Guía Completa para Presentación de SkyRoute: Introducción + Código Explicado

---

Introducción

Hoy vamos a trabajar con un sistema llamado SkyRoute, diseñado para gestionar ventas de pasajes turísticos mediante una base de datos.

¿Qué es una base de datos?

Es un sistema donde guardamos y organizamos datos para poder usarlos fácilmente. Aquí usaremos SQL, un lenguaje para crear, modificar y consultar bases de datos.

Conceptos Clave

- Tabla: Contenedor de datos relacionados, como clientes o ventas.
- Registro: Una fila con información concreta dentro de una tabla.
- Clave Primaria (PK): Identifica cada registro de forma única.
- Clave Foránea (FK): Relaciona registros entre tablas, manteniendo la integridad de datos.

Ahora, vamos a crear la base de datos y las tablas con código, y explicar qué hace cada parte.

Paso 1: Crear la base de datos

```
CREATE DATABASE IF NOT EXISTS prog_apb;
USE prog_apb;
```

- CREATE DATABASE crea una nueva base llamada prog_apb solo si no existe para evitar errores.
- USE selecciona esa base para trabajar en ella.

Paso 2: Crear la tabla clientes

```
CREATE TABLE IF NOT EXISTS clientes (
    id INT             AUTO_INCREMENT PRIMARY KEY,
    razon_social       VARCHAR(100) NOT NULL,
    CUIT               VARCHAR(100) NOT NULL UNIQUE,
    email              VARCHAR(100) NOT NULL UNIQUE
);
```

- Crea la tabla clientes para guardar la información de los clientes.
- id es un número que se genera automáticamente y es único (clave primaria).
- razon_social, CUIT y email son datos del cliente. Los dos últimos no pueden repetirse (restricción UNIQUE).

Paso 3: Crear la tabla destinos

```
CREATE TABLE IF NOT EXISTS destinos (
    id INT             AUTO_INCREMENT PRIMARY KEY,
    ciudad             VARCHAR(100) NOT NULL,
    pais               VARCHAR(255) NOT NULL,
    costo_base         DECIMAL(10, 2) NOT NULL
);
```

- La tabla destinos guarda las ciudades y países a donde se puede viajar.
- costo_base es el precio base del pasaje para ese destino.

Paso 4: Crear la tabla ventas

```
CREATE TABLE IF NOT EXISTS ventas (
    id INT             AUTO_INCREMENT PRIMARY KEY,
    cliente_id         INT NOT NULL,
    destino_id         INT NOT NULL,
    fecha              DATETIME NOT NULL,
    monto              DECIMAL(10, 2) NOT NULL,
    estado             VARCHAR(50) DEFAULT 'Activa',
    FOREIGN KEY (cliente_id) REFERENCES clientes(id),
    FOREIGN KEY (destino_id) REFERENCES destinos(id)
);
```

- La tabla `ventas` registra cada venta hecha.
- `cliente_id` y `destino_id` son claves foráneas que vinculan las ventas con clientes y destinos existentes.
- `fecha` es la fecha y hora de la venta.
- `monto` es el precio pagado.
- `estado` indica si la venta está activa (valor por defecto 'Activa').

Paso 5: Insertar datos de ejemplo en las tablas

Insertar clientes

```
INSERT INTO clientes (razon_social, CUIT, email) VALUES
('Jorge Perez', '20-12345678-9', 'jperez@gmail.com'),
('Juan Lopez', '27-87654321-3', 'juan.lopex@gmail.com'),
('María García', '23-11223344-5', 'maria.g@gmail.com');
```

- Registramos tres clientes con datos únicos para probar.

Insertar destinos

```
INSERT INTO destinos (ciudad, pais, costo_base) VALUES
('París', 'Francia', 1200.50),
('Tokio', 'Japón', 1850.00),
('Río de Janeiro', 'Brasil', 750.25),
('Salta', 'Argentina', 400.00);
```

- Añadimos cuatro destinos disponibles para viajar.

Insertar ventas

```
INSERT INTO ventas (cliente_id, destino_id, fecha, monto) VALUES
(1, 1, '2025-06-01 10:00:00', 1200.50),
(2, 2, '2025-06-01 12:00:00', 1850.00),
(3, 3, '2025-06-02 14:30:00', 750.25);
```

- Registramos tres ventas, cada una vinculando un cliente con un destino.

Paso 6: Consultas para extraer información

Aquí algunos ejemplos que ayudan a entender y analizar los datos.

Listar todos los clientes

```
SELECT * FROM clientes;
```

Muestra todos los clientes registrados.

Mostrar ventas en una fecha específica

```
SELECT * FROM ventas
WHERE DATE(fecha) = '2025-06-01';
```

Filtra ventas hechas el 1 de junio de 2025.

Última venta por cliente

```
SELECT c.id AS cliente_id, c.razon_social, MAX(v.fecha) AS ultima_venta
FROM clientes c
JOIN ventas v ON c.id = v.cliente_id
GROUP BY c.id, c.razon_social;
```

Muestra la última compra hecha por cada cliente.

Destinos que empiezan con "S"

```
SELECT * FROM destinos
WHERE ciudad LIKE 'S%';
```

Filtra destinos cuya ciudad comienza con la letra "S".

Cantidad de ventas por país

```
SELECT d.pais, COUNT(*) AS cantidad_ventas
FROM ventas v
JOIN destinos d ON v.destino_id = d.id
GROUP BY d.pais;
```

Cuenta cuántas ventas se hicieron para cada país.

Total recaudado por cliente (ventas activas)

```
SELECT c.razon_social, SUM(v.monto) AS total_recaudado
FROM clientes c
JOIN ventas v ON c.id = v.cliente_id
WHERE v.estado = 'Activa'
GROUP BY c.razon_social;
```

Suma el total pagado por cada cliente considerando solo ventas activas.

Conclusión

Este sistema básico muestra cómo organizar información de clientes, destinos y ventas con buenas prácticas de bases de datos relacionales. Además, permite hacer consultas útiles para la gestión y análisis de datos.

A continuación le dejo el link de descarga del SQL prog_abp :

https://drive.google.com/file/d/1GGXlhLzsGiK74-LSukMGSymcpLtEBxLp/view?usp=sharing