



Diseño de Bases de Datos

Clase 6

Curso 2020

Prof. Luciano Marrero

Pablo Thomas

Rodolfo Bertone

Agenda

Lenguaje de Consultas Estructurado (SQL)

- Definiciones
- DDL / DML
- Consultas/Updates
- Ejemplos

Lenguaje de Consultas Estructurado (SQL)

historia

1986 Es un estándar ANSI

1992 se amplia el estándar (SQL2 o SQL 92)

1999 Se crea SQL 2000 incorporando expresiones regulares, consultas recursivas y características de OO

2003 Surge SQL 3 agrega características de XML

2006 se definen características que lo acercan al mundo W3C

Lenguaje de definición de datos

- Es muy amplio
- Solo veremos las operaciones mas comunes
- CREATE DATABASE
- DROP DATABASE
- CREATE TABLE
- ALTER TABLE
- DROP TABLE

SQL → DDL

► Ejemplos de Creación de Tablas

```
CREATE TABLE empresas ( idempresa INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
                          empresa VARCHAR(100) NOT NULL,  
                          abreviatura VARCHAR(10) NULL,  
                          cuit VARCHAR(13) NULL,  
                          direccion VARCHAR(100) NULL,  
                          observaciones TEXT NULL,  
                          PRIMARY KEY(idempresa),  
                          UNIQUE INDEX empresas_index19108(empresa));
```

SQL → DDL

► Ejemplos de Creación de Tablas

```
CREATE TABLE pacientesempresas
( idpacienteempresa INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  idpaciente_os INTEGER UNSIGNED NOT NULL,
  idempresa INTEGER UNSIGNED NOT NULL,
  fecha_desde DATE NOT NULL,
  fecha_hasta DATE NULL,
  PRIMARY KEY(idpacienteempresa),
  INDEX empleadosempresas_FKIndex1(idempresa),
  INDEX pacientesempresas_FKIndex2(idpaciente_os),
  FOREIGN KEY(idempresa) REFERENCES empresas(idempresa)
    ON DELETE RESTRICT
    ON UPDATE NO ACTION,
  FOREIGN KEY(idpaciente_os) REFERENCES pacientes_os(idpaciente_os)
    ON DELETE RESTRICT
    ON UPDATE NO ACTION);
```

SQL → DDL

➤ Ejemplo de modificación de tablas

```
ALTER TABLE empresas (  
  ADD COLUMN razon_social VARCHAR(100) NOT NULL,  
  DROP COLUMN cuit,  
  ALTER COLUMN direccion VARCHAR(50) NULL);
```


SQL → DML

➤ Estructura básica

```
SELECT lista_de_atributos  
FROM lista_de_tablas  
WHERE condición
```

```
SELECT atr1, atr2, atr3  
FROM tabla1, tabla2  
WHERE atr4="Valor"
```

Equivale a

$$\pi_{\text{atr1, atr2, atr3}} (\sigma_{\text{atr4} = \text{'valor'}} (\text{tabla1} \times \text{tabla2}))$$

SQL → DML

Tablas sobre las cuales se resolverán las consultas

- Asociados=(idsocio, nombre, dirección, teléfono, sexo, estadocivil, fechanacimiento, idlocalidad)
- Deportes=(iddeporte, nombre, monto_cuota, idsede)
- Practica = (idsocio, iddeporte)
- Localidad =(idlocalidad, nombre)
- Sedes = (idsede, nombre, dirección, idlocalidad)

SQL → DML

- Ejemplo 1: nombre de todos los asociados

```
SELECT nombre  
FROM asociados
```

- La Cláusula Select puede contener
 - * (incluye todos los atributos de las tablas que aparecen en el from)
 - Distinct (eliminan tuplas duplicadas)
 - All (valor por defecto, aparecen todas las tuplas)

SQL → DML

- Ejemplo 2: mostrar todos los datos de los asociados

SELECT *

FROM asociados

- Ejemplo 3: mostrar todos los id de localidades donde viven los asociados, sin repetir valores

SELECT DISTINCT(idlocalidad)

FROM asociados

SQL → DML

- Ejemplo 4: mostrar de cuanto sería la cuota de cada deporte si se incrementara en un 25%

```
SELECT montocuota*1.25  
FROM deportes
```

SQL → DML

- Ejemplo 4_1: suponga que la tabla PERSONA (idpersona, nombre, apellido, direccion, telefono)
Mostrar el nombre y apellido de todas las personas

```
SELECT nombre, apellido  
FROM persona
```

Nombre	Apellido
Juan	Perez
Pedro	Gomez

```
SELECT apellido, nombre  
FROM persona
```

Apellido	Nombre
Perez	Juan
Gomez	Pedro

```
SELECT CONCAT(RTRIM(apellido),",", " ", RTRIM(nombre) )  
FROM persona
```

CONCAT(RTRIM(apellido),",", " ", RTRIM(nombre))
Perez, Juan
Gomez, Pedro

SQL → DML

➤ Cláusula WHERE

- Ejemplo 5: **Operadores lógicos:** mostrar los asociados varones casados.

