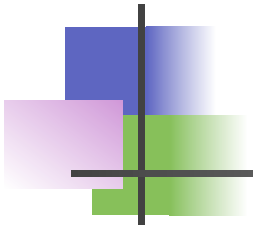
Departamento de
Ciencias de la Computación y Tecnologías de Información
Universidad del Bío-Bío
Sede Chillán

Bases de Datos

Lenguaje SQL

Mª Angélica Caro Gutiérrez
http://www.face.ubiobio.cl/~mcaro/
mcaro@ubiobio.cl



Lenguaje SQL

- Introducción
- Conceptos básicos
- Consultas básicas en SQL
- Consultas complejas en SQL
- Vistas (tablas virtuales) en SQL

Consultas Complejas en SQL

- SELECT: Consulta para recuperar datos de la BD
- La sintaxis de la orden SELECT consta básicamente de las cláusulas SELECT y FROM como obligatorias y de otras varias cláusulas opcionales:

<cláusula SELECT> <cláusula FROM>
[<cláusula WHERE>]
[<cláusula GROUP BY> [<cláusula HAVING>]]
[<cláusula ORDER BY>]

Consultas Complejas en SQL

- CONSULTAS ANIDADAS:
  - Una consulta anidada es una consulta que tiene otra consulta en su interior la cual se denomina subconsulta
  - La subconsulta puede, a su vez, contener otra subconsulta
  - Las subconsultas suelen aparecer:
    - en la cláusula WHERE,
    - en la cláusula FROM o
    - en la cláusula HAVING

Consultas Complejas en SQL

- CONSULTAS ANIDADAS:
  - Obtener el título de los libros cuya fecha de fin de préstamo es > al 31-3-2011

```
SELECT titulo FROM libro WHERE codigo
IN (SELECT codigo FROM prestamo
     WHERE fecha_fin > '31-3-2011');
```

→ **SUBCONSULTA**

Cuya salida sería:

	titulo character var
1	BASES DE DATO

Consultas Complejas en SQL

- CONSULTAS ANIDADAS:
  - Obtener el nombre de los usuarios que no han pedido libros después del 31-3-2011

```
SELECT nombre FROM usuario WHERE carnet
NOT IN (SELECT carnet FROM prestamo
        WHERE fecha_inicio > '31-3-2011');
```

Cuya salida sería:

	nombre character var
1	JAIME DURAN
2	JOSEFA PEREZ

Consultas Complejas en SQL

- CONSULTAS ANIDADAS:
  - Obtener el nombre de los usuarios que no han pedido libros de la clase 1

```
SELECT nombre FROM usuario WHERE carnet
NOT IN (SELECT carnet FROM prestamo WHERE codigo
        IN (SELECT codigo FROM libro WHERE clase = 1));
```

Cuya salida sería:

	nombre character var
1	JOSEFA PEREZ

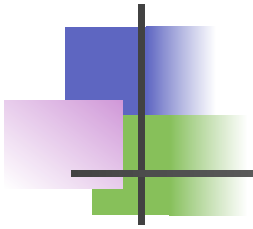
Consultas Complejas en SQL

- CONSULTAS ANIDADAS:
  - Obtener el nombre de los usuarios que han pedido libros del editor ALFAGUARA

```
SELECT nombre FROM usuario WHERE carnet
IN (SELECT carnet FROM prestamo WHERE codigo
    IN (SELECT codigo FROM libro
        WHERE editor = 'ALFAGUARA'));
```

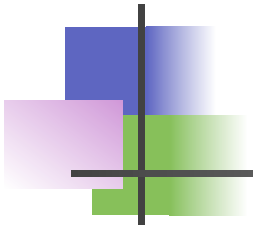
Cuya salida sería:

	nombre character var
1	JAIME DURAN
2	JOSEFA PEREZ



Consultas Complejas en SQL

- Operadores en las consultas anidadas:
  - IN Compara el valor v con un conjunto o multiconjunto de valores V y evalúa a TRUE si v es uno de los elementos de V
  - ANY⇐⇒SOME, ALL (>,>=,<,<=,=,<>)
  - =ANY o =SOME es equivalente a IN



