grande
SELECT
    CARRERA
FROM
    ALUMNO
WHERE
    EDAD = (SELECT MAX(EDAD) FROM ALUMNO)
GROUP BY
    CARRERA;
```

15) Nombre de la carrera que tiene a la persona más joven. Usar subconsultas.

```
-- Nombre de la carrera que tiene a la persona más joven usando subconsultas
SELECT
    CARRERA
FROM
    ALUMNO
WHERE
    EDAD = (SELECT MIN(EDAD) FROM ALUMNO);
```

16) Nombre de la carrera que tiene a la persona más grande. Usar subconsultas

```
-- Nombre de la carrera que tiene a la persona más grande usando subconsultas
SELECT
   CARRERA
FROM
   ALUMNO
WHERE
   EDAD = (SELECT MAX(EDAD) FROM ALUMNO);
```

17) Promedio de edad por carrera

```
-- Obtener el promedio de edad por carrera
SELECT
   CARRERA,
   ROUND(AVG(EDAD), 2) AS PROMEDIO_EDAD
FROM
   ALUMNO
GROUP BY
   CARRERA
ORDER BY
   CARRERA;
```

18) borrar al asesor Adolfo Millan

```
-- Establecer ID_ASESOR a NULL para los alumnos de Adolfo Millan
UPDATE ALUMNO
SET ID_ASESOR = NULL
WHERE ID_ASESOR = 'as-3';

-- Eliminar el asesor Adolfo Millan
DELETE FROM ASESOR
WHERE NOMBRE_ASESOR = 'Adolfo Millan';
```

19) Actualizar el id del asesor Fernando Arreola, asignar "as-5"

```
-- Actualizar el ID del asesor Fernando Arreola a "as-5"
UPDATE ASESOR
SET ID_ASESOR = 'as-5'
WHERE NOMBRE_ASESOR = 'Fernando Arreola';

-- Actualizar los registros relacionados en la tabla ALUMNO
UPDATE ALUMNO
SET ID_ASESOR = 'as-5'
WHERE ID_ASESOR = 'as-4';
```

20) inicie sesion en dos terminales , conectandose en ambas a la base de datos donde almaceno los datos del presente ejercicio
- en ambas sesiones, inicie una transaccion
- en la ventana A, actualice el apellido del alumno con id = 7 por "Suarez"
- en la ventana B, seleccione toda la información del alumno con id = 7. ¿Qué observa?
- en la ventaba A, ingrese commit
- en la ventana B, seleccione toda la información del alumno con id = 7, posteriormente rollback. ¿Qué observa?
- justifique los resultados obtenidos

Sesión A
1.Iniciar sesión en la base de datos.

2.Iniciar una transacción.

```
BEGIN;
```

3.Actualizar el apellido del alumno con ID 7 a "Suarez"

```
UPDATE ALUMNO
SET NOMBRE_ALUMNO = 'Gabriela Suarez'
WHERE ID_ALUMNO = 7;
```

Sesión B
1.Iniciar sesión en la base de datos.

2.Iniciar una transacción.

```
BEGIN;
```

3.Seleccionar toda la información del alumno con ID 7.

```
SELECT * FROM ALUMNO
WHERE ID_ALUMNO = 7;
```

Sesión A
1.COMMIT para confirmar los cambios.

```
COMMIT;
```

Sesión B
1.Selecciona nuevamente toda la información del alumno con ID 7.

```
SELECT * FROM ALUMNO
```

```
WHERE ID_ALUMNO = 7;
```

2.Realiza ROLLBACK para deshacer la transacción en la Sesión B.

```
ROLLBACK;
```

3.Seleccionar nuevamente toda la información del alumno con ID 7.

```
SELECT * FROM ALUMNO
WHERE ID_ALUMNO = 7;
```