**MYSQL**

**ÍNDICE**

[1 - Conectar/desconectar del servidor](#_n8f90rqaa90o)

[2 - Podemos ejecutar comandos del sistema operativo del anfitrión con:](#_enh01uyxgjwy)

[3 - El diccionario de datos](#_ouaimodfexua)

[4 - Cada usuario tendrá una visión del SGBD en función de los permisos que tenga](#_dhzaz3w53try)

[5 – Comando para ver las BBDD a las que tiene acceso el usuarios](#_2zymaq1hxkql)

[6 – Abrir una base de datos](#_hcwyvnr8iydv)

[7 – Ver todas las tablas o relaciones de una BD](#_u2njf1tn5h4z)

[8 – Ver la descripción y algunas propiedades de una tabla o relación](#_k7xmjxak60fd)

[9 – Podemos obtener info genérica como:](#_mu6avcmzwrk1)

[10 – Lenguaje SQL contiene los 3 lenguajes de SGBD](#_qmsdvzpiexlt)

[-DDL (Lenguaje de Definición de Datos)](#_z0v6nlvb31nj)

[-DCL (Lenguaje de Control de Datos)](#_vcpi65b51fr8)

[-DML (Lenguaje de Manipulación de Datos)](#_mhmd7px6jejm)

[11 – Crear y borrar una BD](#_5egjt4sm3mjj)

[12 – Crear un usuario](#_hcup7oya89sk)

[13 – Borrar usuario](#_hfeyvac9v795)

[14 – Creación de tablas o relaciones](#_tu0uf755yipq)

[a) Tipos de datos caracteres](#_ri0adshlca2r)

[b) Tipos de datos numéricos](#_x3ehevsyq826)

[c) Tipos de datos tiempo](#_4edklwq3znq6)

[d) Datos de tipo lógico o booleano](#_qeizn0547hb5)

[Crear una nueva relación o tabla](#_y4hynojymee7)

[RESTRICCIONES SEMÁNTICAS](#_sbxstj9hsd4f)

[Borrar una relación o tabla](#_eyrchgd8fksj)

[Modificar una relación o tabla existente](#_z3owu0qrvtpx)

[a) Añadir un nuevo atributo a una tabla existente](#_sj7n5tb7rt6p)

[b) Borrar un atributo existente](#_z3dwuouru0al)

[c) Modificar un atributo existente](#_7lf9g3lwbm4n)

[CONSTRAINTS](#_554bousdvraa)

[Situaciones en las que debemos usar un CONSTRAINTS](#_oao0gkrcdizj)

[a) Cuando la PK (Primary Key) de una relación está formada por más de un atributo](#_494k4pw4rwzm)

[b) Para la realización de chequeos o validaciones](#_k5xwuxrlrgt1)

[c) Declaración de FK (Foreign Key)](#_uz9qiy5ruqqp)

[LOS SCRIPT SQL](#_omnhcagmiu0w)

[Cláusula IF NOT EXISTS](#_k55z1yn118p5)

[Usuarios de MYSQL](#_praqrm9pkz0g)

[LENGUAJE DE CONTROL DE DATOS (DCL)](#_915q4k7r33a2)

[EL VERBO GRANT](#_5ih1xxdfm3zz)

[EL VERBO REVOKE](#_34kkoqp7te5)

[PROCESO PARA LA CREACIÓN DE BASES DE DATOS](#_p2ww9vlxabxi)

[LENGUAJE DE MANIPULACIÓN DE DATOS (DML)](#_z5rs2rcqttvg)

[AÑADIR DATOS – VERBO INSERT](#_x2uucu6hv7f7)

[BORRAR DATOS – VERBO DELETE](#_3hym37uijcus)

[MODIFICAR DATOS – VERBO UPDATE](#_2rpau8ufxgfr)

[CONSULTA DE DATOS – VERBO SELECT](#_a5jf6sijd0cl)

[Consultas básicas sobre una única tabla](#_kg0xj6twncle)

[Operador proyección](#_gthur8jdj0il)

[Operador selección](#_2tzml8urlkp1)

[ORDER BY](#_ghjoidj34tur)

[Cláusula DISTINCT](#_82uvyve9ymkf)

[Expresiones](#_ogc6i081spjk)

[Predicado NULL](#_uvp60l2yvs9z)

[Predicado BETWEEN](#_q9x89z7drbn6)

[Predicado IN](#_z23v9yu6e7e5)

[Predicado LIKE](#_vyolutb5767)

[TRATAMIENTO DE VALORES NULOS](#_lmnpczm84r6t)

[ANEXO: Funciones de cadena de mysql (no están en el estándar)](#_g09lphdhl6o3)

[Predicado BETWEEN](#_jy0du9hrysea)

[PREDICADOS Y SUBCONSULTAS](#_129fzvoc2yvx)

[Subconsultas que devuelven un único valor](#_rzhn7j68o6dw)

[Subconsultas que devuelven más de un valor](#_ncibxwl0b0x7)

[Predicados cuantificados](#_2ie74xj3dcbr)

[CASOS ESPECIALES](#_rvl72i6f95z0)

[Predicado IN](#_z2z2e1h87h63)

[Predicado EXISTS](#_gy9aofk2zlfe)

[DATOS DE TIPO TIEMPO](#_87lxbubsvjcn)

[REGISTROS ESPECIALES](#_v208n94owfua)

[Funciones que operan con datos de tipo tiempo](#_ibju8o203wwx)

[1.- Funciones que permiten descomponer un dato tiempo](#_fxotk8puorwj)

[2.- Funciones para formatear datos tiempo en mysql](#_5ea1zf3zxvnd)

[3.- Aritmética de fechas](#_tqdztwonxx2d)

[3.1.- Incrementar o decrementar un dato de tipo tiempo](#_8gksg61ea9sa)

[3.2.- Diferencia entre datos de tipo tiempo](#_5p2gq9z7n95s)

[CONSULTAS CON AGRUPAMIENTO DE FILAS](#_7auwxg1ury2x)

[1 - Agrupamiento implícito. Funciones colectivas o de columna](#_fo4dr9jxlobl)

[Reglas de uso](#_4nu2ki2a3m9u)

[Formatos de uso](#_chytx09r02d5)

[2 - Agrupamiento explícito](#_8uffsy3esr43)

[CONSULTAS SOBRE VARIAS TABLAS](#_7uy0f0d1t1hh)

[1 - Álgebra relacional](#_tm7mfe417ehn)

[1.1 - Atendiendo a su naturaleza](#_xk5q1g7ghwr4)

[1.2 - Atendiendo el número de operandos](#_pwfyzayvh9j6)

[Operadores Algebráicos](#_6rl22jnq0d)

[1 - UNIÓN](#_w3va1q3xe883)

[2 - DIFERENCIA](#_3sq6jwwzftd8)

[3 -INTERSECCIÓN](#_gfmhbmgzvoif)

[4 - PROYECCIÓN](#_decb8izavmp2)

[5- SELECCIÓN](#_a11bhjyahs8t)

[6 - PRODUCTO CARTESIANO](#_8yyfs6t0mu8r)

[Cualificación de nombres](#_qs26ccux230a)

[7 - REUNIÓN](#_21wsgvj8jkn4)

[MÉTODO PARA LA REALIZACIÓN DE CONSULTAS DE TODO TIPO](#_kaambj4v0mw)

# 1 - Conectar/desconectar del servidor

mysql [-h nombre\_host] -u nombre\_usuario -p

mysql -u root -p

contraseña: root

exit o quit (Para salir)

dentro del servidor todas las órdenes terminan en ;

# 2 - Podemos ejecutar comandos del sistema operativo del anfitrión con:

system comando\_SO

# 3 - El diccionario de datos

Son los metadatos del sistema gestor de la base de datos. En mysql está compuesto de 4 Bases de datos, que son:

-information\_schema: Contiene información sobre las bases de datos, las tablas, las vistas, etc.

-performance\_schema: Contiene info sobre el rendimiento del servidor

-mysql: Contiene información sobre los parámetros del SGBD, usuarios y privilegios.

-sys: Contiene información sobre los procedimientos y funciones almacenadas.

# **4 - Cada usuario tendrá una visión del SGBD en función de los permisos que tenga**

El usuario root tiene todos los permisos y, por lo tanto , acceso a toda la información.

# 5 – Comando para ver las BBDD a las que tiene acceso el usuarios

show databases;

# 6 – Abrir una base de datos

use nombre\_bd;

No existe un comando para cerrar una BD, simplemente al abrir otra, cambia.

# 7 – Ver todas las tablas o relaciones de una BD

show tables;

# 8 – Ver la descripción y algunas propiedades de una tabla o relación

describe nombre\_tabla;

# 9 – Podemos obtener info genérica como:

usuario conectado, versión, fecha y hora actuales, etc.

select user(), select version(), select current\_date, select current\_time

# 10 – Lenguaje SQL contiene los 3 lenguajes de SGBD

## -DDL (Lenguaje de Definición de Datos)

Es el lenguaje que se utiliza para la creación y gestión de objetos en la BD.

