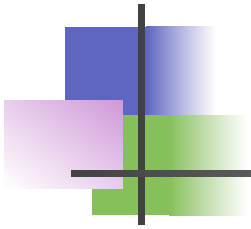
Departamento de
Ciencias de la Computación y Tecnologías de Información
Universidad del Bío-Bío
Sede Chillán

Bases de Datos

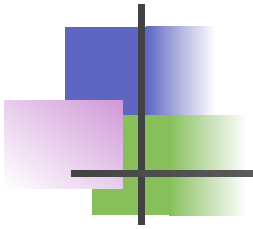
Lenguaje SQL

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



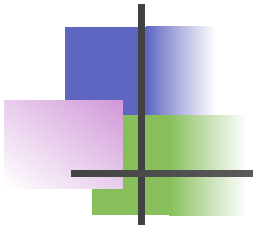
Lenguaje SQL

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



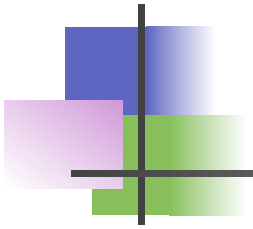
Introducción

- SQL: Structured Query Language, diseñado por la IBM
- Estándar SQL ANSI/ISO, con variaciones entre los SQL comerciales.
- SQL es el lenguaje más usado en los SGBD relacionales.
- Funcionalidades:
  - **Lenguaje de manipulación de datos(LMD)**. Este subconjunto del SQL sirve para realizar consultas, insertar, eliminar y modificar tuplas
  - **Lenguaje de definición de datos (LDD)**. Este subconjunto del SQL sirve para crear, eliminar y modificar las definiciones de tablas y vistas
  - **Triggers y restricciones de integridad**. Los triggers son acciones ejecutadas por el SGBD cuando cambios en la BD cumplen ciertas condiciones
  - **SQL embebido y SQL dinámico**.
    - El SQL embebido permite insertar código SQL en un programa escrito en otro lenguaje e.g., C, JAVA, etc.
    - El SQL dinámico permite que una consulta sea construida y ejecutada en tiempo de ejecución



Introducción

- Funcionalidades:
  - **Ejecución cliente-servidor y acceso remoto a BDs**. SQL provee comandos para que las aplicaciones clientes se conecten a un servidor SQL remoto, o accedan datos a través de una red
  - **Manejo de transacciones**. Comandos de SQL permiten al usuario controlar la ejecución de una transacción.
  - **Seguridad**. SQL provee mecanismos para controlar el acceso de usuarios a las tablas y vistas
  - **Características avanzadas**. El SQL:1999 incluye aspectos de OO (orientación a objetos), consultas recursivas (con restricciones), apoyo a las consultas de toma de decisiones, data mining, datos espaciales, XML, etc.



Lenguaje SQL

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

Conceptos Básicos

- Algunas Sentencias:

Manipulación de Datos	SELECT	Recupera datos de la BD
	INSERT	Añade nuevas tuplas a la BD
	DELETE	Elimina tuplas de la BD
	UPDATE	Modifica datos existentes en la BD
Definición de Datos	CREATE TABLE	Añade una nueva tabla a la BD
	DROP TABLE	Elimina una tabla de la BD
	ALTER TABLE	Modifica la estructura de una tabla existente en la BD
Control de acceso	GRANT	Concede privilegios de acceso a los usuarios
	REVOKE	Suprime privilegios de acceso a los usuarios
Control de transacciones	COMMIT	Finaliza la transacción actual
	ROLLBACK	Aborta la transacción actual
SQL Programático	DECLARE	Define un cursor para una consulta
	OPEN	Abre un cursor para recuperar resultados de una consulta
	FETCH	Recupera una tupla de resultados de una consulta

Conceptos Básicos

- Algunos tipos de datos:

char(n)	Cadena de caracteres de longitud fija n
varchar(n)	Cadena de caracteres de longitud variable n
date	Fecha
int	Entero (interger)
smallint	Entero pequeño
time	Hora del día, en horas, minutos y segundos
timestamp	Combinación de date y time
boolean	Booleano lógico(verdadero/falso)
Float(n)	Números en coma flotante, precisión n dígitos

Conceptos Básicos

- Constantes:
  - SQL permite el uso de constantes numéricas, de cadena, fecha y hora y simbólicas (current _date).
- Funciones SQL:
  - Char_Length(cadena)
  - Lower(cadena)
  - Upper(cadena)
  - Float(int)
  - …
- Falta de valores = Null
- 👁: Siempre es conveniente revisar las particularidades de cada implementacion de SQL