SELECT nombre

FROM asociados

WHERE sexo = "masculino" AND estadocivil = "casado"

SQL → DML

➤ Cláusula WHERE

- Ejemplo 6: **Operador BETWEEN**: mostrar los deportes cuya cuota esté entre 400 y 600 pesos mensuales

```
SELECT nombre  
FROM deportes  
WHERE montocuota >= 400 AND montocuota <= 600
```

```
SELECT nombre  
FROM deportes  
WHERE montocuota BETWEEN 400 and 600
```


SQL → DML

➤ Cláusula FROM

- Ejemplo 7: **Producto Cartesiano**: mostrar para cada asociado su nombre y la localidad de residencia

```
SELECT asociado.nombre, localidad.nombre  
FROM asociado, localidad  
WHERE asociado.idlocalidad = localidad.idlocalidad
```

SQL → DML

➤ Cláusula FROM

- Ejemplo 8: **Producto Natural** mostrar para cada asociado su nombre y la localidad de residencia

```
SELECT asociado.nombre, localidad.nombre  
FROM asociado INNER JOIN localidad ON (asociado.idlocalidad=localidad.idlocalidad)
```

SQL → DML

➤ Cláusula FROM

```
SELECT asociado.nombre, localidad.nombre  
FROM asociado, localidad  
WHERE asociado.idlocalidad = localidad.idlocalidad
```



```
SELECT asociado.nombre, localidad.nombre  
FROM asociado INNER JOIN localidad ON (asociado.idlocalidad=localidad.idlocalidad)
```

SQL → DML

➤ Operación de Renombrar

- Ejemplo 9: mostrar todos los deportes salvo el mas costoso.

```
SELECT DISTINCT deporte.nombre
```

```
FROM deportes, deporte dep
```

```
WHERE dep. Montocuota > deporte.montocuota
```

- Ejemplo 10: presentar todos los asociados con nombre, dirección y idlocalidad. El listado debe figurar con la leyenda DIRECCIÓN LEGAL.

```
SELECT nombre, direccion AS DIRECCION LEGAL, idlocalidad
```

```
FROM asociados
```

Nombre	Cuota
Futbol	1000
Basquet	2000
Tenis	3000

Dep.nombre	Dep.monto	Deporte.nombre	Deporte.monto
Futbol	1000	Futbol	1000
Futbol	1000	Basquet	2000
Futbol	1000	Tenis	3000
Basquet	2000	Futbol	1000
Basquet	2000	Basquet	2000
Basquet	2000	Tenis	3000
Tenis	3000	Futbol	1000
Tenis	3000	Basquet	2000
Tenis	3000	Tenis	3000

Nombre	DIRECCION LEGAL	idLocalidad
Juan perez	Cerrito 123	1
Jose Gomez	Chacabuco 232	3

SQL → DML

➤ Renombre, otro uso

➤ Ejemplo 10_1: mostrar los asociados y los deportes que practica

SELECT a.nombre, d.nombre

FROM asociados a INNER JOIN practica p ON (a.idsocio = p.idsocio)

INNER JOIN deportes d ON (d.iddeporte = p.iddeporte)

SELECT a.nombre, d.nombre

FROM asociados a, practica p, deportes d

WHERE a.idsocio = p.idsocio AND p.iddeporte = d.iddeporte

SQL → DML

➤ Operaciones sobre cadenas

➤ Operador **Like**: %, _

- Ejemplo 11: mostrar aquellos asociados cuyo nombre empiece con RA

```
SELECT nombre  
FROM Asociados  
WHERE nombre = "RA"
```

```
SELECT nombre  
FROM asociados  
WHERE nombre like "RA%"
```

- **"Alfa%"**: cualquier cadena que empiece con Alfa
- **"%casa%"**: cualquier cadena que tenga casa en su interior
- **"_ _ _"**: cualquier cadena con tres caracteres
- **"_ _ _%"**: cualquier cadena con al menos tres caracteres.

```
SELECT nombre  
FROM deportes  
WHERE nombre like "%BALL%"
```

