**1. Responsabilidades del DBA Junior (Disponibilidad, Integridad, Seguridad)**

**Ejercicio 1: Análisis de rol del DBA Junior**  
**Consigna:**  
Lee el siguiente caso y responde:

Una empresa tiene un sistema de base de datos que almacena la información de clientes, productos y ventas. Un día, el sistema se cae por varias horas y se pierden algunos datos. Además, se descubrió que un empleado sin autorización accedió a información sensible de clientes.

**Preguntas y respuestas:**

1. **¿Qué responsabilidades del DBA Junior se vieron comprometidas?**
2. **¿Qué medidas se podrían haber tomado para garantizar la disponibilidad, integridad y seguridad de los datos?**

* **Disponibilidad**:
  + Tener sistemas de respaldo (back ups) y recuperación ante fallos.
  + Monitoreo constante del sistema.
  + Plan de contingencia ante caídas del servidor.
* **Integridad**:
  + Validar los datos al momento de ingresarlos.
  + Implementar restricciones en la base de datos (claves primarias, foráneas, tipos de datos).
  + Uso de transacciones para operaciones críticas.
* **Seguridad**:
  + Definir permisos de acceso según roles.
  + Autenticación y control de usuarios.
  + Encriptación de información sensible.

### ****¿Por qué es importante preservar la información como fuente objetiva para la toma de decisiones?****

Preservar la información permite mantener datos confiables, precisos y actualizados, lo que es esencial para tomar decisiones basadas en hechos. Si los datos se pierden o se alteran, las decisiones pueden ser erróneas, lo que afecta negativamente a la organización. La integridad de los datos garantiza que los informes generados reflejen la realidad y respalden una planificación eficiente y responsable.

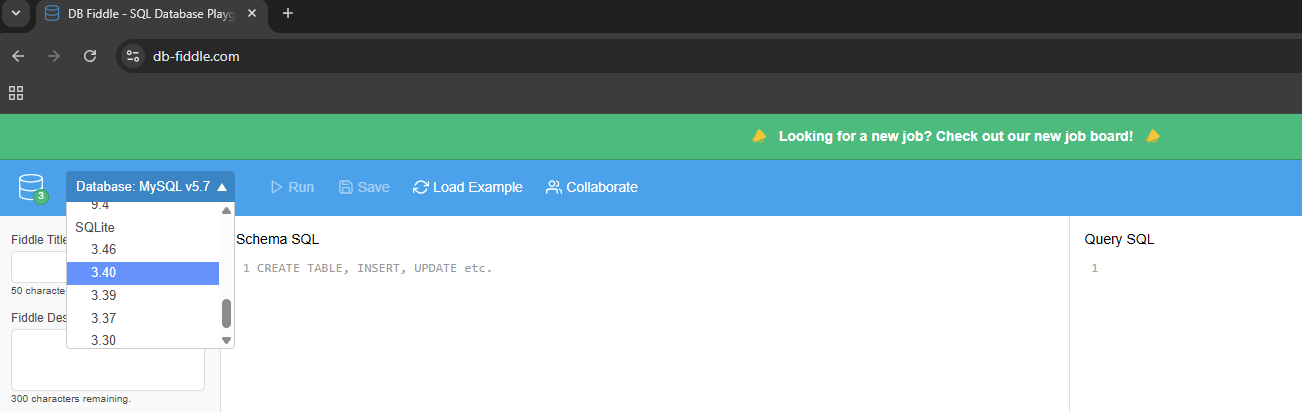
1. **SQL Básico: crear, modificar, eliminar y consultar datos**

**Usamos** [**https://www.db-fiddle.com/**](https://www.db-fiddle.com/)

Elegimos **SQLite versión 3.30**

Borramos todo lo que esté completo por defecto

**Se ve así:**

****

Borrar si hay código por defecto

En bases de datos relacionales, la **clave primaria (Primary Key)** identifica de manera única cada registro dentro de una tabla, asegurando que cada fila sea distinguible.

**La clave foránea (Foreign Key),** por otro lado, establece relaciones entre tablas, permitiendo que una tabla haga referencia a una fila en otra tabla usando su clave primaria.

**Clave Primaria (Primary Key):**

Es un atributo o un conjunto de atributos que **identifican de manera única** cada registro en una tabla.

Normalmente se utiliza para **identificar cada fila de manera única y evitar duplicados.**

**No puede tener valores nulos**.

Ejemplo: En una tabla de clientes, el ID del cliente podría ser la clave primaria.

**Clave Foránea (Foreign Key):**

Es un atributo o un conjunto de atributos en una tabla que **se utiliza para hacer referencia a la clave primaria de otra tabla.**

Permite establecer **relaciones entre tablas**, como "un cliente puede hacer varios pedidos".

En el ejemplo de clientes y pedidos, la tabla de pedidos podría tener una clave foránea que hace referencia al ID del cliente en la tabla de clientes.

**Ayuda a asegurar la integridad referencial**, garantizando que las referencias entre tablas sean válidas.

**En resumen**: La clave primaria identifica registros únicos dentro de una tabla, mientras que la clave foránea establece relaciones entre tablas, utilizando la clave primaria de la tabla relacionada para vincular los datos.

Supongamos la base de datos **Ventas2025** con tres tablas:

* Clientes(id, nombre, email)
* Productos(id, nombre, precio)
* Ventas(id, cliente\_id, producto\_id, cantidad, fecha)

**Ejercicio 2: Crear tablas**

**Consigna:**

1. Escribe los comandos para crear las tres tablas anteriores.
2. Agrega las claves primarias (PRIMARY KEY) que relacionen cliente\_id con Clientes(id) y producto\_id con Productos(id).

