Introducción Básica SQL

Base de Datos

Esteban Schab – Pablo Pescio

Introducción

- **SQL** (*Structured Query Language*) lenguaje de consulta estructurado.
- Toma como base los lenguajes de consultas propuestos por E.F.CODD en 1970 Álgebra Relacional y Cálculo Relacional.
- El primer prototipo fue desarrollado por IBM en 1986 para su producto IBM research y SYSTEM R, denominado SEQUEL.
- Es el principal motivo del éxito de las bases de datos relacionales.

Estándar SQL

- Fue adoptado por los DBMS.
- En 1986 se creó el estándar por ANSI (American National Standart Institute) y luego adoptado por ISO (International Standart Organization).
- Se denominó SQL-86 ó SQL-1.
- En 1992 se crea SQL-2 y posteriormente SQL-3.

Estándar SQL

- Permite migrar entre distintos DBMS.
- Si bien los DBMS o SGBD incorporan varias de las definiciones del estándar existen algunas sentencias no soportadas.
- Posee extensiones para conexiones con leguajes de programación como C++, Java, Python, entre otros.

SQL

SQL está dividido en dos grandes partes, ambas ya definidas con anterioridad en esta materia:

. **Lenguaje de definición de datos DDL.**

. **Lenguaje de manipulación de datos DML.**

SQL cuestiones previas

- Si bien se basa en las definiciones formales del modelo relacional, tiene distintas denominaciones, ej:
  - A la relación la denomina *table* de tabla
  - A los atributos los llama *Column* o columna

DML

- Sentencias principales:
  - **CREATE**
  - **DROP**
  - **ALTER**

SQL

. Lenguaje de definición de datos DDL.

. Lenguaje de manipulación de datos DML.

DATABASE

- **Crear una base de datos**
  - **CREATE DATABASE *nombredb;***

  Ej: CREATE DATABASE facultad;

  También se puede utilizar la sentencia SCHEMA.
- **Borrar una base de datos**
  - **DROP DATABASE *nombredb;***

  Ej.: DROP DATABASE facultad;

TABLE

CREATE

```
CREATE TABLE alumno(
dni             integer not null,
apellido        varchar(30),
nombres         varchar(40),
tipodni         char not null,
PRIMARY KEY dni,
FOREIGN KEY (tipodni)
REFERENCES tipo(tipodni)
        ON DELETE restrict,
        ON UPDATE cascade);
```

TABLE

- **Eliminar**
  - DROP TABLE alumno;
- **Modificar**
- ALTER TABLE alumno (--)
  - ADD COLUMN mail varchar (50);
  - DROP COLUMN nombre;
  - ALTER COLUMN nombre varchar text;

SQL

. Lenguaje de definición de datos DDL.

. **Lenguaje de manipulación de datos DML.**

Tipos de datos de SQL-92

- **Numéricos**
  - Integer, smallint, float, real, decimal(i,j), numeric(i,j)
- **Caracteres**
  - Char, char(n), varchar(n) char varying (n)
- **Bits**
  - Bit(n), bit varying(n)
- **Boolean**
- **Date**
  - Date, time, time with time zone, timestamp

DML

- Consultas, estructura básica
- Funciones de agregación
- Funciones de agrupación
- Consultas anidadas
- Productos JOIN
- Operaciones de conjuntos
- Operadores especiales
- ABM (insert – delete – update)

Consultas, estructura básica

SELECT *lista de atributos*
FROM *nombre de tabla*
WHERE *predicado* (condición)
Ej.
SELECT dni, apellido, nombre
FROM estudiante
WHERE dni > 30000000

CONSULTAS BÁSICAS

- WHERE puede no aparecer en una consulta.
- Ej.
- SELECT isbn, titulo, editorial
- FROM libro

Relación con A.R.

SELECT *atrib1, atrib2* → PROJECT

FROM *tabla* → nombre de relación

WHERE *predicado* → SELECT de A.R.

Ej.

SELECT dni, apellido

FROM alumno, materia

WHERE alumno.dni = materia.idal

π dni,apellido σ(alumno.dni = materia.idal) (alumno $\mathbf{x}$ materia)

SELECT, opciones

SELECT

a, b (lista de atributos)

* (devuelve todos los atributos)

DISTINCT (por defecto SELECT devuelve ALL, todas las tuplas, sin eleminar los duplicados. Si no queremos los duplicados tenemos que indicarle con DISTINCT)

dni AS documento, apell AS apellido (AS se utiliza para renombrar los atributos en la tabla de salida, NO les cambia el nombre en la tabla original).

(sueldo * 1.10) AS aumento

FROM

FROM

- alumno

- alumno a, materia m, carrera c (se pueden renombrar las tablas, esto sirve, como lo veremos en un ejemplo para hacer más sencilla la redacción de las consultas) En estos casos tanto en el Select como en el Where se utiliza por ejemplo a.dni, m.id, c.nombre, nombre de tabla“.”atributo.

WHERE

WHERE

- sueldo >= 5000 (se pueden utilizar operaciones permitidas para el tipo de dominio de cada atributo).
- alumno.dni = materia.dni AND alumno.legajo = 25054 (se pueden utilizar los operadores AND, OR para filtrar por más de una condición)

BETWEEN

- BETWEEN se utiliza para seleccionar tuplas entre dos valores.
- Va siempre luego del WHERE
- WHERE atributo BETWEEN valor1 AND valor2;
- Ej.

```
SELECT apellido, nombre
FROM empleado
WHERE sueldo BETWEEN 10000 AND 15000;
```

LIKE

- **LIKE** se utiliza para comparación de cadenas ***%*** _

  SELECT … FROM …

  WHERE **LIKE** nombre *like* ‘D%’

  todo los nombres que comienzan con D

  S… F… WHERE **LIKE** cuit ’27_ _ _ _ _ _ _ _ _ _’ todo los cuit que comiencen con 27 y tengan 9 lugares luego.