- Ejemplo 12: mostrar los deportes que incluyan la palabra BALL en su nombre
- Ejemplo 12_1: mostrar los asociados que tengan una letra C como tercera letra de su nombre

```
SELECT nombre  
FROM asociados  
WHERE nombre like "__c%"
```

SQL → DML

- Operadores que permite **ORDENAR** las tuplas

- **ORDER BY** *atributo*:

- especifica el atributo por el cual las tuplas serán ordenadas
 - Ejemplo 13: presentar todos los asociados ordenados por nombre.

```
SELECT nombre  
FROM asociados  
ORDER BY nombre
```

- **Desc, asc**: por defecto ascendente, se puede especificar descendente.

SQL → DML

- Ejemplo 14 : presentar los asociados que nacieron el mes de agosto ordenados por fecha desde el 31 al 1 de agosto.

SELECT nombre

FROM asociados

WHERE MONTH(fecha_nacimiento) = 8

ORDER BY DAY(fecha_nacimiento) DESC

■ Fechas???

- Crisis del año 2000
- Cambio en el manejo de fechas
- Funciones de fechas?? YEAR MONTH DAY DATE TIME etc.

SQL → DML

24

Otras operaciones del algebra, Operaciones sobre conjuntos

- Unión: agrupa las tuplas resultantes de dos subconsultas.
- Union all conserva duplicados
 - Ejemplo 15: asociados que practican futbol o voley
- Intersección:
 - Ejemplo 16: asociados que practican futbol y voley
- Diferencia: (except)
 - Ejemplo 17: asociados que p

```
(SELECT a.nombre
FROM  asociados a INNER JOIN practica p ON a.idsocio = p.idsocio
      INNER JOIN deportes d ON p.iddeporte = d.iddeporte
WHERE d.nombre = "FUTBOL" )
UNION / UNION ALL / INTERSECT / EXCEPT
(SELECT a.nombre
FROM  asociados a INNER JOIN practica p ON a.idsocio = p.idsocio
      INNER JOIN deportes d ON p.iddeporte = d.iddeporte
WHERE d.nombre = "VOLEY" )
```

SQL → DML

- Funciones de agregación:
 - Promedio (avg): aplicable a atributos numéricos, retorna el promedio de la cuenta
 - Mínimo (min): retorna el elemento más chico dentro de las tuplas para ese atributo
 - Máximo (max): retorna el elemento más grande dentro de las tuplas para ese atributo
 - Total (sum): aplicable a atributos numéricos, realiza la suma matemática
 - Cuenta (count): cuenta las tuplas resultantes.

SQL → DML

► Funciones de agregación:

► Limitaciones

- Debe aparecer en el select (por ahora)
- NO puede aparecer en el where
- Devuelve un solo valor, por ende POR AHORA, no puede haber otro atributo en el select si hay una funcion de agregacion.

► Ejemplo 18: mostrar la cantidad de asociados del club

SELECT COUNT (*)

FROM asociados

SQL → DML

- Ejemplo 19: mostrar el asociado que aparece primero en una lista alfabética

SELECT MIN(nombre)

FROM asociados

- Ejemplo 20: mostrar cual es la cuota promedio que se cobra

SELECT AVG (montocuota)

FROM deportes

- Ejemplo 21: mostrar cual será la recaudación mensual de futbol.

SELECT SUM (d.montocuota)

FROM deportes d INNER JOIN practica p ON (p.iddeporte = d.iddeporte)

WHERE d.nombre = "futbol"

- Ejemplo 22: mostrar el deporte que cobra la cuota mas alta

SELECT MAX (montocuota)

FROM deportes

SELECT nombre, MAX(montocuota)

FROM deportes

SELECT nombre

FROM deportes

WHERE montocuota = (SELECT MAX(montocuota)

FROM deportes)

SQL → DML

➤ Operación de Agrupamientos (**GROUP BY**):