Consultas Complejas en SQL

- Ejemplo ALL:
  - Los nombres de los empleados que cuyo salario es mayor que el de todos los empleados del departamento 5

```
SELECT apellido, nombre FROM empleado
WHERE salario > ALL (SELECT salario FROM empleado WHERE
Dno =5 );
```

Consultas Complejas en SQL

- CONSULTAS ANIDADAS CORRELACIONADAS:
  - En las consultas anidadas vistas hasta el momento la subconsulta interior ha sido completamente independiente de la consulta exterior
  - En general, la subconsulta interior puede depender de la fila (tupla) que se está examinando en cada momento en la consulta exterior
  - Ejemplo:

```
SELECT E.Nombre, E.Apellido1 FROM EMPLEADO AS E
WHERE E.Dni IN (SELECT DniEmpleado FROM Subordinado
WHERE E.Nombre = NombreSubordinado AND E.sexo = sexo);
```

Consultas Complejas en SQL

- CONSULTAS ANIDADAS CORRELACIONADAS:
  - Ambigüedades de los nombres de los atributos:

Obtener el nombre y apellido de cada empleado que tenga un familiar con el mismo nombre de pila y sexo que el empleado.

```
SELECT E.Nombre, E.Apellido1 FROM EMPLEADO AS E
WHERE E.Dni IN (SELECT DniEmpleado FROM Subordinado
WHERE E.Nombre = NombreSubordinado AND E.sexo = sexo);
```

  - La regla es: la referencia a un atributo no calificado se refiere a la relación declarada en la consulta anidada más interior

Consultas Complejas en SQL

- CONSULTAS ANIDADAS CORRELACIONADAS:
  - Siempre que una claúsula WHERE de una consulta anidada hace referencia a un atributo de una relación declarada en una consulta externa, la consulta anidada se evalúa una vez por cada tupla (o combinación de tuplas) en la consulta externa.
  - En la consulta anterior: Por cada tupla empleado, se evalua la consulta anidada.
  - En general una consulta anidada y que emplee los operadores = o IN siempre puede expresarse como una consulta de un sólo bloque Ejemplo:

```
SELECT E.Nombre, E.Apellido1 FROM Empleado AS E, Subordinado
   as D WHERE E.Dni = D.DniEmpleado AND
E.Nombre = D.NombreSubordinado AND E.sexo = D.sexo);
```

Consultas Complejas en SQL

- CONSULTAS ANIDADAS CORRELACIONADAS:
  - Claúsula EXIST (NOT EXIST), sirve para comprobar si el resultado de una consulta anidada correlacionada es o no vacío.

```
SELECT nombre FROM usuario A WHERE
EXISTS (SELECT * FROM prestamo WHERE
        A.carnet = carnet AND fecha_inicio > '31-3-2011');
```

  - La subconsulta anterior depende claramente de la fila actual A de USUARIO y se debe volver a evaluar para cada fila de USUARIO

Consultas Complejas en SQL

- CONSULTAS ANIDADAS CORRELACIONADAS:
  - Obtener el nombre de las personas que no han pedido nunca un libro

```
SELECT nombre FROM usuario A WHERE NOT EXISTS
(SELECT * FROM prestamo WHERE A.carnet = carnet);
```

  - Obtener el nombre de las personas que han pedido libros y que estos son solo de clase 1

```
SELECT nombre FROM usuario A WHERE EXISTS
(SELECT * FROM prestamo B
   WHERE A.carnet = B.carnet AND
   NOT EXISTS (SELECT * FROM libro C
   WHERE B.codigo = C.codigo AND clase <> 1));
```

Consultas Complejas en SQL

- CONJUNTOS EXPLICITOS:
  - Obtener el nombre de los libros que pertenecen a las clases 1, 2 y 3

```
SELECT titulo FROM libro
 WHERE clase IN (1,2,3);
```

  - Resultado:

	titulo character varying(30)
1	BASES DE DATOS
2	HIJO DE LADRON
3	VASO DE LECHE

Consultas Complejas en SQL

- Valores null (IS NULL y IS NOT NULL):
  - Obtener el titulo de los libros sin editor

```
SELECT titulo FROM libro
 WHERE editor IS NULL;
```