Sus tres comandos o verbos principales son:

a) CREATE. Para la creación de objetos

b) DROP. Para el borrado o eliminación de objetos

c) ALTER. Para modificar los objetos

## -DCL (Lenguaje de Control de Datos)

Es el lenguaje que se utiliza para la gestión de permisos sobre los objetos del sistema

Sus comandos principales son:

a) GRANT. Para otorgar permisos o privilegios

b) REVOKE. Para denegarlos.

## -DML (Lenguaje de Manipulación de Datos)

Es el lenguaje que se utiliza para la gestión de los datos

Sus comandos principales son:

a) INSERT. Para añadir datos.

b) DELETE. Para borrar datos.

c) UPDATE. Para modificar datos.

d) SELECT. Para consultar datos.

# 11 – Crear y borrar una BD

create database nombre\_bd;

drop database nombre\_bd;

# 12 – Crear un usuario

Puede hacerse con un usuario que tenga permiso para ello (root)

create user nombre\_usuario identified by ‘contraseña’ ;

create user alfonso identified by ‘alfonso’ ;

# 13 – Borrar usuario

drop user nombre\_usuario;

# 14 – Creación de tablas o relaciones

En primer lugar debemos conocer los tipos de datos disponibles en mysql. Podemos consultar el manual o webs como w3schools.

## a) Tipos de datos caracteres

-CHAR. Para un solo caracter.

-VARCHAR (longitud). Son cadenas de caracteres de hasta 65535 caracteres.

-ENUM (‘valor1’, ‘valor2’,…, ‘valorN’). Sólo permite introducir uno de los valores que están entre paréntesis.

## b) Tipos de datos numéricos

-INTEGER. Números enteros (sin decimales).

-DECIMAL (tamaño, decimales). Tamaño indica el nº total de dígitos del número y decimales, el número de decimales.

## c) Tipos de datos tiempo

-DATE. Para fechas con el formato YYYY-MM-DD

-TIME. Para horas con el formato hh:mm:ss

-TIMESTAMP. Indica un instante y está compuesto por la fecha y por la hora con el formato YYYY-MM-DD hh:mm:ss

## d) Datos de tipo lógico o booleano

-BOOLEAN. En los que 0 equivale a FALSO y 1 equivale a VERDADERO

# Crear una nueva relación o tabla

El comando genérico sería:

CREATE TABLE nombre\_tabla

(

nombre\_atributo1 tipo\_dato [restricción semántica],

nombre\_atributo2 tipo\_dato [restricción semántica],

……..

nombre\_atributoN tipo\_dato [restricción semántica]

);

# RESTRICCIONES SEMÁNTICAS

(en SQL se llaman CONSTRAINTS)

-PRIMARY KEY

-NOT NULL

-UNIQUE

-CHECK

-FOREIGN KEY

# Borrar una relación o tabla

drop table nombre\_tabla;

# Modificar una relación o tabla existente

## a) Añadir un nuevo atributo a una tabla existente

ALTER TABLE nombre\_tabla ADD nombre\_atributo tipo\_dato [restricción];

ALTER TABLE empleado ADD edad INTEGER NOT NULL;

## b) Borrar un atributo existente

ALTER TABLE nombre\_tabla DROP nombre\_atributo;

## c) Modificar un atributo existente

ALTER TABLE nombre\_tabla MODIFY nombre\_atributo tipo\_dato [restricción];

ALTER TABLE empleado MODIFY dni VARCHAR(9);

# CONSTRAINTS

Existen restricciones que pueden indicarse en la misma línea de declaración del atributo y otras que requieren su definición mediante un CONSTRAINT

Todo CONSTRAINT tienen el mismo esquema:

CONSTRAINT nombre\_del\_constraint

(NO PUEDE HABER DOS CONSTRAINT CON EL MISMO NOMBRE EN LA MISMA BD)

(

cuerpo\_del\_constraint

);

# Situaciones en las que debemos usar un CONSTRAINTS

## a) Cuando la PK (Primary Key) de una relación está formada por más de un atributo

CONSTRAINT pk\_nombretabla

PRIMARY KEY( at1, at2,…, atN)

CREATE TABLE vivienda

(

calle VARCHAR(30),

extension INTEGER NOT NULL,

dormitorios INTEGER NOT NULL,

CONSTRAINT pk\_vivienda

PRIMARY KEY(calle, numero)

);

## b) Para la realización de chequeos o validaciones

CONSTRAINT ck\_nombreatributo

CHECK(condición o predicado)

| Vamos a modificar la tabla ‘vivienda’ añadiendo 2 chequeos (que la extensión sea mayor que 0 y que el número de dormitorios sea mayor que 1) |
| --- |
| ALTER TABLE vivienda ADD CONSTRAINT ck\_extension CHECK (extension>0);  ALTER TABLE vivienda ADD CONSTRAINT ck\_dormitorios CHECK(dormitorios>1); |

## c) Declaración de FK (Foreign Key)

CONSTRAINT fk\_tablaactual\_tablareferenciada

FOREIGN KEY (at1, at2, …, atN)

REFERENCES nombretablareferenciada(at1, at2, …, atN)

[ON DELETE NO ACTION | CASCADE | SET NULL | SET DEFAULT]

[ON UPDATE NO ACTION | CASCADE | SET NULL | SET DEFAULT]

| Crear una tabla ‘nómina’ relacionada con la tabla ‘empleado’  dni → PK, FK, OUC, ODC  mes → PK  año → PK  total → NN |
| --- |
| CREATE TABLE nomina  (  empleado VARCHAR(9),  mes ENUM (‘ENE’, ‘FEB’, ‘MAR’, ‘ABR’, ‘MAY’, ‘JUN’, ‘JUL’, ‘AGO’, ‘SEP’, ‘OCT’, ‘NOV,’ ‘DIC’),  agno INTEGER,  total DECIMAL(7,2) NOT NULL,  CONSTRAINT pk\_nomina  PRIMARY KEY(empleado, mes, agno),  CONSTRAINT fk\_nomina\_empleado  FOREIGN KEY(empleado)  REFERENCES empleado(dni)  ON DELETE CASCADE  ON UPDATE CASCADE  ); |
| Otra posibilidad  CREATE TABLE nomina  (  empleado VARCHAR(9),  mes INTEGER  agno INTEGER,  total DECIMAL(7,2) NOT NULL,  CONSTRAINT pk\_nomina  PRIMARY KEY(empleado, mes, agno),  CONSTRAINT fk\_nomina\_empleado  FOREIGN KEY(empleado)  REFERENCES empleado(dni)  ON DELETE CASCADE  ON UPDATE CASCADE,  CONSTRAINT ck\_mes  CHECK(mes>=1 and mes<=12)  ); |

\*\*\*UN CASO ESPECIAL: La declaración de una clave alterna o alternativa\*\*\*

Las claves alternas son claves candidatas que no han sido elegidas como PK.

Llevan asociada las restricciones semánticas UNIQUE y NOT NULL

| Añadir una clave alterna a la tabla ‘vivienda’ llamada ‘ref\_catastral’ |
| --- |
| ALTER TABLE vivienda ADD ref\_catastral VARCHAR(15) UNIQUE NOT NULL; |

# LOS SCRIPT SQL

Un script SQL es un fichero de texto plano (codificación utf-8) que guardamos con la extensión .sql

Se pueden crear con editores sencillos como:

-En Linux: gedit, nano, vi

-En Windows: notepad o bloc de notas, notepad++

Una vez creado el script el comando para su ejecución es:

source ruta script.sql;

# Cláusula IF NOT EXISTS

Se utiliza para evitar la creación de objetos que ya hayan sido previamente creados.

La sintaxis es:

CREATE tipo\_objeto IF NOT EXISTS nombre\_objeto resto\_de\_opciones\_del\_objeto

*Para una tabla*

CREATE TABLE IF NOT EXISTS nomina

(

empleado VARCHAR(9),

mes INTEGER

agno INTEGER,

total DECIMAL(7,2) NOT NULL,

CONSTRAINT pk\_nomina

PRIMARY KEY(empleado, mes, agno),

CONSTRAINT fk\_nomina\_empleado

FOREIGN KEY(empleado)

REFERENCES empleado(dni)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT ck\_mes

CHECK(mes>=1 and mes<=12)

);

*Para una BD*

CREATE DATABASE IF NOT EXISTS nombre\_bd;

# Usuarios de MYSQL

Los nombres de los usuarios de mysql se componen de dos partes:

nombre\_usuario@nombre\_o\_ip\_del\_host

Donde el nombre o ip del host indica desde donde se puede conectar el usuario. Si no se indica nada, el usuario puede conectarse desde cualquier ip.

En nuestro caso hemos deshabilitado el acceso remoto de manera que solo nos conectamos desde localhost o en modo local.

Por lo tanto, no indicaremos host a la hora de crear un usuario.

Lo haremos:

CREATE USER IF NOT EXIST nombre\_usuario IDENTIFIED BY ‘contraseña’;

# LENGUAJE DE CONTROL DE DATOS (DCL)

Es el lenguaje que se utiliza para la gestión de permisos y sus dos verbos principales son:

a) GRANT. Para otorgar permisos.

b) REVOKE. Para retirarlos.

En primer lugar debemos distinguir entre:

-Privilegios. Se trata de un permiso en un ámbito en concreto.

