

Instituto Tecnológico de Mexicali



Ingeniería Sistemas Computacionales

Fundamentos de Base de Datos

Tema:

“U3_Tarea#1”

Estudiante:

Mac Callum Merino Fatima Berenice

No. De control:

C21490774

Docente:

José Ramón Bogarín Valenzuela

Mexicali, B.C., 10 de abril de 2025.

-- Crear tablas

```
CREATE TABLE estudiantes (  
  id SERIAL PRIMARY KEY,  
  nombre VARCHAR(100),  
  email VARCHAR(100),  
  fecha_nacimiento DATE  
);
```

```
CREATE TABLE cursos (  
  id SERIAL PRIMARY KEY,  
  nombre_curso VARCHAR(100),  
  duracion_meses INT  
);
```

```
CREATE TABLE matriculas (  
  id SERIAL PRIMARY KEY,  
  id_estudiante INT REFERENCES estudiantes(id),  
  id_curso INT REFERENCES cursos(id),  
  fecha_matricula DATE  
);
```

-- Insertar datos en estudiantes

```
INSERT INTO estudiantes (nombre, email, fecha_nacimiento) VALUES  
(  
'Ana Torres', 'ana@example.com', '1998-03-12'),  
(  
'Luis Gómez', 'luis@example.com', '2000-07-22'),  
(  
'Carla Ruiz', 'carla@example.com', '1995-11-05');
```

-- Insertar datos en cursos

```
INSERT INTO cursos (nombre_curso, duracion_meses) VALUES  
(  
'Bases de Datos', 4),  
(  
'Programación Web', 6);
```

-- Insertar datos en matriculas

```
INSERT INTO matriculas (id_estudiante, id_curso, fecha_matricula) VALUES  
(  
1, 1, '2025-01-10'),  
(  
2, 1, '2025-01-12'),  
(  
3, 2, '2025-02-05'),  
(  
1, 2, '2025-02-10');
```

-- Consultas CLE

-- Estudiantes matriculados en "Bases de Datos"

```
SELECT e.nombre  
FROM estudiantes e
```

```
JOIN matriculas m ON e.id = m.id_estudiante
JOIN cursos c ON c.id = m.id_curso
WHERE c.nombre_curso = 'Bases de Datos';
```

```
-- Cursos con cantidad de estudiantes matriculados
SELECT c.nombre_curso, COUNT(m.id_estudiante) AS total_estudiantes
FROM cursos c
LEFT JOIN matriculas m ON c.id = m.id_curso
GROUP BY c.nombre_curso;
```

```
-- Estudiantes mayores de 25 años
SELECT nombre, fecha_nacimiento,
       DATE_PART('year', AGE(fecha_nacimiento)) AS edad
FROM estudiantes
WHERE DATE_PART('year', AGE(fecha_nacimiento)) > 25;
```

```
-- Edad promedio de los estudiantes
SELECT ROUND(AVG(DATE_PART('year', AGE(fecha_nacimiento)))) AS
edad_promedio
FROM estudiantes;
```

```
-- Estudiantes ordenados por fecha de nacimiento
SELECT nombre, fecha_nacimiento
FROM estudiantes
ORDER BY fecha_nacimiento ASC;
19:02
```

Problema a resolver: "Analítica y Gestión Académica"

Contexto

Una institución educativa quiere aprovechar su sistema de base de datos para obtener información útil sobre sus estudiantes, los cursos ofrecidos y las matrículas realizadas. Como analista de datos, se te solicita realizar una serie de tareas para mejorar la toma de decisiones académicas.

Parte 1: Verificación y Ajustes de Estructura (LDD)

Verifica si la base de datos contiene una columna para almacenar el número de teléfono de los estudiantes. Si no existe, agrégala a la tabla estudiantes.

```

-- Parte 1: Verificación y Ajustes de Estructura (LDD)

-- Verifica si la base de datos contiene una columna para almacenar el número de teléfono de los estudiantes.
-- Si no existe, agrégala a la tabla estudiantes.
✓ ALTER TABLE estudiantes
  ADD COLUMN telefono VARCHAR(15);

