



El Gestor de Bases de Datos MySQL

Lenguajes Interpretados en el Servidor

Objetivos

1. Introducir al estudiante a la utilización de bases de datos MySQL.
2. Desarrollar la habilidad de crear una base de datos en MySQL desde consola y desde algún cliente web o de escritorio.
3. Dominar los aspectos necesarios para la creación de tablas, definición de campos y establecimiento de relaciones entre tablas.
4. Aplicar la sintaxis de las instrucciones SQL fundamentales para creación de tablas, manejo de registros y establecimiento de permisos.

Contenido

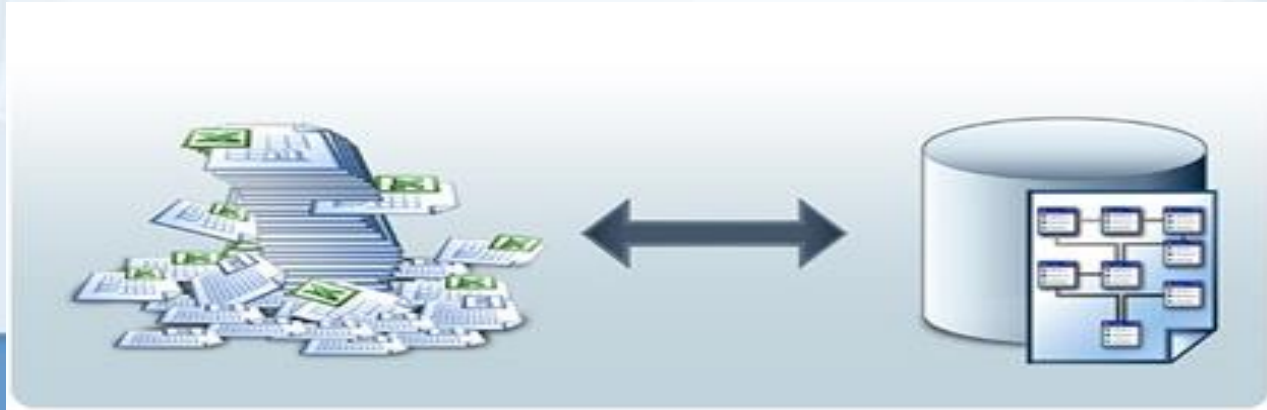
1. Bases de datos vs archivos.
2. Bases de datos relacionales.
3. Conceptos de bases de datos.
 1. Tablas.
 2. Filas y columnas.
 3. Valores.
 4. Claves o índices.
 5. Relaciones.
4. El lenguaje SQL.
5. Definición de datos.
6. Manipulación de datos.
7. Las sentencias SQL.
8. MySQL.

Contenido

1. Utilizar el cliente MySQL.
2. Iniciar sesión en MySQL.
3. Conectarse con el servidor MySQL.
4. Motores de almacenamiento.
 1. MyISAM.
 2. InnoDB.
5. Comparativa entre MyISAM e InnoDB.
6. phpMyAdmin.

Bases de datos vs Archivos

1. Para almacenar información en una aplicación web, es recomendable el uso de bases de datos en lugar de archivos.
2. Las razones son diversas, pero las principales son tres.