-Roles. Contienen un conjunto de permisos asociados a un tipo de usuario. Existen roles predefinidos pero también podemos crear nuestros propios roles.

PERMISOS O PRIVILEGIOS

Existen diferentes tipos:

-Permisos para crear usuarios

-Permisos a nivel global

-Permisos a nivel de BD

-Permisos a nivel de tabla

-Permisos a nivel de columna

-Permisos a nivel de procedimiento y funciones almacenados

ROLES

Tenemos una serie de roles predefinidos:

-DBA. Equivaldría a convertirlo en root.

-UserAdmi. Permite la gestión completa de usuarios.

-DBManager. Otorga todos los privilegios sobre todas las BBDD del sistema.

También es posible la creación de roles personalizados

## EL VERBO GRANT

GRANT privilegios o roles ON objetos TO usuarios [WITH GRANT OPTION];

La cláusula WITH GRANT OPTION, si se indica, significa que el usuario que recibe los roles o privilegios puede a su vez otorgarlos a otros usuarios.

Nosotros vamos a trabajar exclusivamente con:

-Privilegios DML: INSERT, DELETE, UPDATE y SELECT

-Todos los privilegios: ALL PRIVILEGES

| Supongamos que tenemos una base de datos llamada ‘mibd’ con tres tablas (tabla1, tabla2, tabla3)  Supongamos que tenemos dos usuarios llamados ‘miusuario’ y ‘usuario2’  Otorgar todos los privilegios al usuario ‘miusuario’ sobre todas las tablas de la BD ‘mibd’ |
| --- |
| GRANT ALL PRIVILEGES ON mibd.\* TO miusuario; |
| Otorgar todos los privilegios al usuario ‘miusuario’ sobre todas las tablas de todas las BBDD del servidor. |
| GRANT ALL PRIVILEGES ON \*.\* TO miusuario; |
| Otorgar al usuario ‘usuario2’ privilegio de consulta (solo consulta) sobre la tabla1 y sobre la tabla2 y tabla3 los de consulta e inserción de datos |
| GRANT SELECT ON mibd.tabla1 TO usuario2;  GRANT SELECT, INSERT ON mibd.tabla2 TO usuario2;  GRANT SELECT, INSERT ON mibd.tabla3 TO usuario2; |

NOTA IMPORTANTE: Cada vez que otorgamos o revoquemos privilegios tenemos que confirmarlo con la orden

FLUSH PRIVILEGES;

## EL VERBO REVOKE

REVOKE privilegios o roles, [GRANT OPTION] ON objetos FROM usuarios;

# PROCESO PARA LA CREACIÓN DE BASES DE DATOS

1.- Creamos la BD

2.- Creamos un usuario para esa BD

3.- Otorgamos a ese usuario, todos los privilegios

4 Grabamos los privilegios (FLUSH)

5 Ejecutamos el script de creación de las tablas de la BD

# LENGUAJE DE MANIPULACIÓN DE DATOS (DML)

## AÑADIR DATOS – VERBO INSERT

INSERT INTO nombre\_tabla VALUES (lista\_de\_valores\_ordenada);

-La sentencia anterior inserta una fila en la tabla

-La lista de valores debe ir en el mismo orden en el que los atributos están definidos en la tabla

-Los valores se introducen:

a) Las cadenas de caracteres entre comillas simples ‘ ‘

b) Los números sin comillas y usaremos el punto del teclado numérico para los decimales

c) Los datos de tipo tiempo se introducen también entre comillas siguiendo el formato de cada uno

-DATE: ‘yyyy-mm-dd’

-TIME: ‘hh:mm:ss’

También es posible la inserción masiva de datos, es decir, insertar varias filas en una única sentencia INSERT

INSERT INTO nombre\_tabla (at1, at2, …, atn) VALUES

(valor1, valor2, …, valorn),

(valor1, valor2, …, valorn),

(valor1, valor2, …, valorn),

…………

(valor1, valor2, …, valorn);

La inserción de datos, como otras operaciones DML, debe tener en cuenta una serie de consideraciones

a) Los valores se deben corresponder en tipo a los definidos en la tabla

b) El orden de la inserción debe respetar el Modelo Lógico, como ocurría en la creación

c) Además los valores introducidos en las tablas tienen que satisfacer obligatoriamente las restricciones de integridad del Modelo

## BORRAR DATOS – VERBO DELETE

DELETE FROM nombre\_tabla [WHERE condición o predicado];

\*\*\*IMPORTANTE\*\*\* Si no se indica la cláusula WHERE, se borrarán todos los datos

DELETE FROM tabla1;

(borraría todos los datos de la tabla1)

TRUNCATE TABLE tabla1;

(borraría todos los datos de la tabla1)

La cláusula WHERE en general es:

WHERE atributo(operador de comparación)valor [operador lógico + comparación] …

WHERE predicado simple o complejo

WHERE laboratorio=’lab3’;

## MODIFICAR DATOS – VERBO UPDATE

UPDATE nombre\_tabla SET at1=valor1 [,at2=valor2…] [WHERE condición]

Si no se especifica la condición se actualizan todas las filas de la tabla

## CONSULTA DE DATOS – VERBO SELECT

### Consultas básicas sobre una única tabla

SELECT operador\_proyección

FROM nombre\_tabla

[WHERE condición] (operador selección)

[ORDER BY nombre\_atributo];

### Operador proyección

Sirve para indicar las columnas de las tablas que queremos obtener

- \* Obtenemos todas las columnas de la tabla

- at1 [,at2, …, atn]

b) FROM indica la tabla desde la que queremos consultar

c) Operador selección. Devuelve las filas que cumplen la condición

d) ORDER BY ordena el resultado

| Obtener todos los datos de una tabla |
| --- |
| SELECT \*  FROM temple; |
| Obtener sólo algunos datos |
| SELECT nomem, fecna  FROM temple; |

| Uso de alias. Podemos renombrar las columnas de la tabla para la salida |
| --- |
| -Si el alias está formado por una única palabra  SELECT nomem as Nombre, fecna as Nacimiento  FROM temple; |
| -Si el alias está formado por más de una palabra  SELECT nomem as ‘Nombre del empleado’, fecna as ‘Fecha de nacimiento’  FROM temple; |

### Operador selección

Devuelve sólo las filas que cumplen el predicado

Predicado simple. Expresión con un operador relacional o de comparación

SELECT \*

FROM temple

WHERE numde=100;

SELECT \*

FROM temple

WHERE nomem=’LOPEZ, ANTONIO’;

Predicados compuestos. Unen más de una condición con operadores lógicos (and, or)

SELECT \*

FROM temple

WHERE numde=111 and numhi>1;

### ORDER BY

*ascendente*

SELECT \*

FROM temple

WHERE numde=111

ORDER BY numhi;

*descendiente*

SELECT \*

FROM temple

WHERE numde=111

ORDER BY numhi DESC;

*Varios atributo*

SELECT \*

FROM temple

WHERE numde=111

ORDER BY numhi DESC, salar;

### Cláusula DISTINCT

Se utiliza en la proyección para eliminar filas repetidas

SELECT numde,

FROM temple

WHERE numhi=0

ORDER BY numde;

SELECT DISTINCT numde

FROM temple

WHERE numhi=0;

### Expresiones

Podemos proyectar atributos de una tabla o expresiones basadas en esa tabla. Normalmente al proyectar una expresión, usaremos un alias.

| Proyectar número de departamento , num de empleado y el salario, nombre del empleado de los trabajadores que trabajan en el departamento del 111. |
| --- |
| SELECT numde, numem, nomem, salar, salar\*0.1 as ‘Paga extra’  FROM temple  WHERE numde=111  ORDER BY numem; |

| La paga extra va a ser igual a 10% del salario por cada hijo a partir del segundo. |
| --- |
| SELECT numde, numem, nomem, salar, (numhi-1)\*salar\*0.1 as ‘Paga extra’  FROM temple  WHERE numde=111 and numhi>0  ORDER BY numem; |

### Predicado NULL

expresión IS [NOT] NULL

SELECT \*

FROM temple

WHERE comis IS NOT NULL;

### Predicado BETWEEN

expresión1 [NOT] BETWEEN expresión2 AND expresión3;

SELECT \*

FROM temple

WHERE salar BETWEEN 1500 AND 2000

ORDER BY salar;

SELECT \*

FROM temple

WHERE salar NOT BETWEEN 1500 AND 2000

ORDER BY salar;

### Predicado IN

expresión [NOT] IN (lista de valores);

Comprueba si la expresión es igual, o no, a alguno de los varios de la lista.

SELECT \*

FROM temple

WHERE numde IN (100, 110, 111)

ORDER BY numde;

### Predicado LIKE

expresión\_alfanumérica [NOT] LIKE cadena\_caracteres

SELECT \*

FROM temple

WHERE nomem=’LOPEZ,ANTONIO’ ;

SELECT \*

FROM temple

WHERE nomem LIKE ’LOPEZ,ANTONIO’ ;

Permite el uso de metacaracteres

Metacarácter ( \_ ) Equivale a un único carácter

Metacarácter ( % ) Equivale a una cadena de caracteres

Nombre empieza por a:

SELECT \*

FROM temple

WHERE nomem LIKE ‘%,A%’ ;

Segunda letra del nombre es a:

SELECT \*

FROM temple

WHERE nomem LIKE ‘%,\_A%’ ;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### TRATAMIENTO DE VALORES NULOS

La construcción de predicados compuestos debe tener en cuenta la posibilidad de que alguno de los operandos o predicados simples, sean valores nulos.