  - Obtener el nombre y dirección de todos los usuarios que posean dirección:

```
SELECT nombre, direccion FROM usuario
 WHERE direccion IS NOT NULL;
```

Consultas Complejas en SQL

- Tablas combinadas (JOIN)
  - Recuperar los nombres y dirección de los empleados que trabajan en el departamento de Investigación

```
SELECT NOMBRE, APELLIDO1, DIRECCION FROM
  (EMPLEADO JOIN DEPARTAMENTO ON DNO =
  NUMERODPTO) WHERE NOMBREDPTO=
  'Investigacion';
```

Consultas Complejas en SQL

- Tablas combinadas (NATURAL JOIN)
  - Recuperar los nombres y dirección de los empleados que trabajan en el departamento de Investigación

```
SELECT NOMBRE, APELLIDO1, DIRECCION FROM
  (EMPLEADO NATURAL JOIN DEPARTAMENTO AS
  DEPTO(NOMBRED, DNO, DNIS, FECHAING)) WHERE
  NOMBRED = 'Investigacion';
```

Consultas Complejas en SQL

- Variantes de la Cláusula JOIN
  - SQL soporta algunas variedades interesantes de la operación join que aprovechan los valores nulos, las que se denominan Outer Joins
  - Considere la sgte operación: Navegantes $\lhd\rhd_{idn=idn}$ Reservas

Navegantes

idn	nombre	edad	categoria
22	*Pedro*	45	4
23	*Andres*	35	6
33	*Loreto*	31	6
29	*Natalia*	40	7
30	*Esteban*	50	8

Reservas

idn	idb	fecha
23	102	10.11.00
22	102	10.11.00
33	101	05.01.02

  - El resultado es:

idn	nombre	edad	categoria	idn	idb	fecha
22	*Pedro*	45	4	22	102	10.11.00
23	*andres*	35	6	23	102	10.11.00
33	*loreto*	31	6	33	101	05.01.02

Consultas Complejas en SQL

- Cláusula JOIN
  - Sin embargo, podría ser interesante mantener las tuplas de Navegantes que no tienen reservas en el resultado. Para esto usamos el Outer Join
  - Con Outer Join, las tuplas que no tienen reservas aparecen en el resultado del join y los atributos correspondientes a reservas toman valores nulos
  - Existen tres variantes de Outer Join:
    - Left outer join
    - Right outer join
    - Full outer join

Consultas Complejas en SQL

- Cláusula LEFT OUTER JOIN
  - Consideremos la consulta:

```
SELECT *
FROM Navegantes NATURAL LEFT OUTER JOIN Reservas
```

  - El resultado es:

idn	nombre	edad	categoria	idn	idb	fecha
22	*Pedro*	45	4	22	102	10.11.00
23	*andres*	35	6	23	102	10.11.00
33	*loreto*	31	6	33	101	05.01.02
29	*natalia*	40	7	NULL	NULL	NULL
30	*esteban*	50	8	NULL	NULL	NULL

Consultas Complejas en SQL

- Cláusula RIGHT OUTER JOIN
  - Consideremos las siguientes relaciones:

Reservas		
idn	idb	fecha
23	102	10.11.00
22	102	10.11.00
33	101	05.01.02

Botes		
idb	nombreb	color
101	*marino*	azul
102	*inter-lagos*	rojo
103	*clipper*	verde
104	*inter-lagos*	rojo

  - SELECT * FROM Reservas NATURAL RIGHT OUTER JOIN Botes

idn	idb	fecha	idb	nombreb	color
23	102	10.11.00	102	*inter-lagos*	rojo
22	102	10.11.00	102	*inter-lagos*	rojo
33	101	05.01.02	101	*marino*	azul
NULL	NULL	NULL	103	*clipper*	verde
NULL	NULL	NULL	104	*inter-lagos*	rojo

Consultas Complejas en SQL

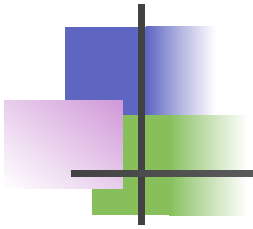
- Cláusula FULL OUTER JOIN
  - Consideremos las siguientes relaciones:

Reservas		
idn	idb	fecha
23	102	10.11.00
22	102	10.11.00
33	101	05.01.02
33	106	06.01.02

Botes		
idb	nombreb	color
101	*marino*	azul
102	*inter-lagos*	rojo
103	*clipper*	verde
104	*inter-lagos*	rojo

  - SELECT * FROM Reservas NATURAL FULL OUTER JOIN Botes

idn	idb	fecha	idb	nombreb	color
23	102	10.11.00	102	*inter-lagos*	rojo
22	102	10.11.00	102	*inter-lagos*	rojo
33	101	05.01.02	101	*marino*	azul
NULL	NULL	NULL	103	*clipper*	verde
NULL	NULL	NULL	104	*inter-lagos*	rojo
33	106	06.01.02	NULL	NULL	NULL



Consultas Complejas en SQL

- Operador UNION y UNION ALL
  - Al hacer la unión de dos sentencias select, se devolverán los resultados de cada una de las dos sentencias select, eliminando los duplicados.
  - Para poder hacer la unión es necesario que ambas sentencias select se hagan sobre el mismo número de columnas y con el mismo tipo. El nombre de las columnas no tienen porque ser iguales.
  - Existe una variante, el operador UNION ALL que no elimina duplicados (en este caso no se puede usar la restricción DISTINCT).

Consultas Complejas en SQL

- Operador UNION y UNION ALL
  - Trabajando con las tablas del ejemplo:

    SELECT autor FROM libro UNION

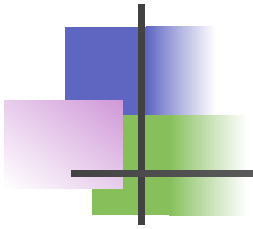
    SELECT nombre FROM usuario;
  - Cuya salida sería:

UNION

	autor character varying(30)
1	ELMASRI, RAMES
2	JAIME DURAN
3	JOSEFA PEREZ
4	JUAN CARLOS IBARRA
5	ROJAS, MANUEL

UNION ALL

	autor character varying(30)
1	ELMASRI, RAMES
2	ROJAS, MANUEL
3	ROJAS, MANUEL
4	JUAN CARLOS IBARRA
5	JAIME DURAN
6	JOSEFA PEREZ



Consultas Complejas en SQL

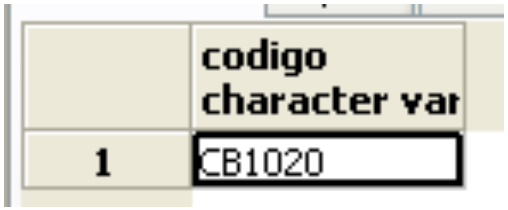
- Operador INTERSECT
  - Al hacer la intersección de dos sentencias select, se devolverán los resultados comunes a las dos sentencias select.
  - Para poder hacer la intersección es necesario que ambas sentencias select se hagan sobre el mismo número de columnas y con el mismo tipo. El nombre de las columnas no tienen porque ser iguales.

Consultas Complejas en SQL

- Operador INTERSECT
  - Trabajando con las tablas del ejemplo:

```
SELECT codigo FROM libro WHERE clase = 1 INTERSECT
SELECT codigo FROM prestamo;
```

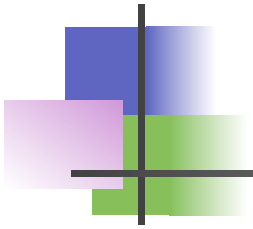
Cuya salida sería:



	codigo character var
1	CB1020

  - Equivalente a:

```
SELECT DISTINCT A.codigo FROM libro A, prestamo B WHERE
  A.codigo = B.codigo and clase = 1;
```



Consultas Complejas en SQL

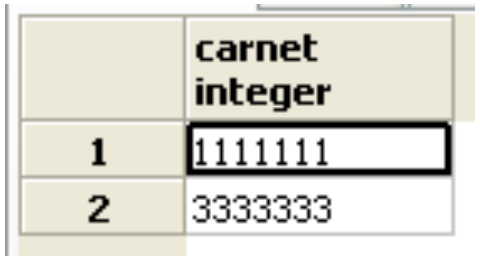
- Operador EXCEPT
  - Este operador permite la exclusión de tuplas resultantes de una sentencias select respecto de las tuplas resultantes de otra sentencia select.
  - Para poder usar este operador es necesario que ambas sentencias select se hagan sobre el mismo número de columnas y con el mismo tipo. El nombre de las columnas no tienen porque ser iguales.