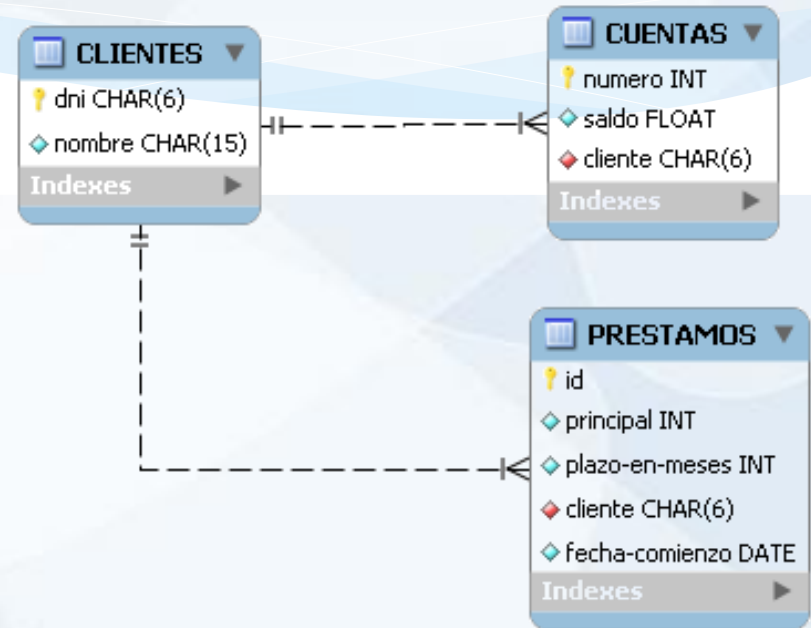


Bases de datos vs Archivos

1. Entre más crece la cantidad de información almacenada en un archivo la gestión de datos se vuelve cada vez más lenta, puesto que el acceso a los archivos es, en la mayoría de los casos, secuencial.
2. Los accesos concurrentes de varios usuarios a un archivo son extremadamente ineficientes e imposibilitan limitar accesos de forma selectiva.
3. Se propicia la aparición de inconsistencias por la imposibilidad de restringir accesos simultáneos a los archivos.

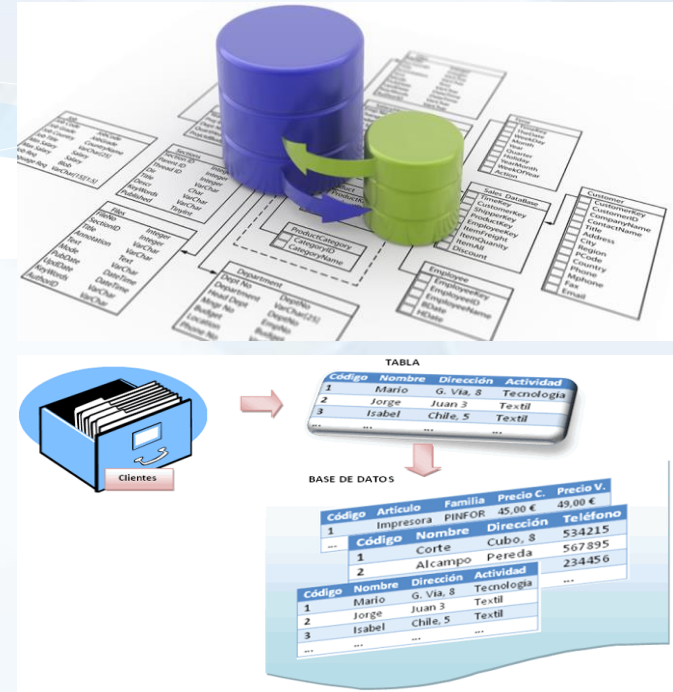
Bases de datos relacionales

1. Las bases de datos relacionales son el tipo de base de datos más utilizado en aplicaciones informáticas y particularmente en aplicaciones web.
2. Estas bases de datos están basadas en el álgebra relacional que incluye conceptos como tablas, filas, columnas, claves, relaciones, etc.



Tablas

1. Las bases de datos relacionales están compuestas por relaciones, conocidas regularmente como tablas.
2. Una tabla es una estructura compuesta por datos distribuidos en filas y columnas, como una hoja de excel.