(VER IMÁGEN DEL PDF DEL AULA VIRTUAL)

En el PDF ‘Predicados’ que está en el aula virtual se detallan todos los casos

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SQL estándar (versión del 92) incorpora dos funciones escalares que podemos usar con cadenas

- LENGTH(expresión) . Me devuelve la longitud de la expresión. Si la expresión es una cadena de caracteres, me devuelve el número de caracteres. Si es un número, me devuelve el número de bytes que internamente se utilizan para almacenarlo

SELECT nomem, LENGTH(nomem)

FROM temple

WHERE numde=111

ORDER BY 1;

- SUBSTR(cadena, posición, longitud) . Devuelve una subcadena de una cadena. Donde:

a) cadena. Es la cadena de la que partimos

b) posición. Es la posición en la cadena en la que comienza la subcadena a extraer

c) longitud. Es el número de caracteres o tamaño de la subcadena a extraer

SELECT SUBSTR(nomem,9,5)

FROM temple

WHERE nomem LIKE ‘AGUIRRE,AUREO’;

### ANEXO: Funciones de cadena de mysql (no están en el estándar)

* CONCAT (cadena1, cadena2 [, cadena3, … , cadenan];

Une las cadenas que se le pasan.

* LOCATE (subcadena, cadena, [posición]);

Busca la primera ocurrencia de la subcadena, dentro de la cadena, a partir de la posición opcional (si no se indica, desde el principio)

| Extraer los nombres de pila de lo empleados |
| --- |
| SELECT nomem, SUBSTR(nomem, LOCATE(‘,’, nomem)+1, (LENGTH(nomem)-LOCATE(‘,’, nomem))) AS Apellido  FROM temple  ORDER BY 1; |

| Obtener el nombre completo, solo el apellido, solo el nombre de los empleados |
| --- |
| SELECT nomem AS ‘Nombre completo’,  SUBSTR(nomem, 1, LOCATE(‘,’, nomem)-1) AS Apellido,  SUBSTR(nomem, LOCATE(‘,’, nomem)+1, (LENGTH(nomem)-LOCATE(‘,’, nomem))) AS Nombre  FROM temple  ORDER BY 1; |

| Obtener por orden alfabético, nombre y salario de los empleados que tengan más de 3 hijos |
| --- |
| (NOTA: Nombre y apellidos)  SELECT nomem, salar  FROM temple  WHERE numhi>3  ORDER BY nomem; |

| Obtener los números de departamento ordenados donde trabajan empleados cuyo salario sea inferior a los 1500 euros (1 euro = 166,38 pesetas) |
| --- |
| SELECT DISTINCT numde  FROM temple  WHERE salar<1500; |

| Obtener por orden alfabético el nombre de los departamentos cuyo director (tdir) lo es en funciones (F) |
| --- |
| SELECT nomde  FROM tdepto  WHERE tidir=’F’  ORDER BY nomde; |

| Suponiendo que en los próximos 3 años el coste de la vida va aumentar en un 6% por año y que se van a subir los salarios en la misma proporción, obtener por orden alfabético nombre de empleado, sueldo anual actual (14 pagas) y el que tendrá en cada uno de los próximos años de los empleados que tengan más de 4 hijos |
| --- |
| (NOTA: Nombre y apellidos)  SELECT nomem, salar\*14 AS ‘Sueldo anual actual’,  (salar+(salar\*0.06))\*14 AS ‘Agno siguiente’,  (salar+(salar\*0.12))\*14 AS ‘Segundo agno’,  (salar+(salar\*0.18))\*14 AS ‘Tercer agno’  FROM temple  WHERE numhi>4  ORDER BY nomem; |

| Obtener los números de los departamentos en los que haya un empleado cuya comisión sea superior al 20% de su salario. |
| --- |
| SELECT DISTINCT numde  FROM temple  WHERE comis>(salar\*0.2); |

* REVERSE (cadena);

Escribe la cadena al revés

### Predicado BETWEEN

valor1 [NOT] BETWEEN valor2 AND valor3

1.- valor1 BETWEEN valor2 AND valor3

Si alguno de los valores v1, v2, v3 es NULO -> NULO

2.- valor1 NOT BETWEEN valor2 AND valor3

1. Si v1 es NULL -> NULO
2. Si v1 no es NULO

* Si v2 y v3 son NULO -> NULO
* Si v2 no es NULO y v3 es NULO -> Si v1 < v2 -> v
* Si v2 es NULO y v3 no es NULO -> Si v1 > v3 -> v

### PREDICADOS Y SUBCONSULTAS

Una subconsulta es una consulta dentro de otra, concretamente en la proyección (WHERE). Además, las consultas pueden anidarse.

#### Subconsultas que devuelven un único valor

Podemos construir predicados de la siguiente forma genérica:

SELECT proyección

FROM tabla

WHERE expresión (operador relacional >, <, =, …) (SELECT proyección

FROM tabla

[WHERE condición])

ORDER BY at;

Donde la subconsulta devuelve un único valor

| Obtener todos los datos de los empleados que son más jóvenes que GARCIA,AUGUSTO |
| --- |
| SELECT \*  FROM temple  WHERE fecna > (SELECT \* FROM temple WHERE nomem LIKE ‘GARCIA,AUGUSTO’)  ORDER BY fecna; |

| Obtener todos los datos de los empleados que tienen más hijos que GIL,GLORIA(3 hijos) |
| --- |
| SELECT \*  FROM temple  WHERE numhi > (SELECT numhi FROM temple WHERE nomem LIKE ‘GIL,GLORIA’)  ORDER BY numhi; |

#### Subconsultas que devuelven más de un valor

##### Predicados cuantificados

Utilizan las palabras reservadas:

1. ALL. Si el predicado se cumple para todos los valores que devuelven la subconsulta.
2. SOME o ANY. Comprueba si el predicado se cumple para alguno de los valores que devuelve la subconsulta.

| Obtener los datos de los empleados que tienen un salario superior a cualquier empleado del departamento 110. |
| --- |
| SELECT \*  FROM temple  WHERE salar > ALL (SELECT salar FROM temple WHERE numde=110)  ORDER BY salar; |

| Obtener todos los datos de los empleados que hayan ingresado en la empresa después de alguno de los empleados del departamento 110. |
| --- |
| SELECT \*  FROM temple  WHERE fecin > SOME (SELECT fecin FROM temple WHERE numde=110)  ORDER BY fecin; |

| Obtener todos los datos de los empleados cuyo salario sea 3 veces superior al salario de TORRES,HORACIO |
| --- |
| SELECT \*  FROM temple  WHERE salar > (SELECT salar\*3 FROM temple WHERE nomem LIKE ‘TORRES,HORACIO’)  ORDER BY salar; |

| Obtener por orden alfabético el nombre y los salarios de los empleados cuyo salario es inferior al cuádruple de la comisión más baja existente. |
| --- |
| SELECT nomem, salar  FROM temple  WHERE salar < ALL (SELECT comis\*4 FROM temple WHERE comis IS NOT NULL)  ORDER BY nomem; |

##### CASOS ESPECIALES

1.- Si la subconsulta devuelve una tabla vacía

El predicado es verdadero con ALL

El predicado es falso con SOME o ANY

2.- Si la subconsulta devuelve valores NULOS o valores válidos mezclados con valores NULOS

* Con ALL :
* Será falso si es falso para uno de ellos
* Si es verdadero para todos los valores válidos, será nulo
* Con SOME :
* Será verdadero, si el predicado es verdadero para al menos uno de ellos
* Si es falso para todos los valores válidos, será nulo

3.- Si la subconsulta devuelve sólo valores NULOS entonces es NULO(similar a falso) en todos los casos

| Obtener por orden de número de empleado el nombre y el salario total(salario + comisión) de los empleados cuyo salario total supere los 1800 euros |
| --- |
| SELECT numem, nomem, salar+comis AS ‘Salario total’  FROM temple  WHERE salar+comis > 1800  ORDER BY numem; |

| Obtener por orden alfabético los nombres y el salario de aquellos cuyos salarios supere al máximo salario del departamento 122 |
| --- |
| SELECT nomem, salar  FROM temple  WHERE salar > ALL (SELECT salar FROM temple WHERE numde=122)  ORDER BY nomem; |

| Obtener por orden alfabético los nombres de los empleados cuyo apellido tenga exactamente 7 letras |
| --- |
| SELECT nomem  FROM temple  WHERE nomem LIKE ‘\_\_\_\_\_\_\_,%’  ORDER BY nomem;  NOTA: (Hay siete barra bajas después del LIKE) |

| Obtener los nombre, salarios y fecha de ingreso en la empresa de los empleados que ingresaron después del uno de enero del 1988 o bien que tiene un salario inferior a 1200 euros |
| --- |
| SELECT nomem, salar, fecin  FROM temple  WHERE fecin > ‘1988-01-01’ OR salar < 1200  ORDER BY nomem; |

| Obtener por fecha de nacimiento todos los datos de los empleados que son más jóvenes que cualquier empleado del departamento 100 y que tienen comisión |
| --- |
| SELECT \*  FROM temple  WHERE fecna > ALL (SELECT fecna FROM temple where numde=100) AND comis IS NOT NULL)  ORDER BY fecna DESC; |

##### Predicado IN

Comprueba si la expresión es igual a alguno de los valores de la subconsulta.