Conceptos Básicos

- Documentación :
  - https://www.postgresql.org/

Conceptos Básicos

- Creación de una la tabla: CREATE TABLE

- Sintaxis general de la sentencia:

```
CREATE TABLE <nombre de tabla>
(nombre_columna1 tipo [restricción de columna],
.....
nombre_columnaN tipo [restricción de columna],
[restricción_de_tabla]);
```

- La sentencia **CREATE TABLE** se utiliza para crear una tabla dentro de la cual habrá columnas que contienen datos y restricciones

Conceptos Básicos

- CREATE TABLE, más en detalles:

CREATE TABLE <nombre tabla>
(<nombre columna> <tipo de dato>
[NOT NULL] [UNIQUE]
[CONSTRAINT <nombre restricción>]
[PRIMARY KEY] [REFERENCES] [DEFAULT] [CHECK]
| [PRIMARY KEY (<lista columnas>)]
| [FOREIGN KEY (<lista columnas>) REFERENCES (nombretabla)]
| [UNIQUE (<lista columnas>)] [CONSTRAINT <nombre restricción>],[,...])
| [CHECK (condición de búsqueda)]

Conceptos Básicos

- CREATE TABLE, restricciones de columnas:

NOT NULL. La columna no permitirá valores nulos.

CONSTRAINT. Permite asociar un nombre a una restricción.

DEFAULT valor. La columna tendrá un valor por defecto. El SGBD utiliza este valor cuando no se especifica un valor para dicha columna.

PRIMARY KEY. Permite indicar que esta columna es la clave primaria.

REFERENCES. Es la manera de indicar que este campo, es clave ajena y hace referencia a una clave candidata de otra tabla. Esta **foreign key** es sólo de una columna.

UNIQUE. Obliga a que los valores de una columna tomen valores únicos (no puede haber dos filas con igual valor). Se implementa creando un índice para dicha(s) columna(s).

CHECK (condición). Permite indicar una condición que debe de cumplir esa columna.

Conceptos Básicos

- CREATE TABLE, restricciones de tablas:

PRIMARY KEY (columna1, columna2…). Permite indicar las columnas que forman la clave primaria.

FOREIGN KEY (columna1, columna2….) REFERENCES NombreTabla. Indica las columnas que son clave ajena referenciando a una clave candidata de otra tabla.

UNIQUE (columna1, columna2…). El valor combinado de una o varias columnas es único.

CHECK (condición). Permite indicar una condición que deben cumplir las filas de la tabla. Puede afectar a varias columnas.

Conceptos Básicos

- Creación de una la tabla:

```
CREATE TABLE EMPLEADO (
        NOMBRE VARCHAR(15) NOT NULL,
        APELLIDO1 VARCHAR(15) NOT NULL,
        APELLIDO2 VARCHAR(15) NOT NULL,
        DNI CHAR(9) NOT NULL,
        FECHANAC DATE,
        DIRECCION VARCHAR(15),
        SEXO CHAR,
        SUELDO DECIMAL(10,2) CHECK (SUELDO > 0) ,
        SUPERDNI CHAR(9),
        DNO INT NOT NULL,
    PRIMARY KEY (DNI),
    FOREIGN KEY (DNO) REFERENCES DEPARTAMENTO(NUMERODPTO));
```

Conceptos Básicos

- Creación de tablas, ejercicio clase: Departamento y Localizaciones_dpto

  - **CREATE TABLE** DEPARTAMENTO
    (NOMBREDPTO VARCHAR(29),
    NUMERODPTO INT **NOT NULL PRIMARY KEY**,
    DNIDIRECTOR CHAR(9) **NOT NULL**,
    FECHAINGRESODIRECTOR DATE,
    FOREIGN KEY (DNIDIRECTOR) **REFERENCES** EMPLEADO(DNI));

  - **CREATE TABLE** LOCALIZACIONES_DPTO
    (NUMERODPTO INT **NOT NULL**,
    UBICACIONDPTO VARCHAR(20) **NOT NULL**,
    PRIMARY KEY (NUMERODPTO, UBICACIONDPTO),
    FOREIGN KEY (NUMERODPTO) **REFERENCES** DEPARTAMENTO);

Conceptos Básicos

- La cláusula **Foreign Key** tiene unas opciones optativas que se explican a continuación:
  - Tratamiento de nulos: ¿Cómo debe tratar el SGBD un valor NULL en una o más columnas de la clave ajena, cuando lo compare con las filas de la tabla padre?
    - Modo de borrado: Para determinar la acción que se debe realizar cuando se **elimina** una fila referenciada, se debe utilizar una **regla de supresión opcional** para la relación (CASCADE, SET NULL, SET DEFAULT, NO ACTION).
    - Modo de modificación: Una regla de actualización para la relación, que determina la acción que se debe realizar cuando se **modifica** la clave candidata de la fila referenciada (CASCADE, SET NULL, SET DEFAULT, NO ACTION).