Consultas Complejas en SQL

- Operador EXCEPT
  - Trabajando con las tablas del ejemplo:

```
SELECT carnet FROM prestamo EXCEPT
SELECT carnet FROM usuario WHERE nombre = 'JAIME DURAN';
```

Cuya salida sería:



	carnet integer
1	1111111
2	3333333

Consultas Complejas en SQL

- OPERADORES DE AGREGACION:
  - SQL soporta cinco operaciones de agregación:
    - COUNT ([DISTINCT] A): calcula el número de valores (únicos) de la columna A. (Número de tuplas)
    - SUM ([DISTINCT] A): calcula la suma de todos los valores (únicos) de la columna A (Número de tuplas)
    - AVG ([DISTINCT] A): calcula el promedio de todos los valores (únicos) de la columna A
    - MAX (A): calcula el valor máximo de la columna A
    - MIN (A): calcula el valor mínimo de la columna A

Consultas Complejas en SQL

- EJEMPLO DE CONSULTAS DE AGREGACION:
  - Encontrar el promedio de días de préstamo de todas las clases de libros:

```
SELECT AVG (tiempo_prestamo)
FROM clase;
```

  - Dado el esquema: ALUMNO(ID,NOMBRE,EDAD, CIUDAD), encontrar el promedio de edad de los alumnos de la ciudad de Chillán:

```
SELECT AVG (EDAD)
FROM ALUMNO
WHERE CIUDAD = 'Chillan'
```

Consultas Complejas en SQL

- EJEMPLO DE CONSULTAS DE AGREGACION:
  - Obtener el tiempo máximo de prestamo:

    SELECT MAX (tiempo_prestamo) from clase;

  - Obtener el tiempo mínimo de prestamo:

    SELECT MIN (tiempo_prestamo) from clase;

  - Obtener el titulo, autor y tiempo de préstamo de los libros con más tiempo de préstamo:

    SELECT titulo, autor, tiempo_prestamo
    FROM libro, clase WHERE tiempo_prestamo = (SELECT MAX(tiempo_prestamo) from clase)
    AND clase = clave;

Consultas Complejas en SQL

- EJEMPLO DE CONSULTAS DE AGREGACION:
  - Contar el número total de libros de que se dispone:

```
SELECT COUNT(*)
FROM libro;
```

  - Contar el número de usuarios (distintos) que han pedido libros:

```
SELECT COUNT(DISTINCT carnet)
FROM prestamo;
```

  - Contar el número de usuarios que han pedido 2 o más libros:

```
SELECT COUNT(*) FROM usuario A
WHERE (SELECT COUNT(*)
FROM prestamo B
WHERE A.carnet = B.carnet) >=2;
```

Consultas Complejas en SQL

- EJEMPLO DE CONSULTAS DE AGREGACION:

  - Recuperar el total de empleados del departamento de investigación

```
SELECT COUNT(*)
FROM (EMPLEADO JOIN DEPARTAMENTO ON DNO = NUMERODPTO)
WHERE NOMBREDPTO= 'Investigacion';
```

  - Recupere el nombre de los empleados que tienen más de dos cargas familiares (dependientes)

```
SELECT APELLIDO1, NOMBRE
FROM EMPLEADO
WHERE (SELECT COUNT(*)
FROM SUBORDINADO
   WHERE DNI=DNIEMPLEADO) > 2;
```

Consultas Complejas en SQL

- EJEMPLO DE CONSULTAS DE AGREGACION:
  - Encontrar la suma de todos los salarios de los empleados del departamento de investigación, así como el sueldo máximo, el salario mínimo y el salario promedio de dicho departamento.

```
SELECT SUM(SUELDO),
          MAX(SUELDO),
           MIN(SUELDO),
           AVG(SUELDO)
FROM (EMPLEADO JOIN DEPARTAMENTO ON DNO = NUMERODPTO)
WHERE NOMBREDPTO = 'Investigacion';
```