SELECT proyección

FROM tabla

WHERE expresión IN (SELECT proyección FROM tabla [WHERE condición];

| Obtener por orden alfabético los nombres de los empleados que trabajan en el mismo departamento de Flos, Dorotea y Gálvez, Pilar |
| --- |
| SELECT nomem  FROM temple  WHERE numde IN (SELECT numde FROM temple WHERE nomem LIKE (‘GALVEZ,PILAR’, ‘FLOR,DOROTEA’))  ORDER BY nomem;  La consulta anterior es igual a la siguiente:  SELECT nomem  FROM temple  WHERE numde = ANY (SELECT numde FROM temple WHERE nomem LIKE (‘GALVEZ,PILAR’, ‘DOROTEA,FLOS’))  ORDER BY nomem; |

Si la expresión es NULL, el predicado es NULL

Si la subconsulta devuelve una tabla vacía, el predicado es falso

##### Predicado EXISTS

Comprueba si la subconsulta devuelve algún valor, en cuyo caso es verdadero, y si no devuelve algún valor, entonces es falso. Nunca devuelve nulo.

SELECT proyección

FROM tabla

WHERE EXISTS (SELECT proyección FROM tabla [WHERE condición]);

| Obtener los nombres de los centros donde existan departamentos con tipo de dirección en funciones (tidir=’F’) |
| --- |
| SELECT nomce  FROM tcentr  WHERE EXISTS (SELECT numce FROM tdepto WHERE tidir LIKE ‘F’); |

| Obtener por orden alfabético los nombres de los empleados cuya extensión telefónica es 250 o 750 |
| --- |
| SELECT nomem  FROM temple  WHERE extel=250 OR extel=750  ORDER BY nomem; |

| Obtener por orden alfabético los nombre y los salarios que, o bien no tiene hijos y gana más de 1200 euros, o bien tienen hijos y ganan menos de 1800 |
| --- |
| SELECT nomem, salar  FROM temple  WHERE (numhi=0 AND salr>1200) OR (numhi>0 AND salar<1800)  ORDER BY nomem; |

| Obtener todos los datos de los empleados que nacieron en el año 1965 y que tienen menos hijos que cualquier empleado del departamento 100 |
| --- |
| SELECT \*  FROM temple  WHERE fecna LIKE ‘1965%’ AND numhi < ALL (SELECT numhi FROM temple WHERE numde=100); |

| Obtener todos los datos de lo empleados cuyo nombre de pila esté formado por más de 5 letras y que tengan un salario superior a cualquier empleado que trabaje en el departamento 111 |
| --- |
| SELECT \*  FROM temple  WHERE nomem LIKE ‘%,\_\_\_\_\_’ AND salar > ALL (SELECT salar FROM temple WHERE numde=111);  NOTA: (En la parte de “ LIKE ‘%,\_\_\_\_\_’ “ hay 5 barras bajas) |

| Obtener por orden alfabético todos los datos de los empleados que, o bien no tiene hijos, o bien su salario por hijo supere los 600 euros si hay alguno sin comisión en el departamento 111 o 112 |
| --- |
| SELECT \*  FROM temple  WHERE (numhi= 0) OR  ORDER BY nomem; |

Ejercicio

examen 2021

bloque 2

2.1 NO, no tiene on update cascade

2.2 Sí, ALTER TABLE categoria ADD descripcion VARCHAR(60);

2.3 Sí, UPDATE categoria SET decripcion=’Camisetas’ WHERE nombre LIKE ’Manga%’;

2.4 Sí, ALTER TABLE contiene ADD CONSTRAINT ck\_unidades CHECK(unidades>0);

2.5 No puede realizarse si hay un usuario en la tabla user y tenga un pedido y usuario de la tabla pedido no tiene borrado en cascada.

# DATOS DE TIPO TIEMPO

Tres tipos de datos tiempo:

1. Fecha (DATE)

En mysql su formato general es ‘yyyy-mm-dd’

1. Hora (TIME)

En mysql su formato general es ‘hh:mm:ss’

1. Instante (STAMP)

En mysql su formato es ‘yyyy-mm-dd hh:mm:ss’

Para trabajar con datos de tipo tiempo, en realidad usamos una cadena de caracteres que se ajusta a unos determinados formatos o patrones establecidos.

Cuando introducimos datos de tiempo de manera explícita, en realidad lo que hacemos es introducir una cadena de caracteres cuyo formato admite el SGBD como un dato de tipo tiempo.

Ello supone, además, que puedo realizar operaciones de comparación ( <, >, =, … ) entre un dato de tipo tiempo almacenado en la BD y una cadena de caracteres que cumpla en el formato.

## REGISTROS ESPECIALES

Existen dos registros especiales de tiempo, que también se pueden tratar como funciones.

CURRENT\_DATE () Devuelve la fecha actual del sistema

CURRENT\_TIME () Devuelve la hora

En mysql, además, tenemos:

CURRENT\_TIMESTAMP() Devuelve el instante actual

## Funciones que operan con datos de tipo tiempo

### 1.- Funciones que permiten descomponer un dato tiempo

YEAR (expresión)

MONTH (expresión)

DAY (expresión)

HOUR (expresión)

MINUTE (expresión)

SECOND (expresión)

| Obtener los datos de los empleados que nacieron en la década de los 60 |
| --- |
| SELECT \*  FROM temple  WHERE year(fecna) between 1960 and 1969  ORDER BY fecna; |

### 2.- Funciones para formatear datos tiempo en mysql

DATE\_FORMAT (fecha, formato)

* fecha es un dato fecha almacenado o una cadena con formato válido
* formato que se va a usar

TIME\_FORMAT(hora, formato)

* hora es un dato hora almacenado o una cadena con formato válido
* formato que se va a usar

| Pasar las fechas a nuestro formato habitual |
| --- |
| SELECT nomem, time\_format(fecna, ‘%d/%m/%Y’) as Nacimiento  FROM temple  ORDER BY 2; |

### 3.- Aritmética de fechas

Un valor de tipo tiempo puede restarse de otro o puede incrementar o decrementar con una duración PERO EL RESULTADO OBTENIDO ES UNA DURACIÓN o un número de días en otros sistemas

#### 3.1.- Incrementar o decrementar un dato de tipo tiempo

1. Incrementar y decrementar una fecha

DATE\_ADD(fecha, INTERVAL valor intervalo). Suma a una fecha una duración indicado por un intervalo

* fecha. Es una fecha
* INTERVAL. Es una palabra reservada del lenguaje que hay que poner obligatoriamente
* valor. Es el número de los intervalos que se le suma a la fecha
* intervalo. Indica qué tipo de intervalo se le va a sumar (year, month, week, day, second, minute, …) OJO!!!! en singular

| Añadir 20 años a la edad de nacimiento de los empleados |
| --- |
| UPDATE temple SET fecna=DATE\_ADD(fecna, INTERVAL 20 YEAR),  fecin=DATE\_ADD (fecin, INTERVAL 20 YEAR); |

DATE\_SUB(fecha, INTERVAL valor intervalo). Es análoga a la anterior, pero en este caso resta una duración a la fecha dada

#### 3.2.- Diferencia entre datos de tipo tiempo

1. DATEDIFF(fecha1, fecha2). Devuelve el número de días entre dos valores de tipo fecha (fecha1- fecha2)

* fecha1 es obligatorio y debe ser la más reciente
* fecha2 es obligatorio

1. TIMESTAMPDIFF(intervalo, dato\_tiempo1, dato\_tiempo2). Devuelve la diferencia en intervalos entre la fecha1 y la fecha2