Conceptos Básicos

- Ejemplo:

```
CREATE TABLE DEPARTAMENTO
   (NOMBREDPTO VARCHAR(29),
    NUMERODPTO INT NOT NULL PRIMARY KEY,
    DNIDIRECTOR CHAR(9) NOT NULL,
    FECHAINGRESODIRECTOR DATE,
    FOREIGN KEY (DNIDIRECTOR) REFERENCES EMPLEADO(DNI)
    ON DELETE SET NULL  ON UPDATE CASCADE);
```

- **ON DELETE Set Null** Significa que si se borra algún departamento de la tabla DEPARTAMENTO, por ejemplo el campo DNO de las filas de la tabla EMPLEADO que le referenciaban se pone como Null.

- **ON UPDATE CASCADE** Significa que si se modifica la clave de una fila de la tabla DEPARTAMENTO, también se modificará en las filas de la tabla EMPLEADO que le referencian.

Conceptos Básicos

- Renombrar una tabla

  RENAME TABLE <nombre tabla existente> **TO** <nuevo nombre tabla>

- Eliminación de tablas:

  DROP TABLE <nombre tabla**> [CASCADE, RESTRICT]**

  Ejemplos:

  DROP TABLE SUBORDINADO CASCADE
  (La tabla se borra, así como las posibles restricciones relativas a esta tabla)

  DROP TABLE EMPLEADO RESTRICT
  (La tabla se borra sólo si no se hace referencia a ella en ninguna restricción, p.e. en la definición de claves ajenas)

Conceptos Básicos

- Modificación del esquema de la tabla:

ALTER TABLE <nombre tabla>
{ **ADD** <nombre columna nueva> <tipo de dato> [NOT NULL]
MODIFY <nombre columna> [DEFAULT | DROP DEFAULT] valor
DROP <nombre columna> [CASCADE | RESTRICT]
ADD [PRIMARY KEY (nombre columna) |
FOREIGN KEY (nombre columna) REFERENCES nombre_tabla | UNIQUE (nombre columna) | CHECK (condición)
DROP CONSTRAINT nombre-restricción [CASCADE| RESTRICT]

Dentro de la tabla podemos **Add** (añadir) o **Drop** (borrar) columnas y restricciones (PRIMARY KEY, FOREING KEY, UNIQUE, CHECK, CONSTRAINT).

ALTER TABLE EMPLEADO **ADD** CARGO VARCHAR(10);

Conceptos Básicos

- Inserción de filas en una tabla

  INSERT INTO <nombre tabla> [(<lista-de columnas>)]
  {VALUES (<lista-de valores>) |<sentencia SELECT>}

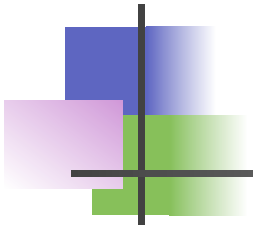
  Se añaden una o varias filas a una tabla y se puede emplear para dos objetivos distintos:

  - **a) Insertar una fila en una tabla**

    INSERT INTO <nombre tabla> [(<lista-de columnas>)] VALUES (<lista-de valores>)

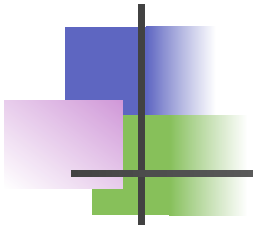
  - **b) Insertar en una tabla el resultado de una consulta (inserción multifila)**

    INSERT INTO <nombre tabla> [(<lista-de columnas>)] <sentencia SELECT>



Conceptos Básicos

- Inserción de filas en una tabla
  - Los valores de tipo DATE o CHAR se deben encerrar entre comillas simples.
  - Si no se indica la lista de columnas, se consideran todas las de la tabla.
  - En la opción b) la tabla donde se inserta se puede utilizar en la consulta SELECT.
  - En la opción b) la consulta no puede incluir una cláusula ORDER BY.
  - Los valores insertados deben ser de un tipo compatible con el de las columnas de la tabla.



Conceptos Básicos

- Eliminación de filas en una tabla

DELETE FROM <nombre tabla> [WHERE <condición>]

Permite eliminar filas de una tabla. Si no se pone condición, se borran todas las filas de la tabla.