**Escritura en el simulador**

CREATE TABLE Consumidores (

id INT PRIMARY KEY,

nombre VARCHAR(50),

email VARCHAR(50)

);

CREATE TABLE Producción (

id INT PRIMARY KEY,

nombre VARCHAR(50),

precio DECIMAL(10,2)

);

CREATE TABLE Ventas\_25 (

id INT PRIMARY KEY,

consumidores\_id INT,

produccion\_id INT,

cantidad INT,

fecha DATE,

FOREIGN KEY (consumidores\_id) REFERENCES Consumidores(id),

FOREIGN KEY (produccion\_id) REFERENCES Produccion(id)

);

**Ejercicio 3: Insertar y modificar datos**

**Consigna:**

1. Agrega dos clientes más y dos productos.
2. Inserta al menos dos ventas más.
3. Modifica el email de "Ana Gómez".
4. Elimina un producto.

Escritura en el simulador/MySQL Workbench

-- Insertar algunos datos

INSERT INTO Consumidores VALUES (1, 'Ana Gómez', 'ana@mail.com');

INSERT INTO Consumidores VALUES (2, 'Carlos Pérez', 'carlos@mail.com');

INSERT INTO Consumidores VALUES (3, 'Lucía Díaz', 'lucia@mail.com');

INSERT INTO Produccion VALUES (1, 'Laptop', 950.00);

INSERT INTO Produccion VALUES (2, 'Mouse', 25.50);

INSERT INTO Produccion VALUES (3, 'Teclado', 45.00);

INSERT INTO Ventas\_25 VALUES (1, 1, 1, 2, '2025-04-10');

INSERT INTO Ventas\_25 VALUES (2, 2, 2, 1, '2025-04-11');

INSERT INTO Ventas\_25 VALUES (3, 3, 1, 3, '2025-04-12');

-- Modificar el email de Ana Gómez

UPDATE Consumidores

SET email = 'ana.gomez@mail.com'

WHERE nombre = 'Ana Gómez';

-- Eliminar un producccion

DELETE FROM Produccion

WHERE id = 3; -- Elimina el Teclado

**Ejercicio 4: Consultar Datos**

**Consigna:**

1. Haz una consulta para ver todas las ventas realizadas en abril 2025.
2. Muestra los nombres de los clientes que hayan comprado más de una unidad de un producto.

Escritura en el simulador

-- Ventas de abril 2025

SELECT \* FROM Ventas\_25

WHERE fecha BETWEEN '2025-04-01' AND '2025-04-30';

-- Clientes que hayan comprado más de una unidad

SELECT c.nombre, p.nombre AS producto, v.cantidad

FROM Ventas\_25 v

JOIN Consumidores c ON v.consumidores\_id = c.id

JOIN Productos p ON v.producto\_id = p.id

WHERE v.cantidad > 1;

1. **Tipos de bases de datos**

**Ejercicio 5: Comparación de tipos**

**Consigna:** Completa la siguiente tabla con tus propias palabras:

|  |  |  |  |
| --- | --- | --- | --- |

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipo de base de datos** | **¿Estructurada?** | **Ejemplo de uso** | **Ejemplo** |
| **Relacional** |  |  |  |
| **NoSQL** |  |  |  |
| **Distribuida** |  |  |  |
| **En memoria** |  |  |  |

1. **Ejemplo completo para mostrar resultados**

-- Crear tablas

CREATE TABLE Clientes (

id INT PRIMARY KEY,

nombre VARCHAR(50),

email VARCHAR(50)

);

CREATE TABLE Productos (

id INT PRIMARY KEY,

nombre VARCHAR(50),

precio DECIMAL(10,2)

);

CREATE TABLE Ventas (

id INT PRIMARY KEY,

cliente\_id INT,

producto\_id INT,

cantidad INT,

fecha DATE,

FOREIGN KEY (cliente\_id) REFERENCES Clientes(id),

FOREIGN KEY (producto\_id) REFERENCES Productos(id)

);

-- Insertar datos

INSERT INTO Clientes (id, nombre, email)

VALUES (1, 'Ana', 'ana@example.com'),

(2, 'Luis', 'luis@example.com');

INSERT INTO Productos (id, nombre, precio)

VALUES (1, 'Producto A', 100.50),

(2, 'Producto B', 200.00);

INSERT INTO Ventas (id, cliente\_id, producto\_id, cantidad, fecha)

VALUES (1, 1, 2, 3, '2025-04-17'),

(2, 2, 1, 1, '2025-04-16');

-- Consulta para mostrar resultados

SELECT v.id AS venta\_id,

c.nombre AS cliente,

p.nombre AS producto,

v.cantidad,

p.precio,

v.cantidad \* p.precio AS total,

v.fecha

FROM Ventas v

JOIN Clientes c ON v.cliente\_id = c.id

JOIN Productos p ON v.producto\_id = p.id;

1. **Modelo de Datos en PowerBI**

**Ejercicio 6: Relaciones entre tablas**

**Contexto:** Se cargaron tres tablas a Power BI: Clientes, Productos y Ventas (como en los ejercicios anteriores).

**Consigna:**

1. Establece las relaciones entre las tablas (usa los campos cliente\_id y producto\_id).
2. Identifica el tipo de relación: ¿uno a muchos o muchos a uno?
3. Explica con tus palabras por qué son necesarias las relaciones.