* intervalo. Es uno de los intervalos usados en las funciones DATE\_ADD y DATE\_SUB
* dato\_tiempo1. Es obligatorio y representa el más antiguo
* dato\_tiempo2. Es obligatorio y representa el más actual

| Vamos a calcular la edad y la antigüedad en la empresa de los empleados |
| --- |
| SELECT nomem, TIMESTAMPDIFF(year, fecna, current\_date) AS Edad, TIMESTAMPDIFF(year, fecin, current\_date) AS Antiguedad  FROM temple; |

| Vamos a mostrar los datos de los empleados con los siguientes criterios:   * Vamos a separar nombres y apellidos * Las fechas de nacimiento e ingreso las vamos a mostrar en formato europeo * Vamos añadir la edad y la antigüedad |
| --- |
| SELECT SUBSTR(nomem, LOCATE(‘,’, nomem)+1), (LENGTH(nomem)-LOCATE(‘,’, nomem))) AS Nombre,  SUBSTR(nomem, 1, LOCATE(‘,’, nomem) - 1 ) AS Apellido,  DATE\_FORMAT(fecna, ‘%d/%m/%Y’) AS ‘Fecha de nacimiento’,  TIMESTAMPDIFF(year, fecna, current\_date) AS Edad, DATE\_FORMAT(fecin, ‘%d/%m/%Y’) AS ‘Fecha ingreso’,  TIMESTAMPDIFF(year, fecin, current\_date) AS Antiguedad  FROM temple; |

b

| Obtener todos los datos de los empleados que nacieron durante el segundo/último semestre |
| --- |
| SELECT \*  FROM temple  WHERE month(fecna) between 7 and 12; |

| ¿Hay algún empleado que cumpla años hoy? |
| --- |
| SELECT \*  FROM temple  WHERE day(fecna)=7 and month(fena)=2;  SELECT \*  FROM temple  WHERE day(fecna)=day(current\_date) AND month(fecna)=month(current\_date); |

| Obtener todos los datos de los empleados que nacieron en los meses pares de la década de los 60 |
| --- |
| SELECT \*  FROM temple  WHERE (year(fecna) BETWEEN 1960 AND 1969) AND  (month(fecna) IN (2, 4, 6, 8, 10, 12))  ORDER BY fecna; |

| Le vamos a dar una paga extra a los empleados por su antigüedad. 2000 euros por cada 10 años completos de antigüedad |
| --- |
| SELECT nomem, TIMESTAMPDIFF(year, fecin, CURRENT\_DATE) AS Antigüedad,  TRUNCATE(TIMESTAMPDIFF(year, fecin, CURRENT\_DATE) / 10, 0) \* 2000 AS Paga  FROM temple  ORDER BY fecin; |

| Obtener todos los datos de los empleados que son mayores que cualquier empleado del departamento 111 |
| --- |
| SELECT nomem, TIMESTAMPDIFF(year, fecna, CURRENT\_DATE) AS Edad  FROM temple  WHERE TIMESTAMPDIFF(year, fecna, CURRENT\_DATE) > ALL(SELECT TIMESTAMPDIFF(year, fecna, CURRENT\_DATE) FROM temple WHERE numde=111)  ORDER BY 2; |

# CONSULTAS CON AGRUPAMIENTO DE FILAS

## 1 - Agrupamiento implícito. Funciones colectivas o de columna

Se crea un grupo sin necesidad de indicarlo, con todas las filas que satisfacen la consulta.

A ese grupo se le aplica una función que realiza un determinado cálculo.

Estas funciones se llaman colectivas ya que operan sobre un conjunto de valores.

Y se llaman también de columna porque operan sobre una de las columnas de la consulta, o bien sobre una expresión basada en una de las columnas.

Son las siguientes:

AVG (valores) - Devuelve la media de los valores.

MAX (valores) - Devuelve el máximo de los valores.

MIN (valores) - Devuelve el mínimo de los valores

SUM (valores) - Devuelve la suma de los valores.

COUNT(valores) - Devuelve el número de valores.

### Reglas de uso

* Antes de aplicar cualquiera de estas funciones a los valores, si éstos incluyen NULOS, se eliminan antes de su operación
* Las funciones MAX, MIN y SUM devuelve el mismo tipo de dato que sus valores
* La función AVG siempre devuelve un valor decimal

### Formatos de uso

* Para todas las funciones

nombre\_funcion ([DISTINCT] nombre\_columna)

SELECT max(salar), min(salar), avg(salar), count(DISTINCT comis)

FROM temple

Si se indica DISTINCT, antes de la operación se eliminan los valores duplicados

* Para todas las funciones menos para COUNT

nombre\_funcion (expresión)

select max(TIMESTAMPDIFF(year, fecna, CURRENT\_DATE))

FROM temple;

SELECT MAX(TIMESTAMPDIFF(year, fecna, CURRENT\_DATE)) AS ‘Más veterano’, MIN(TIMESTAMPDIFF(year, fecna, CURRENT\_DATE)) AS ‘Más jóven’,

AVG(TIMESTAMPDIFF(year, fecna, CURRENT\_DATE)) AS Media

FROM temple;

* Sólo aplicable a la función COUNT

COUNT(\*)

Devuelve el número de filas que satisfacen la consulta

SELECT COUNT(\*)

FROM temple;

SELECT COUNT(\*)

FROM temple

WHERE numde=111;

SELECT COUNT(\*)

FROM temple

WHERE TIMESTAMPDIFF(year, fecna, CURRENT\_DATE) > 50;

\*\*\*\*\*\*\*\*\*\*\*\*\*ATENCIÓN\*\*\*\*\*\*\*\*\*\*\*\*\*\*

La proyección de una función colectiva es incompatible con la proyección de un dato individual, ya que la función colectiva devuelve un valor y el dato individual devuelve tantos valores como filas satisfagan la consulta.

## 2 - Agrupamiento explícito

Podemos crear grupos con aquellos valores de una consulta que se repitan.

Esto se realiza mediante la cláusula GROUP BY, cuya sintaxis es la siguiente.

SELECT proyección (\*1)

FROM tabla

[WHERE selección]

[GROUP BY at1 [at2, …, atn]]

[HAVING condición (2\*)]

[ORDER BY at1 [, at2, …, atn]]

(1\*) En la proyección implícita hemos visto que es incompatible la proyección de una función colectiva con una columna individual.

Cuando creamos grupos de manera explícita (GROUP BY) el concepto es similar, es decir, sólo se podrá proyectar el atributo por el que se crean los grupos junto con funciones de columna.

Ejemplo

| Obtener cuántos empleados trabajan en cada departamento |
| --- |
| SELECT numde, COUNT(\*)  FROM temple  GROUP BY numde; |

Podemos hacer agrupamientos basados en expresiones en vez de atributos.

| Obtener cuántos empleados hay con la misma edad |
| --- |
| SELECT TIMESTAMPDIFF(year, fecna, CURRENT\_DATE) AS Edad, COUNT(\*) AS Empleados  FROM temple  GROUP BY 1  ORDER BY 1; |

Podemos hacer agrupamientos por más de un atributo

| Obtener por departamento cuántos empleados tienen el mismo número de hijos |
| --- |
| SELECT numde, numhi, COUNT(\*) AS Empleados  FROM temple  GROUP BY 1, 2  ORDER BY 1; |

(2\*) Cláusula HAVING. Permite discriminar o descartar grupos previamente creados con GROUP BY.

Lleva una condición que se establece sobre el grupo, no sobre los elementos individuales. Por tanto, esa condición debe ser una función colectiva sobre el grupo.

| Obtener por departamento, cuántos empleados trabajan en cada departamento, para los departamentos en los que trabajen más de tres empleados |
| --- |
| SELECT numde, count(\*) AS Empleados  FROM temple  GROUP BY 1  HAVING (COUNT(\*) > 3)  ORDER BY 1; |

\*\*\*\*\*\*ATENCIÓN\*\*\*\*\*\*

No se debe confundir la condición sobre el grupo(HAVING) con la condición sobre la consulta(WHERE). Recuerda que el HAVING, al ser sobre el grupo, es una condición basada en una condición de columna, mientras que el WHERE es una condición que puede estar basada en cualquier atributo o expresión.

| Para los empleados que tengan más de 40 años, obtener cuántos empleados hay con la misma edad, siempre que haya más de un empleado con la misma edad. |
| --- |
| SELECT TIMESTAMPDIFF(year, fecna, CURRENT\_DATE) AS Edad, COUNT(\*)  FROM temple  WHERE TIMESTAMPDIFF(year, fecna, CURRENT\_DATE) > 40  GROUP BY 1  HAVING (COUNT(\*) > 1)  ORDER BY 1; |

| Obtener por departamento, la media del salario, cuantos salarios distintos hay en cada departamento, cuantas comisiones distintas hay y cual seria la media de estas comisiones ordenador por departamento |
| --- |
| SELECT numde, AVG(salar), COUNT(DISTINCT salar), COUNT(DISTINCT comis), SUM(comis)/COUNT(\*)  FROM temple  GROUP BY numde  ORDER BY numde; |

| Obtener por departamento, la media del salario, cuantos salarios distintos hay en cada departamento, cuantas comisiones distintas hay y cual seria la media de estas comisiones, de los empleados que trabajan en el departamento 111 |
| --- |
| SELECT AVG(salar), COUNT(DISTINCT salar), COUNT(DISTINCT comis), SUM(comis)/COUNT(\*)  FROM temple  WHERE numde=111; |

| Hallar el salario medio por departamento para aquellos departamentos cuyo salario máximo es inferior al salario medio de todos los empleados |
| --- |
| SELECT numde, AVG(salar) AS ‘Salario medio’  FROM temple  GROUP BY numde  HAVING (MAX(salar) < (SELECT AVG(salar) FROM temple))  ORDER BY numde; |

| Para los departamentos en los que la antigüedad media de los empleados supere la de la empresa, hallar el salario mínimo, máximo y la media |
| --- |
| SELECT numde, MIN(salar), MAX(salar), AVG(salar)  FROM temple  GROUP BY numde  HAVING AVG(TIMESTAMPDIFF(year, fecin, CURRENT\_DATE)) > (SELECT AVG(TIMESTAMPDIFF(year, fecin, CURRENT\_DATE)) FROM temple)  ORDER BY numde; |