  % reemplaza cualquier número de susbtring

  _ reemplaza una sóla posición.

  S… F… WHERE **LIKE** nombre ‘%cor%’ contenga ‘cor’ en el medio.

ORDER BY

- **ORDER BY** le da un orden al conjunto resultado.
- SELECT dni, apellido, nombre, cp
  FROM estudiante
  WHERE cp = 3260
  ORDER BY apellido, nombre;
  DESC forma descendente
  ASC ascendente (por defecto)

Ejemplos

- Usuario (dni, genero, apellido, nombre, cp)
- Ciudad (cp, nombre)
- Prestamo (dni, idlibro,fechaprestamo, fechadevolucion)
- 1) listar los usuarios de Concordia
- 2) listar las usuarias que hayan retirado algún libro.
- 3) listar los libros que no fueron devueltos

AGREGACIÓN

- Funciones de agregación
- MAX () máximo
- MIN () mínimo
- AVG () promedio
- COUNT () cuenta
- SUM () suma

Agregación ejemplos

- SELECT MAX(sueldo), MIN (sueldo) SUM (sueldo), AVG (sueldo)

  FROM empleado e, departamento d

  WHERE e.dpto = d.dpto AND d.nombre = ‘SISTEMAS’;
- SELECT COUNT (*)

  FROM alumno a, inscripción i

  WHERE a.dni = i.dni AND i.anio = 2016;

Funciones de gupo

- **GROUP BY** (agrupa el resultado)
  SELECT apellido, nombre
  FROM empleado, departamento
  GROUP BY departamento.id
  HAVING (condición de grupo)
 SELECT count (*)
  FROM empleado, departamento
  GROUP BY departamento.id
  HAVING sueldo > 15000;

CONJUNTOS - UNIÓN

- Ejemplo, todas las materias para las cuales RIQUELME este anota como alumno ó como docente

(SELECT nombrematería

FROM alumno a, inscripto i, materia m

WHERE a.dni = i.dni AND i.mat = m.mat AND a.apellido = 'RIQUELME')

UNION

(SELECT nombrematería

FROM docente d, dicta i, materia m

WHERE d.dni = i.dni AND i.mat = m.mat AND a.apellido = 'RIQUELME')

UNION devuelve una sola vez las tuplas duplicadas.

Las otras funciones de conjunto con INTERSECT y EXCEPT (diferencia)

RELAX!

SUBCONSULTAS DE CONJUNTOS

- **IN**
- **NOT IN**
- **ALL**
- **SOME**
- **EXIST**
- **NOT EXIST**

IN – NOT IN

- Ejemplo, lista los alumnos que participan en proyecto cuya duración es mayor o igual a 24 meses.

```
SELECT apellido, nombre
FORM alumno
WHERE idproyecto IN (
SELECT idproyecto
FROM proyecto
WHERE duracion >= 24)
```

Se utiliza NOT IN para devolver los NO incluidos en ciertos gurpos.

SOME - ALL

- **=SOME** significa igual a alguno
- **>ALL** mayor que todos
- SELECT apellido, nombre, sueldo

  FROM docente

  WHERE sueldo > SOME (SELECT

  sueldo FROM docente d, departamento dto
  WHERE d.numdto = dto.numdto AND
  dto.nombre = ‘COMPUTACION’)

  ¿Qué devuelve el ejemplo anterior?

EXIST

- **EXIST** prueba la existencia de tuplas en una subconsulta
- Ej. Apellido de los usuario que hayan retirado algún libro cuyo título contenga la palabra ‘química’.

```
SELECT apellido
FROM usuario
WHERE EXIST (SELECT dni
FROM usuario u, prestamo p, libro l
WHERE u.dni = p.dni AND p.isbn = l.sbn AND l.titulo like ‘%QUIMICA%’)
```

NULOS

- **NULL – NOT NULL**
- Se utilizan NULL o NOT NULL para probar si un atributo es nulo o no lo és.
- Ej.
- SELECT apellido, nombre
  FROM empleado
  WHERE categoria IS NULL;

PRODUCTOS NATURALES JOIN

- **INNER JOIN**
- Ej.

  SELECT apellido, nombciu
  FROM alumno a **INNER JOIN** ciudad c
  ON (a.cp = c.cp)

  LEFT JOIN – RIGHT JOIN – FULL JOIN

  devuelven todas las tuplas de la relación de la izquierda, de la relación de la derecha o de ambas, aunque no coincidan.

INSERT

- INSERT INTO tabla (lista-de-atributos) VALUES (valores-para-cada-atributo)
- Si se omite la lista-de-atributos se toma como está definida la tabla, en ese orden.
- Ej.
- INSERT INTO auto (id, marca, modelo, combustible, puertas) VALUES (500, ‘AUDI’, ‘A4’,’NAFTA’,4);

INSERT

- Se pueden insertar valores desde otra tabla.
- Ej.
- INSERT into buenproveedor (id, nombre) VALUES (SELECT id,nombre FROM proveedor p, entregas e WHERE p.id = e.id AND e.fechaprev <= e.fechaefectiva);

DELETE

- DELETE FROM tabla WHERE (condición)
- Ej. 1

  DELETE FROM alumno WHERE idcarrera=3;
- Ej. 2

  DELETE FROM auto;

  ¿qué hace el ejemplo 2?

UPDATE

- UPDATE tabla
- SET atributo-que-se-modifica
- WHERE condición (opcional)
- Ej.

  UPDATE producto

  SET precio = precio * 1.20

  WHERE tipo = ‘PERFUMERIA’;

PREGUNTAS