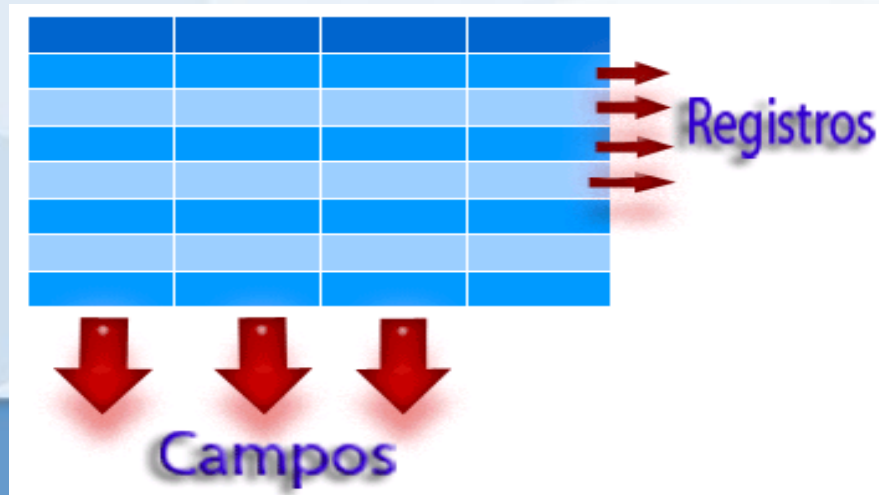


Filas y columnas

1. Una **fila de una tabla** representa información relacionada con una misma ocurrencia de una entidad, llámese entidad al elemento sobre el que se desea almacenar información, como una persona, un cliente, un libro, una cuenta bancaria o un artículo de una tienda.
2. En términos prácticos, si se trata de una tabla de clientes, cada fila representa información de un único cliente. Del mismo modo, si se tratase de una tabla con estudiantes de la universidad, cada fila representaría información relacionada con un único estudiante.
3. Las filas son llamadas también registros o tuplas de la tabla.

Filas y columnas

1. Una **columna de una tabla** representa un dato único e indivisible que está asociado con un único tipo de dato.
2. Las columnas también reciben el nombre de **campos** o **atributos** de la relación o tabla.



A screenshot of a database application showing two tables: "Proveedores" (Suppliers) and "Productos" (Products). The "Proveedores" table has two columns: "Id. de proveedor" and "Nombre de la compañía". The "Productos" table has three columns: "Id. de producto", "Nombre de producto", and "Proveedor". A red arrow points from the "Proveedor" column in the "Productos" table to the "Proveedores" table, indicating a foreign key relationship.

Id. de proveedor	Nombre de la compañía
1	Exotic Liquids

Id. de producto	Nombre de producto	Proveedor
1	Chai	1
2	Chang	1
3	Sirope de anís	1

Valores

1. Cada **fila** o **registro** de una tabla se compone de un conjunto de valores individuales que se corresponden con las columnas o campos definidos para esa tabla.
2. Cada uno de estos valores debe ser del tipo de dato especificado en la definición de la columna para esa tabla.

Campo	id	nombre	email
Tipo ②	INT	VARCHAR	VARCHAR
Longitud/Valores*1		100	100
Predeterminado ²	None	None	None
Cotejamiento			
Atributos			
Nulo	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Índice	PRIMARY		
AUTO_INCREMENT	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Comentarios			

Campos de texto (VARCHAR)
longitud 100

Campo auto incremental (AUTO_INCREMENT)

Campo clave (PRIMARY)

Campo numérico (INT)

Valores

localhost / localhost / for_example / ...

phpMyAdmin

Database: for_example

for_example (0)

No tables found in database.

Server: localhost Database: for_example Table: example1

Field	Type ?	Length/Values ¹	Default ²	Collation
id	INT		None	
title	VARCHAR	255	None	
content	TEXT		None	
is_public	TINYINT	4	None	
created_at	INT		None	

Table comments:

PARTITION definition: ?

Collation:

Save Or Add 1 field(s) Go

¹ If field type is "enum" or "set", please use values separated by commas.
² If you ever need to put a backslash in a value, precede it with a backslash (for example '\xyz' or 'a\b').