| Para cada departamento, hallar la media de la comisión con respecto a los empleados que la reciben y con respecto al total de empleados de ese departamento |
| --- |
| SELECT numde, AVG(comis), SUM(comis)/COUNT(\*)  FROM temple  GROUP BY numde  ORDER BY numde; |

| Obtener cuántos empleados hay con la misma edad en cada departamento |
| --- |
| SELECT numde, TIMESTAMPDIFF(year, fecna, CURRENT\_DATE) AS EDAD,  COUNT(\*) AS Empleados  FROM temple  GROUP BY numde, TIMESTAMPDIFF(year, fecna, CURRENT\_DATE)  ORDER BY 1, 2; |

| Para los departamentos en los que trabajen más de 2 empleados, obtener por cada uno de ellos el número máximo, mínimo y la media de los hijos de esos empleados |
| --- |
| SELECT numde, MAX(numhi) AS ‘Número máximo de hijos’,  MIN(numhi) AS ‘Número mínimo de hijos’,  AVG(numhi) AS ‘Media de hijos’  FROM temple  GROUP BY numde  HAVING COUNT(\*) > 2; |

| Para los empleados que ingresaron en la empresa durante la primera década del 2000-2009, obtener por departamento, cuántos son y cuál es la media de su salario, para aquellos en que el departamento que su salario sea inferior a la media de todos los empleados de la empresa |
| --- |
| SELECT numde, count(\*), AVG(salar)  FROM temple  WHERE year(fecin) BETWEEN 2000 AND 2009  GROUP BY numde  HAVING salar < (SELECT AVG(salar) from temple); |

# CONSULTAS SOBRE VARIAS TABLAS

## 1 - Álgebra relacional

Codd enunció el álgebra relacional como la base matemática para operar con el Modelo Relacional.

Se trata de un conjunto de operadores que tratan a las relaciones(tablas) como conjuntos y a una fila de una tabla como un elemento de un conjunto.

Estos operadores pueden clasificarse:

### 1.1 - Atendiendo a su naturaleza

1. Operadores clásicos de conjuntos

* Unión
* Diferencia
* Intersección
* Producto Cartesiano

1. Operadores definidos para el Modelo Relacional

* Proyección
* Selección
* Reunión natural (JOIN)
* División

### 1.2 - Atendiendo el número de operandos

Es decir, al número de operaciones sobre el que operan

1. Unarios. Operan sobre una única relación

* Proyección
* Selección

1. Binarios. Necesitan al menos 2 relaciones para operar

* Unión
* Diferencia
* Intersección
* Producto Cartesiano
* Reunión
* División

## Operadores Algebráicos

La aplicación de algunos de estos operadores requiere una condición previa y es la COMPATIBILIDAD ENTRE RELACIONES

Diremos que dos relaciones son compatibles si tiene el mismo número de atributos o columnas, y además esas columnas están definidas en el mismo tipo de dato y en el mismo orden

### 1 - UNIÓN

Es un operador binario que requiere que las 2 relaciones sean compatibles.

Por ejemplo: Sean A y B relaciones compatibles

SELECT \* FROM A UNION SELECT \* FROM B;

Devuelve todas las filas de A y de B sin repetición

SELECT\* FROM A UNION ALL SELECT B;

Devuelve todas las filas de A y B, aunque haya repeticiones

### 2 - DIFERENCIA

Es un operador binario que requiere que las 2 relaciones sean compatibles.

Por ejemplo: Sean A y B relaciones compatibles

A MINUS B; Devuelve las filas de A que no están en B

B MINUS A; Devuelve las filas de B que no están en A (\*No es conmutativa\*)

### 3 -INTERSECCIÓN

Es un operador binario que requiere que las 2 relaciones sean compatibles.

Por ejemplo: Sean A y B relaciones compatibles

A INTERSECT B; Devuelve las filas comunes a ambas tablas

### 4 - PROYECCIÓN

Es un operador unario que requiere sólo una relación para operar

Obtener un subconjunto vertical de una relación

SELECT {\* | at1[,at2…atn] | expresión \*\*\*PROYECCIÓN\*\*\*

FROM TABLA;

### 5- SELECCIÓN

Es un operador unario que requiere sólo una relación para operar

Obtener un subconjunto horizontal de la tabla estableciendo una condición que deben cumplir las filas de la tabla

SELECT proyección

FROM tabla

WHERE (condición); \*\*\*SELECCIÓN\*\*\*

### 6 - PRODUCTO CARTESIANO

Es un operador binario pero que no requiere que las relaciones sean compatibles

Sean A y B dos relaciones no necesariamente compatibles, el resultado del producto cartesiano de A\*B es:

* Una tabla que tiene las columnas de A más las columnas de B
* Tiene tantas filas como resulte de combinar cada fila de A, con todas las de B

Ejemplo

tcentr\*tdepto

SELECT \*

FROM tcentr, tdepto;

### Cualificación de nombres

Cuando trabajamos con más de una relación, puede ocurrir que haya atributos que se llamen igual en más de una de ellas. Esto ocurre con casi todas las claves foráneas.

Para poder referirnos con exactitud a estos atributos se utiliza la cualificación de nombres, que consiste en indicar el nombre de la tabla antes del atributo.

SELECT \*

FROM tcentr, tdepto

ORDER BY tcentr.numce;

Además del nombre de la tabla, también usar un alias de la tabla que se crea en el FROM

Por ejemplo la consulta anterior es similar a:

SELECT \*

FROM tcentr c, tdepto d

ORDER BY c.numce;

### 7 - REUNIÓN

Es el operador más importante del álgebra relacional ya que nos permite acceder a las tuplas de una tabla que están relacionadas con las de otra.

Podemos decir que la reunión es un operador compuesto ya que resulta de la combinación de 2 de los operadores anteriores

reunión natural = producto cartesiano + la selección de los atributos comunes

Ejemplo

Reunión entre centros y departamentos

SELECT \*

FROM tcentre c, tdepto d

WHERE c.numce=d.numce

ORDER BY c.numce;

Reunión entre departamentos y empleados

SELECT \*

FROM tdepto d, temple e

WHERE d.numde=e.numde

ORDER BY d.numde;

La reunión puede hacerse algebraicamente, como acabamos de ver o también puede realizarse mediante el operador específico JOIN

SELECT \*

FROM tdepto d JOIN temple e ON d.numde=e.numde

ORDER BY d.numde;

La reunión es el operador “mágico” que nos permite obtener información de una tabla a partir de datos de cualquier tabla de mi esquema.

Ejemplo

Obtener todos los datos de los empleados que trabajan en departamentos cuyo presupuesto es superior a 50.000 euros

SELECT e.\*

FROM tdepto d, temple e

WHERE d.numde=e.numde AND presu>50000;

Obtener todos los datos de los departamentos en los que trabajen empleados con más de tres hijos

SELECT d.\*

FROM tdept d, temple e

WHERE d.numde=e.numde AND numhi>3

SELECT c.numce, nomce, d.numde, nomde, numem, nomem

FROM tcentre c, tdepto d, temple e

WHERE c.numde=d.numce AND d.dumce=e.numce

ORDER BY

# MÉTODO PARA LA REALIZACIÓN DE CONSULTAS DE TODO TIPO

Se trata de seguir unos pocos pasos en un orden determinado

1 - A partir de los datos que me piden (datos de salida) y de los datos que me dan (datos de entrada), determinar en el Modelo Lógico, las relaciones implicadas y el camino más corto para llegar de unos datos a otros. Realizar la Reunión Natural entre esas tablas (FROM y WHERE)

SELECT

FROM producto p, contiene c, pedido pe, usuario u

WHERE p.ia=c.id\_producto AND c.id\_pedido = pe.id AND pe.usuario = u.id

ORDER BY

2 - Añadir en la selección (WHERE) la condición o condiciones específicas de la consulta

SELECT

FROM producto p, contiene c, pedido pe, usuario u

WHERE p.ia=c.id\_producto AND c.id\_pedido=pe.id AND pe.usuario=u.id AND u.nombre LIKE ‘Alfonso’ AND apellidos LIKE ‘Navajas’

ORDER BY

3 - ¿Hay agrupamientos?

3.1 - Sí (porque me piden como salida una función de columna)

1. ¿Es explícito?

* Sí (lleva GROUP BY) Cuántos son por depto o por expresión
* ¿Hay descartes de grupo?
* Sí (lleva HAVING)
* No (Pasa al punto 4)
* No (Pasa al punto 4)

3.2 - No (paso al punto 4)

4 - ¿Qué me piden? Son los datos de Salida en el SELECT

5 - ¿Se pueden repetir las filas?

