REFERENCIA RÁPIDA DE SQL

SELECT, JOIN, subconsultas, índices, transacciones

SELECT

```
SELECT * FROM users;
SELECT name, email FROM users;
SELECT DISTINCT city FROM users;
SELECT name AS full_name FROM users;
```

WHERE

Operadores de Comparación

`= <> (!=)`	Igual / diferente
`< > <= >=`	Operadores de comparación
`AND OR NOT`	Operadores lógicos
`IS NULL / IS NOT NULL`	Verificación de nulos

Coincidencia de Patrones

```
SELECT * FROM users WHERE name LIKE 'A%';
-- % = cualquier carácter, _ = carácter único
SELECT * FROM users WHERE age IN (20, 25, 30);
SELECT * FROM users WHERE age BETWEEN 18 AND 30;
```

JOIN

Tipos de JOIN

`INNER JOIN`	Filas coincidentes en ambas tablas
`LEFT JOIN`	Todas las filas izquierdas + coincidencias derechas
`RIGHT JOIN`	Todas las filas derechas + coincidencias izquierdas
`FULL OUTER JOIN`	Todas las filas de ambas tablas
`CROSS JOIN`	Producto cartesiano de ambas tablas

Sintaxis de JOIN

```
SELECT u.name, o.total
FROM users u
INNER JOIN orders o ON u.id = o.user_id;

SELECT u.name, o.total
FROM users u
LEFT JOIN orders o ON u.id = o.user_id;
```

INSERT / UPDATE / DELETE

Insertar

```
INSERT INTO users (name, email)
VALUES ('Alice', 'alice@example.com');

INSERT INTO users (name, email) VALUES
('Bob', 'bob@ex.com'),
('Carol', 'carol@ex.com');
```

Actualizar

```
UPDATE users SET email = 'new@ex.com'
WHERE id = 1;
```

Eliminar

```
DELETE FROM users WHERE id = 1;
DELETE FROM users; -- eliminar todas las filas
```

CREATE TABLE

Sintaxis

```
CREATE TABLE users (
  id    INTEGER PRIMARY KEY AUTOINCREMENT,
  name  TEXT NOT NULL,
  email TEXT UNIQUE,
  age   INTEGER DEFAULT 0,
  score REAL
);
```

Tipos de Datos Comunes

`INTEGER`	Números enteros
`REAL`	Números de punto flotante
`TEXT`	Cadena de texto
`BLOB`	Datos binarios
`BOOLEAN`	TRUE / FALSE (almacenado como 0/1)
`DATE / DATETIME`	Valores de fecha y marca de tiempo

Restricciones

`PRIMARY KEY`	Identificador único de fila
`NOT NULL`	Valor obligatorio
`UNIQUE`	Sin valores duplicados
`DEFAULT val`	Valor por defecto si se omite
`CHECK (expr)`	Regla de validación personalizada
`FOREIGN KEY`	Referencia a otra tabla

Funciones de Agregación

`COUNT(*)`	Número de filas
`COUNT(col)`	Valores no nulos en la columna
`SUM(col)`	Suma de la columna numérica
`AVG(col)`	Promedio de la columna numérica
`MIN(col)`	Valor mínimo
`MAX(col)`	Valor máximo

Ejemplo

```
SELECT COUNT(*) AS total,
       AVG(age) AS avg_age,
       MAX(score) AS top_score
FROM users;
```

GROUP BY / HAVING

```
SELECT city, COUNT(*) AS num_users
FROM users
GROUP BY city;

SELECT city, AVG(age) AS avg_age
FROM users
GROUP BY city
HAVING AVG(age) > 25;
```

WHERE filtra filas antes de agrupar; HAVING filtra grupos después de la agregación

ORDER BY / LIMIT

```
SELECT * FROM users ORDER BY name ASC;
SELECT * FROM users ORDER BY age DESC;
SELECT * FROM users
ORDER BY city, name
LIMIT 10 OFFSET 20;  -- saltar 20, tomar 10
```

Subconsultas

En la Cláusula WHERE

```
SELECT name FROM users
WHERE id IN (
  SELECT user_id FROM orders
  WHERE total > 100
);
```

Como Tabla Derivada

```
SELECT city, avg_age FROM (
  SELECT city, AVG(age) AS avg_age
  FROM users GROUP BY city
) WHERE avg_age > 30;
```

Índices

```
CREATE INDEX idx_name ON users(name);
CREATE UNIQUE INDEX idx_email
  ON users(email);
DROP INDEX idx_name;
```

Cuándo Indexar

`Columnas en WHERE`	Acelera el filtrado
`Columnas en JOIN ON`	Acelera las búsquedas de JOIN
`Columnas en ORDER BY`	Acelera el ordenamiento
`Columnas de alta cardinalidad`	Muchos valores únicos se benefician más