➤ Permite agrupar un conjunto de tuplas por algún criterio

➤ Ejemplo 23: obtener la cantidad de asociados que practica cada deporte

```
SELECT COUNT ( * )  
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio)  
                INNER JOIN deportes d ON (d.iddeporte = p.iddeporte )  
WHERE d.nombre = "futbol"
```

```
SELECT COUNT ( * )  
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio)  
                INNER JOIN deportes d ON (d.iddeporte = p.iddeporte )  
WHERE d.nombre = "voley"
```

```
SELECT COUNT ( * )  
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio)  
                INNER JOIN deportes d ON (d.iddeporte = p.iddeporte )  
WHERE d.nombre = "basquet"
```

```
SELECT COUNT ( * )  
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio)  
                INNER JOIN deportes d ON (d.iddeporte = p.iddeporte )  
GROUP BY d.nombre
```

```
SELECT d.nombre, COUNT ( * )  
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio)  
                INNER JOIN deportes d ON (d.iddeporte = p.iddeporte )  
GROUP BY d.nombre
```


SQL → DML

- Ejemplo 24: obtener la recaudación mensual de cada deporte

```
SELECT d.nombre, SUM ( d.monto cuota )
FROM deportes d INNER JOIN practica p ON ( d.iddeporte = p.iddeporte )
GROUP BY d.nombre
```

- Ejemplo 25: informar para cada asociado cuantos deportes practica

```
SELECT a.nombre, COUNT ( * )
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio )
GROUP BY a.nombre
```

```
CREATE VIEW consulta (
SELECT a.idsocio, COUNT ( * )
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio )
GROUP BY a.idsocio )
```

```
SELECT a.nombre, ?????????
FROM asociados a INNER JOIN consultas c ON ( a.idsocio = c.idsocio )
```

```
SELECT a.idsocio, COUNT ( * )
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio )
GROUP BY a.idsocio
```

```
CREATE VIEW consulta (
SELECT a.idsocio, COUNT ( * ) AS Cantidad
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio )
GROUP BY a.idsocio )
```

```
SELECT a.nombre, c.cantidad
FROM asociados a INNER JOIN consultas c ON ( a.idsocio = c.idsocio )
```


SQL → DML

- Ejemplo 26: que modificaría al problema anterior para mostrar solamente los asociados de La Plata

```
CREATE VIEW consulta (  
SELECT a.idsocio, COUNT ( * ) AS Cantidad  
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio)  
GROUP BY a.idsocio )
```

```
SELECT    a.nombre, c.cantidad  
FROM asociados a INNER JOIN consultas c ON (a.idsocio = c.idsocio )  
          INNER JOIN localidades l on (a.idocalidad = l.idlocalidad )  
WHERE l.nombre = " La Plata"
```

```
CREATE VIEW consulta (  
SELECT a.idsocio, COUNT ( * ) AS Cantidad  
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio)  
          INNER JOIN localidades l on (a.idocalidad = l.idlocalidad )  
WHERE l.nombre = " La Plata"  
GROUP BY a.idsocio )
```

```
SELECT    a.nombre, c.cantidad  
FROM asociados a INNER JOIN consultas c ON (a.idsocio = c.idsocio )
```

SQL → DML

➤ Operación de Agrupamientos (**GROUP BY**):

➤ **HAVING**: permite aplicar condiciones a los grupos

➤ Ejemplo 27: mostras todos los deportes y la cantidad de asociados que lo practican siempre y cuando para el deporte se abonen en total mas de 100000 pesos

```
SELECT d.nombre, COUNT ( * )  
FROM asociados a INNER JOIN practica p ON ( a.idsocio = p.idsocio )  
          INNER JOIN deportes d ON ( d.iddeporte = p.iddeporte )  
GROUP BY d.nombre  
HAVING SUM( d.montocuota ) > 100000
```

SQL → DML

► Valores nulos:

- Los atributos con valores nulos deben ser tratados de manera especial
- Ejemplo 28: Mostrar aquellos deportes que no tengan valor ingresado en su cuota

```
SELECT nombre  
FROM deportes  
WHERE montocuota IS NOT NULL
```

SQL → DML

➤ Subconsultas anidadas

➤ Pertenencia a conjuntos: **IN**

➤ Ejemplo 29: mostrar aquellos asociados que practiquen Basquet.

```
SELECT nombre  
FROM asociados  
WHERE idsocio IN ( SELECT idsocio  
                   FROM practica p INNER JOIN deportes d ON (d.iddeporte = p.idpractica )  
                   WHERE d.nombre = "basquet" )
```

➤ Ejemplo 30: mostrar los asociados de Gonnet que practiquen Handball.

```
SELECT nombre  
FROM asociados  
WHERE ( idsocio IN ( SELECT idsocio  
                   FROM practica p INNER JOIN deportes d ON (d.iddeporte = p.idpractica )  
                   WHERE d.nombre = "Handball" )  
and  
idlocalidad IN ( SELECT idlocalidad  
                FROM localidades  
                WHERE nombre = "gonnet" ) )
```