```

id	nombre	email	fecha_nacimiento	telefono
[PK] integer	character varying (100)	character varying (100)	date	character varying (15)
1	Ana Torres	ana@example.com	1998-03-12	6861111111
2	Luis Gómez	luis@example.com	2000-07-22	6862121221
3	Carla Ruiz	carla@example.com	1995-11-05	6861234567

Modifica la tabla cursos para que el nombre del curso no pueda repetirse.

```

-- Modifica la tabla cursos para que el nombre del curso no pueda repetirse.
ALTER TABLE cursos
  ADD CONSTRAINT nombre_curso_unico UNIQUE (nombre_curso);

```

Parte 2: Carga y Ajuste de Datos (LMD)

Actualiza el email de "Luis Gómez" a luisgomez@universidad.edu.

```

-- Parte 2: Carga y Ajuste de Datos (LMD)

--Actualiza el email de "Luis Gómez" a luisgomez@universidad.edu.
UPDATE estudiantes
  SET email = 'luisgomez@universidad.edu'
  WHERE nombre = 'Luis Gómez';

```

Registra una nueva matrícula para "Carla Ruiz" en el curso "Bases de Datos", con fecha 2025-04-01.

```

--Registra una nueva matrícula para "Carla Ruiz" en el curso "Bases de Datos", con fecha 2025-04-01.
INSERT INTO matriculas (id_estudiante, id_curso, fecha_matricula)
  SELECT e.id, c.id, '2025-04-01'
  FROM estudiantes e
  JOIN cursos c ON c.nombre_curso = 'Bases de Datos'
  WHERE e.nombre = 'Carla Ruiz';

```

Elimina la matrícula de "Ana Torres" del curso "Bases de Datos".

```
--Elimina la matrícula de "Ana Torres" del curso "Bases de Datos".
DELETE FROM matriculas
WHERE id_estudiante = (
    SELECT id FROM estudiantes WHERE nombre = 'Ana Torres'
)
AND id_curso = (
    SELECT id FROM cursos WHERE nombre_curso = 'Bases de Datos'
);
```

Parte 3: Consultas Avanzadas (CLE)

Muestra un listado con el nombre de cada estudiante, el nombre del curso al que está matriculado y la fecha de matrícula.

```
--Parte 3: Consultas Avanzadas (CLE)

--Muestra un listado con el nombre de cada estudiante, el nombre del curso al que está matriculado y la fecha de matrícula.
SELECT e.nombre AS estudiante, c.nombre_curso AS curso, m.fecha_matricula
FROM estudiantes e
JOIN matriculas m ON e.id = m.id_estudiante
JOIN cursos c ON c.id = m.id_curso
ORDER BY e.nombre;
```

Muestra cuántos cursos ha tomado cada estudiante.

```
--Muestra cuántos cursos ha tomado cada estudiante.
SELECT e.nombre, COUNT(m.id_curso) AS cantidad_cursos
FROM estudiantes e
LEFT JOIN matriculas m ON e.id = m.id_estudiante
GROUP BY e.nombre;
```

Calcula la edad actual de cada estudiante y ordénalos de mayor a menor edad.

```
--Calcula la edad actual de cada estudiante y ordénalos de mayor a menor edad.
SELECT nombre,
DATE_PART('year', AGE(fecha_nacimiento)) AS edad
FROM estudiantes
ORDER BY edad DESC;
```

Muestra qué curso tiene más estudiantes matriculados.

```
--Muestra qué curso tiene más estudiantes matriculados.
SELECT c.nombre_curso, COUNT(m.id_estudiante) AS total_estudiantes
FROM cursos c
JOIN matriculas m ON c.id = m.id_curso
GROUP BY c.nombre_curso
ORDER BY total_estudiantes DESC
LIMIT 1;
```

Calcula el porcentaje de estudiantes matriculados respecto al total de estudiantes para cada curso.

```
--Calcula el porcentaje de estudiantes matriculados respecto al total de estudiantes para cada curso.  
SELECT  
  c.nombre_curso,  
  COUNT(DISTINCT m.id_estudiante) AS estudiantes_matriculados,  
  ROUND(100.0 * COUNT(DISTINCT m.id_estudiante) / (SELECT COUNT(*) FROM estudiantes), 2) AS porcentaje  
FROM cursos c  
LEFT JOIN matriculas m ON c.id = m.id_curso  
GROUP BY c.nombre_curso;
```