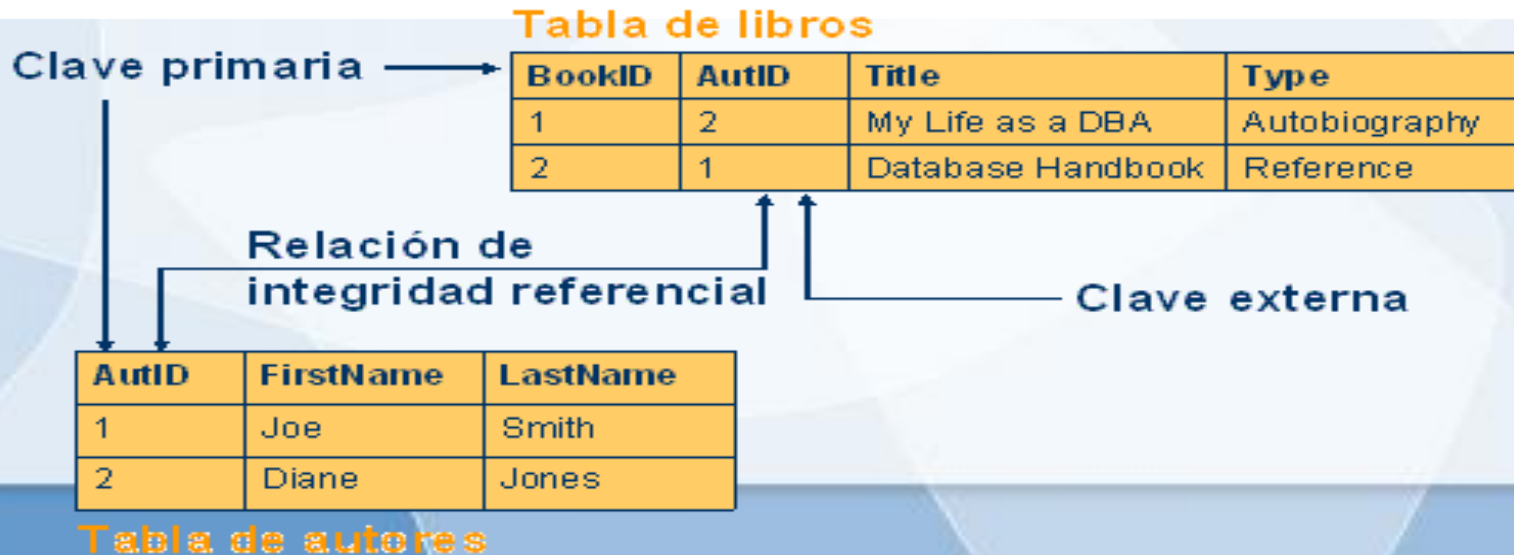
- FLOAT
- DOUBLE
- REAL
-
- BIT
- BOOL
- SERIAL
- DATE and TIME**
- DATE
- DATETIME
- TIMESTAMP**
- TIME
- YEAR
- STRING**
- CHAR
- VARCHAR
-
- TINYTEXT
- TEXT
- MEDIUMTEXT

Claves o índices

1. Las claves o índices son un tipo de campo que se define con el propósito de identificar cada registro de una tabla de forma unívoca.
2. No es conveniente utilizar campos de cadenas que almacenen nombres o apellidos como clave primaria de una tabla, ya que estas pueden tener repeticiones que impidan identificar unívocamente a cada registro.
3. Lo más práctico es utilizar un número de identificación para cada registro. Esta técnica garantiza la exclusividad del registro, diferenciándolo fácilmente del resto.

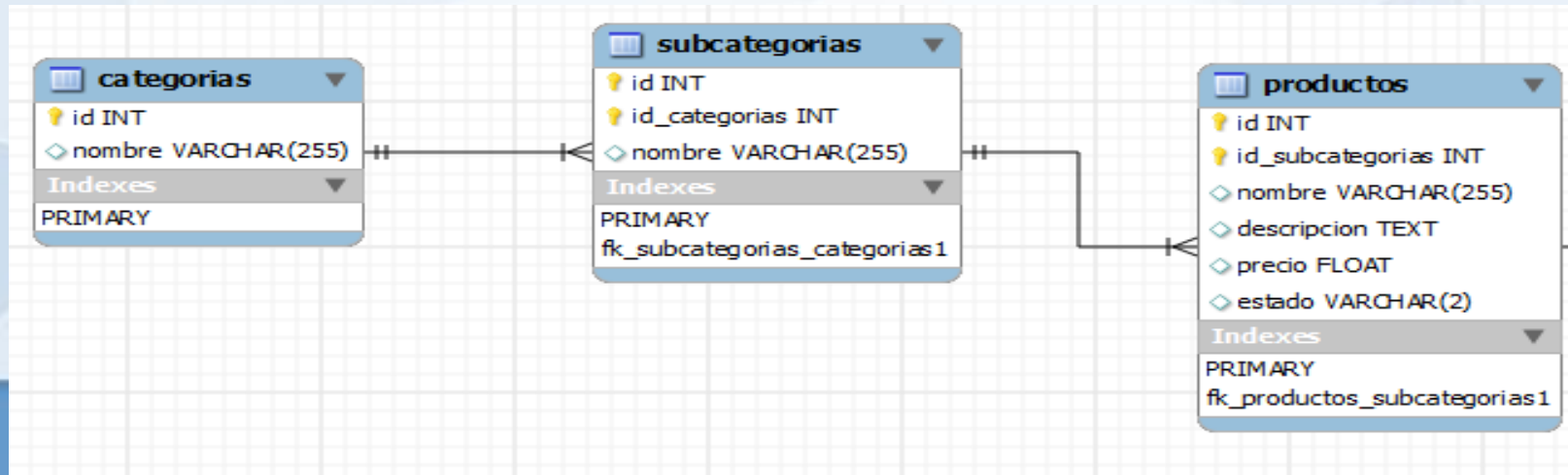
Claves o índices

1. Las bases de datos relacionales suelen constar de varias tablas que utilizan una **clave** o **llave primaria** para relacionar una tabla con otra en la que dicha clave aparece como **llave foránea**.



