Informe detallado sobre SQL Estándar

# Introducción

SQL (Structured Query Language) es un lenguaje de consulta estructurado diseñado para gestionar bases de datos relacionales.

El SQL estándar es definido por el ANSI (American National Standards Institute) y el ISO (International Organization for Standardization),

y establece una serie de normas que los distintos sistemas gestores de bases de datos (DBMS) como PostgreSQL, MySQL, Oracle, SQL Server, entre otros, intentan seguir.

Las versiones principales del estándar incluyen:

- SQL-86 (SQL-87): primera versión aprobada.

- SQL-89 (SQL1): correcciones menores.

- SQL-92 (SQL2): mejora significativa del lenguaje.

- SQL:1999 (SQL3): incluye programación procedural (SQL/PSM), disparadores, objetos.

- SQL:2003: XML, funciones escalares definidas por el usuario.

- SQL:2008: mejoras en MERGE, TRUNCATE, diagnósticos.

- SQL:2011: soporte para datos temporales.

- SQL:2016/2019: JSON, polimorfismo, nuevas funciones de análisis.

# 1. Estructura General de SQL

SQL se divide en varios sublenguajes:

1.1 DDL – Data Definition Language

Define estructuras de datos (tablas, vistas, esquemas, etc.)

- CREATE, ALTER, DROP

Ejemplo:

CREATE TABLE empleados (

id INT PRIMARY KEY,

nombre VARCHAR(100),

salario DECIMAL(10,2),

fecha\_ingreso DATE

);

1.2 DML – Data Manipulation Language

Maneja los datos dentro de las estructuras.

- INSERT, UPDATE, DELETE

Ejemplo:

INSERT INTO empleados (id, nombre, salario, fecha\_ingreso)

VALUES (1, 'Ana García', 35000.00, '2022-01-15');

1.3 DQL – Data Query Language

Consulta información.

- SELECT

Ejemplo:

SELECT nombre, salario FROM empleados WHERE salario > 30000;

1.4 DCL – Data Control Language

Controla permisos y accesos.

- GRANT, REVOKE

Ejemplo:

GRANT SELECT, INSERT ON empleados TO usuario1;

1.5 TCL – Transaction Control Language

Control de transacciones.

- BEGIN, COMMIT, ROLLBACK

Ejemplo:

BEGIN;

UPDATE empleados SET salario = salario \* 1.10;

COMMIT;

# 2. Características del SQL Estándar

2.1 Independencia de Plataforma

SQL está diseñado para ser portable, aunque cada sistema puede tener extensiones propietarias.

2.2 Declarativo

SQL se basa en describir qué se quiere, no cómo obtenerlo.

2.3 Integridad Referencial

Uso de claves foráneas y restricciones para mantener relaciones válidas entre tablas.

Ejemplo:

CREATE TABLE departamentos (

id INT PRIMARY KEY,

nombre VARCHAR(100)

);

ALTER TABLE empleados ADD CONSTRAINT fk\_departamento

FOREIGN KEY (id\_departamento) REFERENCES departamentos(id);

2.4 Funciones y Operadores Estándar

- Funciones escalares: UPPER(), LOWER(), LENGTH()

- Agregación: SUM(), AVG(), COUNT()

- Fecha y hora: CURRENT\_DATE, EXTRACT(), NOW()

2.5 Subconsultas y Joins

JOIN:

SELECT e.nombre, d.nombre FROM empleados e

JOIN departamentos d ON e.id\_departamento = d.id;

Subconsulta:

SELECT nombre FROM empleados

WHERE salario > (SELECT AVG(salario) FROM empleados);

# 3. Características avanzadas del SQL Estándar

3.1 Vistas (Views)

CREATE VIEW vista\_empleados\_altos AS

SELECT nombre, salario FROM empleados WHERE salario > 40000;

3.2 Procedimientos y Funciones (SQL/PSM)

CREATE PROCEDURE aumento\_sueldo (IN porcentaje DECIMAL)

BEGIN

UPDATE empleados SET salario = salario \* (1 + porcentaje/100);

END;

3.3 Disparadores (Triggers)

CREATE TRIGGER log\_insert\_empleado

AFTER INSERT ON empleados

FOR EACH ROW

BEGIN

INSERT INTO log\_cambios (descripcion)

VALUES ('Nuevo empleado añadido: ' || NEW.nombre);

END;

3.4 Tipos de datos definidos por el usuario

CREATE DOMAIN correo\_electronico AS VARCHAR(255)

CHECK (VALUE LIKE '%@%.%');

3.5 JSON (desde SQL:2016)

SELECT JSON\_OBJECT('nombre': nombre, 'salario': salario)

FROM empleados;

# 4. Ejemplo completo de base de datos

CREATE TABLE departamentos (

id INT PRIMARY KEY,

nombre VARCHAR(50)

);

CREATE TABLE empleados (

id INT PRIMARY KEY,

nombre VARCHAR(100),

salario DECIMAL(10,2),

id\_departamento INT,

FOREIGN KEY (id\_departamento) REFERENCES departamentos(id)

);

INSERT INTO departamentos VALUES (1, 'Finanzas'), (2, 'TI');

INSERT INTO empleados VALUES

(1, 'Carlos López', 30000, 2),

(2, 'Marta Ruiz', 42000, 1);

SELECT e.nombre, d.nombre AS departamento

FROM empleados e

JOIN departamentos d ON e.id\_departamento = d.id;

# Conclusión

El SQL estándar proporciona una base sólida, coherente y extensible para el manejo de bases de datos relacionales.

Aunque cada sistema gestor puede incluir extensiones, seguir el estándar garantiza portabilidad y mantenibilidad.