SQL → DML

➤ Subconsultas anidadas

➤ Comparación de Conjuntos

➤ > **some** (<, =, >=, <=, <>)

➤ Ejemplo 31: mostrar todos los deportes, menos el mas económico

```
SELECT nombre  
FROM deportes  
WHERE montocuota > SOME ( SELECT montocuota  
FROM deportes )
```

➤ > **all** (<, =, >=, <=, <>)

➤ Ejemplo 32: presentar el deporte mas oneroso

```
SELECT nombre  
FROM deportes  
WHERE montocuota >= ALL ( SELECT montocuota  
FROM deportes )
```

SQL → DML

- Cláusula **EXIST**: devuelve verdadero si la subconsulta argumento no es vacía.
 - Ejemplo 33: mostrar los asociados que practican futbol y voley (una tercera variante)

```
SELECT a.nombre
FROM asociados a
WHERE EXIST ( SELECT *
               FROM practica p INNER JOIN deportes d ON (d.iddeporte = p.iddeporte)
               WHERE d.nombre = "Futbol" and p.idsocio = a.idsocio)
AND
  EXIST ( SELECT *
          FROM practica p INNER JOIN deportes d ON (d.iddeporte = p.iddeporte)
          WHERE d.nombre = "voley" and p.idsocio = a.idsocio)
```


SQL → DML

- Cláusula **EXIST**: devuelve verdadero si la subconsulta argumento no es vacía.
 - Ejemplo 34: obtener los asociados que practiquen todos los deportes.

```
SELECT a.nombre
FROM asociados a
WHERE NOT EXIST ( SELECT *
                  FROM deportes d
                  WHERE NOT EXIST ( SELECT *
                                   FROM practica p
                                   WHERE p.idsocio = a.idsocio AND
                                         p.iddeporte = d.iddeporte) )
```

Mostrar los asociados
tal que no exista
deporte que el
asociado no
practique

SQL → DML

- Variantes del producto natural
 - Left outer Join:
 - primero se calcula el inner join (ídem anterior) y
 - luego cada tupla t perteneciente a la relación de la izquierda que no encontró un par igual en la tupla de la derecha, aparece igualmente en el resultado final con valores nulos en los atributos del correspondientes a la derecha.
 - Right outer Join: ídem anterior pero aparecen las tuplas t de la relación de la derecha
 - Full outer join: aparecen las tuplas colgadas de ambos lados.

SQL → DML

► Variantes del producto natural

► Ejemplos:

- Ejemplo 36: Presentar para cada deporte un listado de asociados que lo practican. Pueden haber deportes que aun no tengan asociados y deben aparecer igualmente en el listado

```
SELECT d.nombre, a.nombre  
FROM deportes d LEFT JOIN practica p ON ( p.iddeporte = d.iddeporte )  
LEFT JOIN asociados a ON (a.idsocio = p.idsocio )
```

- Ejemplo 37: Presentar para cada asociado los deportes que practican. Pueden que haya alguno que no practique deportes, igualmente debe aparecer.

```
SELECT a.nombre, d.nombre,  
FROM asociados a LEFT JOIN practica p ON (a.idsocio = p.idsocio)  
LEFT JOIN deportes d ON (p.iddeporte = d.iddeporte )
```

SQL → DML

➤ ABM sobre tablas:

➤ Inserción:

- Ejemplo 38: agregar un nuevo asociado a la tabla

```
INSERT INTO asociados ( "Juan Perez", "Balbin 143", "221-5133747", "masculino", "casado", 22-10-1994, 4)
```

➤ Borrado

- Ejemplo 39: quitar de la tabla el deporte bochas

```
DELETE FROM deportes
```

```
WHERE nombre = "Bochas"
```

- Ejemplo 40: como haría en el caso anterior si bochas tuviera deportistas y estuviera definida integridad referencial

```
DELETE FROM practica
```

```
WHERE iddeporte = (SELECT iddeporte  
FROM deportes
```

```
WHERE nombre = "bochas" )
```

SQL → DML

➤ ABM sobre tablas:

- Ejemplo 41: incrementar la cuota de cada deporte en un 20%

UPDATE deportes

SET monto cuota = monto cuota * 1,2

- Ejemplo 42: incrementar la cuota de hockey en un 30%

UPDATE deportes

SET montocuota = montocuota * 1,3

WHERE nombre = "hockey"