Relaciones

1. Las relaciones permiten definir una asociación entre dos tablas de la base de datos, relacionando el campo clave primaria de una de ellas con el campo clave secundaria o foránea de la otra.

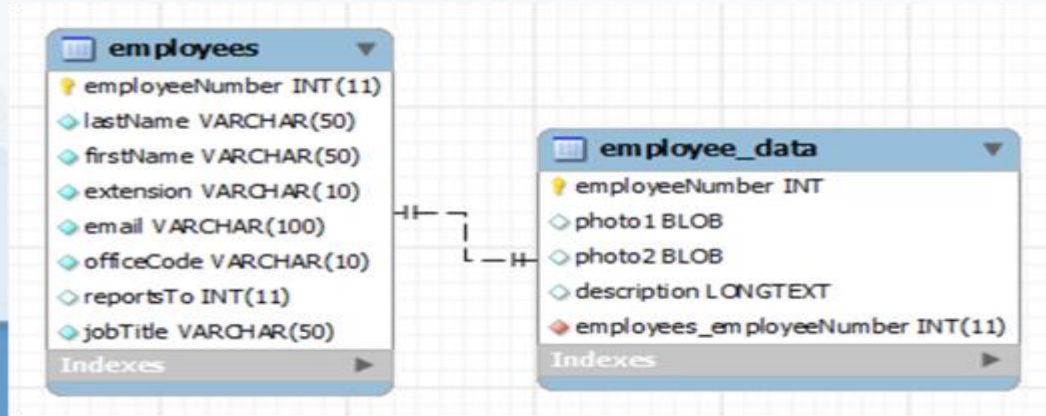


Relaciones

1. Existen tres tipos básicos de relaciones en una base de datos relacional.
2. Estos tipos de relaciones se clasifican en función del número de elementos que haya a cada lado de la relación.
3. Es así que se pueden tener relaciones de:
 1. Uno a uno.
 2. Uno a muchos (uno a varios).
 3. Muchos a muchos (varios a varios).

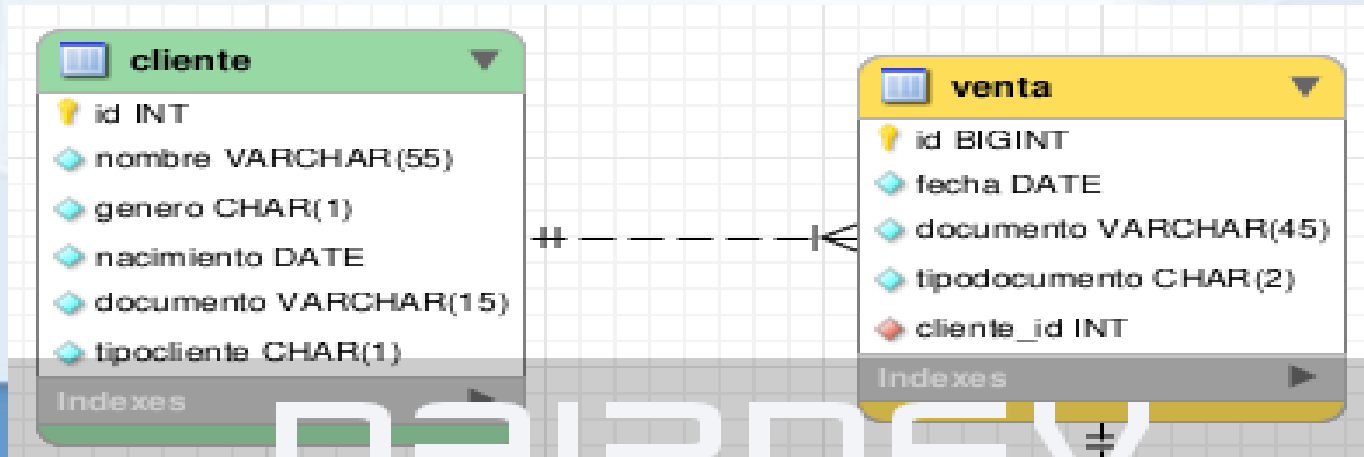
Relación uno a uno

1. Una relación **uno a uno** significa operativamente que sólo existe un elemento a cada lado de la relación (en cada tabla).
2. Técnicamente, se dice que en una **relación uno a uno** para **cada instancia** de la **primera tabla** sólo existe **una instancia coincidente** en la **segunda tabla**.