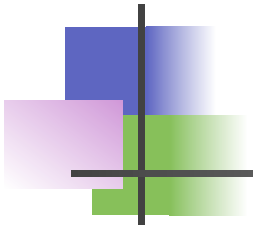
Conceptos Básicos

- Modificación de los datos de las filas existentes en una tabla:

```
UPDATE <nombre tabla> SET
{  <columna> = <expresión> [, ...]
{(<lista-de columnas>) | *} = (<lista-de expresiones>)}
[WHERE <condición>]
```

- Una expresión puede estar formada por una subconsulta SELECT entre paréntesis cuyo resultado es una única fila de una sola columna (un único valor simple).

- La lista de columnas está formada por aquellas columnas a las cuales se les modificará su valor. Si se omite la cláusula WHERE entonces se actualizan todas las filas de la tabla.



Lenguaje SQL

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

Consultas Básicas 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>]

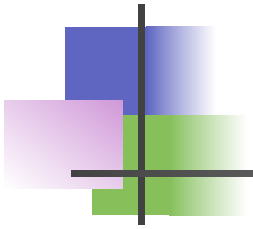
Consultas Básicas en SQL

- <Cláusula SELECT>

  - Permite indicar los datos que queremos seleccionar

    SELECT [ALL | DISTINCT | UNIQUE]
    <lista-de selecciones>

  - ***DISTINCT***: Elimina las filas duplicadas en el resultado de la consulta.
  - ***UNIQUE*** es igual que DISTINCT.
  - ***<lista-de selecciones>:*** Lista de nombre de columnas o expresiones separadas por comas.



Consultas Básicas en SQL

- <Cláusula FROM>
  - Permite indicar las tablas que contienen los datos.

    FROM {<nombre de tabla> [alias de tabla] }

Consultas Básicas en SQL

- <Cláusula WHERE>
  - Sirve para indicar la condición que deben cumplir las filas resultantes.

WHERE <condición>

  - Una condición está formada por una o varias expresiones condicionales conectadas por los operadores lógicos AND, OR y NOT.

Consultas Básicas en SQL

- La expresión condicional de la cláusula WHERE tiene una de las formas siguientes (1/2):

<expresión1> <operador relacional> <expresión2>

Verifica si las dos expresiones satisfacen la comparación.

<expresión1> [NOT] BETWEEN <expresión2> AND <expresión3>

Verifica si la expresión1 tiene un valor comprendido entre los valores de la expresión2 y la expresión3.

<expresión> [NOT] IN (<lista-de valores>)

Verifica si la expresión tiene un valor de los indicados en la lista de valores.

Consultas Básicas en SQL

- La expresión condicional de la cláusula WHERE tiene una de las formas siguientes (2/2):

<nombre de columna> [NOT] LIKE "<string>" [ESCAPE "<carácter de escape>"]

Verifica si el valor de la columna se adapta al patrón de búsqueda (string). Se admiten caracteres comodín ("%" representa cero o más caracteres, "_" representa un único carácter). El carácter de escape '/' permite referirse a los caracteres comodín como caracteres y no como comodines.

<nombre de columna> IS [NOT] NULL

Verifica si el valor de la columna es nulo.

Consultas Básicas en SQL

- Ejemplos:
  - Libro (codigo, autor, titulo, editor, clase),
  - Recuperar el autor y titulo del libro cuyo codigo es 'CB1020'.

```
SELECT AUTOR, TITULO
FROM LIBRO
WHERE CODIGO='CB1020';
```

Consultas Básicas en SQL

- Ejemplos:
  - Libro (codigo, autor, titulo, editor, clase),
  - Recuperar el autor y titulo del libro cuyo editor es 'Alfaguara' y de clase 3.

```
SELECT AUTOR, TITULO
FROM LIBRO
 WHERE EDITOR='ALFAGUARA' AND CLASE=3;
```

Consultas Básicas en SQL

- Ejemplos:
  - Libro (codigo, autor, titulo, editor, clase),
  - Recuperar el editor de todos los libros de clase 3.

```
SELECT EDITOR
FROM LIBRO
WHERE CLASE=3;
```

¿=?

```
SELECT DISTINC EDITOR
FROM LIBRO
WHERE CLASE=3;
```

Consultas Básicas en SQL

- Ejemplos:
  - Libro (codigo, autor, titulo, editor, clase),
  - Recuperar todos los datos de los libros de clase 3.

```
SELECT CODIGO, AUTOR, TITULO, EDITOR, CLASE
FROM LIBRO
 WHERE CLASE=3;
```

ó

```
SELECT *
FROM LIBRO
 WHERE CLASE=3;
```