* Sí (lleva DISTINCT)

6 - ORDER BY

Obtén el nombre del departamento, número del departamento y el presupuesto de los departamentos de los empleados que tienen entre 30 y 39 años

SELECT DISTINCT d.numde, nomde, presu

FROM tdepto d, temple e

WHERE d.numde=e.numde

AND TIMESTAMPDIFF(year, fecna, CURRENT\_DATE) BETWEEN 30 AND 39

Para los empleados que trabajan en el centro sede central y que tienen hijos, obtener cuánto s empleados tienen el mismo número de hijos solo si hay más de dos empleados que tengan el mismo número de hijos

SELECT numhi, COUNT(\*)

FROM tcentr c, tdepto d, temple e

WHERE c.numce=d.numce AND d.numde=e.numde

AND nomce LIKE ‘SEDE CENTRAL’ AND numhi>0

GROUP BY numhi

HAVING COUNT(\*)>2;

Ejemplo

Obtener cuantos países europeos pertenecen a la ONU

SELECT COUNT(\*) AS ‘Paises europeos en la ONU’

FROM pais p, pertenece\_a pe, organizacion o

WHERE p.cod\_pais=pe.cod\_pais AND pe.cod\_organizacion=o.cod\_organizacion

AND continente LIKE ‘Europa’ AND siglas LIKE ‘ONU’

¿Cuántos países por continente pertenecen a la ONU?

SELECT continente, COUNT(\*) AS ‘Paises en la ONU’

FROM pais p, pertenece\_a pe, organizacion o

WHERE p.cod\_pais=pe.cod\_pais AND pe.cod\_organizacion=o.cod\_organizacion

AND siglas LIKE ‘ONU’

GROUP BY continente;

¿Cuántos países por continente, pertenecen a la ONU, pero sólo de los continentes en los que haya al menos 10 países que pertenezcan a la ONU?

SELECT continente, COUNT(\*) AS ‘Paises en la ONU’

FROM pais p, pertenece\_a pe, organizacion o

WHERE p.cod\_pais=pe.cod\_pais AND pe.cod\_organizacion=o.cod\_organizacion

AND siglas LIKE ‘ONU’

GROUP BY continente

HAVING COUNT(\*)>9;

El nombre de los idiomas que se hablan en España

SELECT i.nombre

FROM pais p, se\_habla se, idioma i

WHERE p.cod\_pais=se.cod\_pais AND se.cod\_idioma=i.cod\_idioma

AND p.nombre LIKE ‘España’;

Nombre de los idiomas que se hablan en países de Europa

SELECT DISTINCT i.nombre

FROM pais p, se\_habla se, idioma i

WHERE p.cod\_pais=se.cod\_pais AND se.cod\_idioma=i.cod\_idioma

AND continente LIKE ‘Europa’;

Cuantos idiomas se hablan en cada uno de los países de Europa

SELECT p.nombre, COUNT(\*) AS Idiomas

FROM pais p, se\_habla se, idioma i

WHERE p.cod\_pais=se.cod\_pais AND se.cod\_idioma=i.cod\_idioma

AND continente LIKE ‘Europa’

GROUP BY p.nombre;

\*\*\*\*\*consultas del examen del curso pasado sobre camisetas\*\*\*\*\*

Bloque 3

1.-Obtener, por nombre de categoría, cuántos productos hay en total de cada categoría, el más caro y el más barato de cada una de ellas

SELECT COUNT(\*) AS Total, MAX(precio) AS ‘Más caro’, MIN(precio) AS ‘Más barato’

FROM categoria c, producto p

WHERE c.id=p.categoria

GROUP BY c.nombre;

2.-Obtener todos los datos de los productos de los que se han realizado pedidos durante el año 2020 y que contuviera más de 10 unidades de algún producto

SELECT DISTINCT p.\*

FROM producto p, contiene c, pedido pe

WHERE p.id=c.id\_producto AND c.id\_pedido=pe.id

AND year(pe.fecha)=2020 AND unidades>10;

3.-Obtener nombre de la categoría y nombre del producto, de aquellos productos de los que existe un stock en el almacén superior a la media del stock que existe de todos los productos

SELECT c.nombre, p.nombre

FROM categoria c, producto p

WHERE c.id=p.categoria AND stock > (SELECT AVG(stock) FROM producto);

4.-Obtener todos los datos de los productos de los que ha realizado algún pedido el usuario de nombre ‘Admin’ durante el año 2020

SELECT DISTINCT p.\*

FROM producto p, contiene c, pedido pe, usuario u

WHERE p.id=c.id\_producto and c.id\_pedido=pe.id AND pe.usuario=u.id

AND u.nombre LIKE ‘Admin’ AND year(pe.fecha)=2020;

5.-Para los pedidos realizados en los últimos 6 meses, obtener, por nombre de usuario, cuántos pedidos ha realizado cada usuario y el coste medio de esos pedidos

SELECT nombre, COUNT(\*) AS Pedidos, AVG(coste) AS ‘Coste Medio’

FROM pedido pe, usuario u

WHERE u.id=p.usuario

AND TIMESTAMPDIFF(month, fecha, CURRENT\_DATE) <= 6

GROUP BY nombre;

6.-Obtener, por id de categoría de producto, cuántos pedidos se han realizado durante el 2020 de cada categoría, para aquellas categorías de las que se hayan realizado más de 3 pedidos

SELECT categoria, COUNT(\*) AS ‘Pedidos por categoria’

FROM producto p, contiene c, pedido pe

WHERE p.id=c.id\_producto AND c.id\_pedido=pe.id

AND year(pe.fecha)=2020

GROUP BY categoria

HAVING COUNT(\*) > 3;

Obtener todos los datos de los países que tengan ciudades con un número de habitantes entre medio millón y un millón de personas

SELECT DISTINCT p.\*

FROM pais p, ciudad c

WHERE p.cod\_pais=c.cod\_pais AND c.habitantes BETWEEN 500000 AND 1000000;

Obtener todos los datos de los clientes que tuvieron que reparar un vehículo suyo durante el año 2021

SELECT c.\*

FROM cliente c, vehiculo v, reparacion r

WHERE c.dni=v.cliente AND v.matricula=r.matricula

AND year(v.fecha) = 2021;

Cuántas piezas se han necesitado vehículos de la marca Renault en las reparaciones de la base de datos hasta ahora

SELECT n.pieza, COUNT(\*)

FROM vehiculo v, reparacion r, necesita n

WHERE v.matricula=r.matricula AND r.ref=n.reparacion

AND v.marca LIKE Reanult;

Para los empleados que trabajan en sector industrial, obtener nombre del empleado, salario bruto(salario de la tabla temple), cuota de seguridad social(16% salario bruto) y el salario neto(el resto)

SELECT nomem AS Empleado, salar AS ‘Salario bruto’, salar\*0.16 AS ‘Cuota SS’, salar\*0.84 AS ‘Salario Neto’

FROM tdepto d, temple e

WHERE d.numde=e.numde

AND nomde LIKE ‘SECTOR INDUSTRIAL’;

Para los países en los que se hablen en ingles obtener su nombre y capital siempre que su extensión sea superior a la extensión media de los países en los que se habla en francés

SELECT

FROM pais p, se\_habla se, idioma i

WHERE p.cod\_pais=se.cod\_pais AND se.cod\_pais=i.cod\_idioma

AND i.nombre LIKE ‘Ingles’

AND extension > (SELECT avg(extension)

FROM pais p, se\_habla se, idioma i

WHERE p.cod\_pais=se.cod\_pais AND se.cod\_pais=i.cod\_idioma

AND i.nombre LIKE ‘Frances’);

Dime los nombres de los idiomas que se hablan en países cuyo número de habitantes es superior al número de habitantes de España, Francia y Portugal

SELECT DISTINCT i.nombre

FROM pais p, se\_habla se, idioma i

WHERE p.cod\_pais=se.cod\_pais AND se.cod\_pais=i.cod\_idioma

AND num\_hab > ALL (SELECT num\_hab FROM pais WHERE nombre IN (‘España’, ‘Francia’, ‘Portugal’));

Por nombre de país, cuántos idiomas se hablan en cada país para aquellos países en los que se hablan más de un idioma

SELECT p.nombre AS Idiomas

FROM pais p, se\_habla se, idioma i

WHERE p.cod\_pais=se.cod\_pais AND se.cod\_pais=i.cod\_idioma

GROUP BY p.nombre

HAVING COUNT(\*)>1;

Obtener por continente y por país, cuantos paises limitan con cada uno de los países ordenador por continente y país

SELECT continente, nombre, COUNT(\*) AS ‘Limita con’

FROM pais p, limita\_con l

WHERE p.cod\_pais=l.cod\_pais1

GROUP BY continente, nombre

ORDER BY continente, nombre;

17 -

select titulo, precio\_alquiler

from pelicula p, categoria c

where p.categoria = c.categoria

and precio\_alquiler > (select avg(precio\_alquiler) from pelicula p, categoria c where p.categoria=c.categoria and genero like ‘Suspense’);

18 -

select genero, count(\*) as Total

from alquiler a, pelicula p

where a.titulo = p.titulo

group by genero

having count(\*)>2;

19 -

select max(date\_format(fecha\_alquiler, ‘%d/%m/%Y’)) as ‘Último alquiler’

from alquiler

where titulo like ‘LA NOCHE DEL CAZADOR’;

20 -

select timestampdiff(year, fecna, current\_date) as Edad, count(\*) as Total

from cliente c, alquiler a

where c.dni=a.cliente

group by 1

order by 1;

21 -

alter table cliente add mayor enum(‘S’, ‘N’);

22 -

update cliente set mayor=’N’;

update cliente set mayor=’S’ where timestampdiff(year, fecna, current\_date)>17;