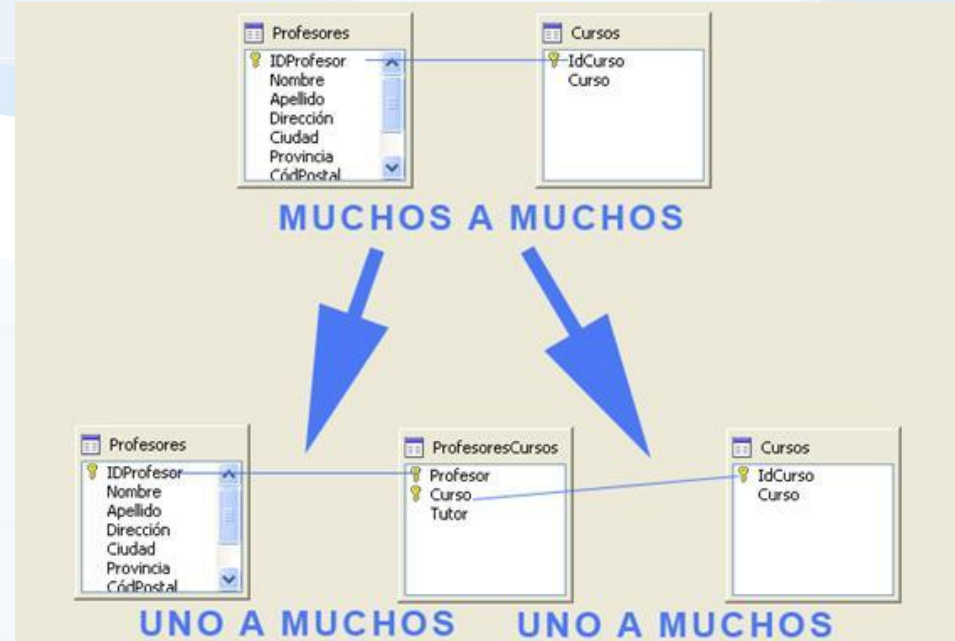
Relación uno a muchos

1. Una **relación uno a muchos** es una relación en la que para cada instancia de la **primera tabla**, existen **varias o muchas instancias** en la **segunda**.
2. Este es el tipo de relación más habitual dentro de las bases de datos.



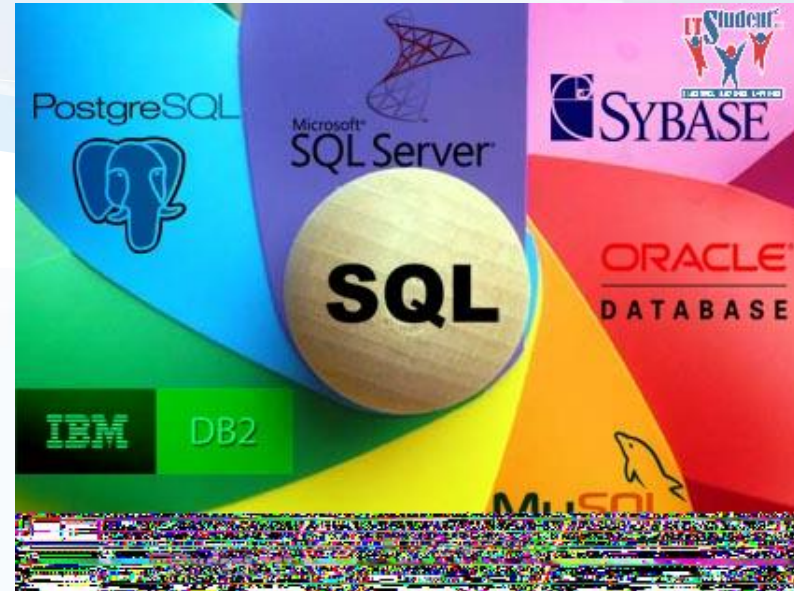
Relaciones muchos a muchos

1. Una **relación muchos a muchos** es una relación que tiene lugar cuando **para cada instancia de la primera tabla existen varias instancias en la segunda** y viceversa.
2. De acuerdo al **modelo relacional** una **relación de muchos a muchos** debe ser resuelta con **dos relaciones de uno a muchos**.



El lenguaje SQL

1. SQL es el Lenguaje Estructurado de Consultas (Structured Query Language).
2. Es un lenguaje estándar para acceder a los sistemas de administración de bases de datos (RDBMS).
3. El lenguaje SQL se utiliza para almacenar y consultar datos desde y hacia una base de datos.



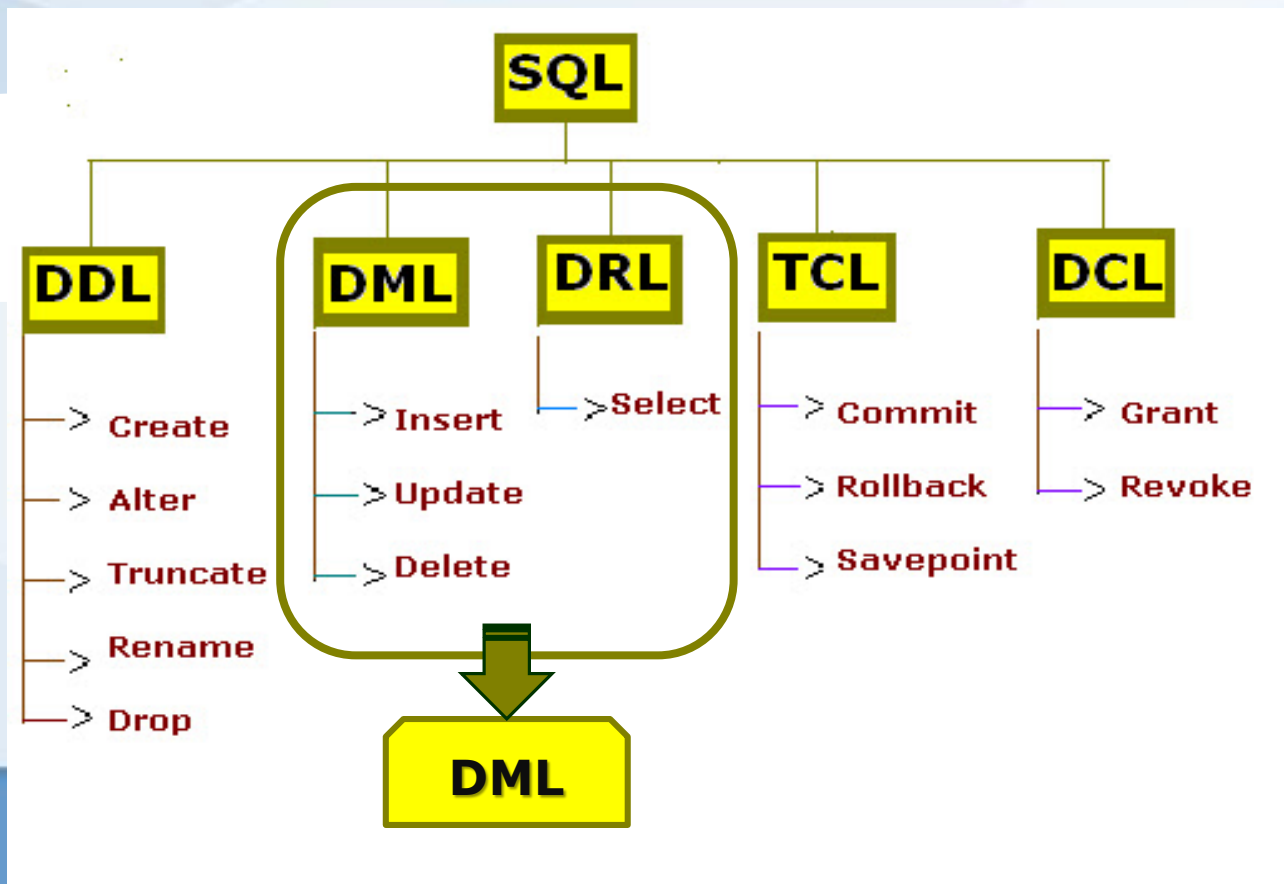
El lenguaje SQL

1. El lenguaje SQL consta de unas 30 sentencias que permiten la creación de bases de datos y tablas, así como de la recuperación, inserción, modificación y eliminación de registros de las tablas.
2. También permite la asignación de permisos y privilegios para los usuarios de la base de datos.
3. Todas estas sentencias están agrupadas en tres grupos:
 - 👉 Sentencias del Lenguaje de Definición de Datos (DDL),
 - 👉 Sentencias del Lenguaje de Manipulación de Datos (DML) y
 - 👉 Sentencias del Lenguaje de Control de Datos (DCL).

El lenguaje SQL

1. Aunque en la actualidad, algunos proveedores de Sistemas de Gestión de Bases de Datos utilizan hasta cinco subdivisiones del lenguaje, la gran mayoría sólo destaca cuatro.
2. Los que dividen en cinco proponen un Lenguaje de Definición de Datos (DDL), un Lenguaje de Manipulación de Datos (DML), Lenguaje de Recuperación de Datos (DRL), Lenguaje de Control de Transacciones (TCL) y un Lenguaje de Control de Datos (DCL).
3. No obstante lo anterior, la mayoría de Sistemas de Gestión de Bases de Datos se manejan solamente cuatro, dejando el Lenguaje de Recuperación de Datos (DRL) que incluye básicamente la sentencia SELECT como parte del mismo Lenguaje de Manipulación de Datos (DML).

El lenguaje SQL



Definición de Datos

1. Dentro del **lenguaje SQL**, utilizado por el gestor MySQL existe una subdivisión conocida como **Lenguaje de Definición de Datos (DDL)**, que son **sentencias que permiten crear bases de datos y tablas, definir llaves primarias e índices o modificar la estructura de tablas, llaves e índices**:
2. Las principales sentencias del sublenguaje DDL son:
 - 👉 **CREATE**: crea bases de datos, tablas, llaves primarias, índices y vistas.
 - 👉 **DROP**: elimina bases de datos, tablas, llaves primarias, índices o vistas.
 - 👉 **ALTER**: modifica la estructura de tablas.

Creación de una base de datos

1. Para crear una base de datos en MySQL se usa el comando **CREATE DATABASE**. De la siguiente forma:

```
CREATE DATABASE nombre_bd  
[create_specification[,create_specification]...];
```

1. El comando anterior crea la base de datos indicada por nombre_bd.
2. Si la base de datos se ha creado con éxito el servidor responde de la siguiente forma:

```
mysql> CREATE DATABASE libreria;  
Query OK, 1 row affected (0.03 sec)
```

Creación de una base de datos

1. Las opciones `create specification` pueden utilizarse, de forma opcional, para especificar características adicionales de la base de datos.
2. Estas características se almacenan en el archivo `db.opt` en el mismo directorio (carpeta) de la base de datos (carpeta de base de datos en `data`).
3. La cláusula `CHARACTER SET` especifica el conjunto de caracteres por defecto de la base de datos.
4. La cláusula `COLLATE` especifica la "colación" o cotejamiento de caracteres que definen las reglas utilizadas para comparar caracteres en un conjunto de caracteres.

Consultando las bases de datos

1. Para poder consultar las bases de datos existentes o para verificar si la base de datos recién creada está presente en las bases de datos del servidor MySQL se ejecuta el siguiente comando:

SHOW DATABASES;

2. Este comando mostraría las bases de datos gestionadas por el servidor:

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| dltop     |
| libreria  |
| mysql     |
| phpmyadmin |
| test      |
+-----+
6 rows in set (0.00 sec)
```

Seleccionar la base de datos

1. En vista de que el servidor MySQL almacena muchas bases de datos, es preciso indicar **con qué base de datos se va a trabajar**. Para ello se utiliza el comando `USE basedatos;` de la siguiente forma:

USE libreria;

2. El comando anterior hace que se seleccione y se abra una conexión con la base de datos libreria.

Creación de una tabla

1. Para crear tablas se utiliza el comando `CREATE TABLE nombretabla;`
2. Tal y como se muestra en el siguiente ejemplo:

```
CREATE TABLE clientes(  
    idcliente int UNSIGNED NOT NULL AUTO_INCREMENT  
    PRIMARY KEY,  
    nombre char(30) NOT NULL,  
    apellido char(30) NOT NULL,  
    direccion char(50) NOT NULL,  
    ciudad char(30) NOT NULL  
);
```


Creación de una tabla

```
mysql> CREATE TABLE autor(  
-> idautor int unsigned not null auto_increment primary key,  
-> nombreautor char(30) not null,  
-> apellidautor char(30) not null,  
-> nacionalidad char(25) not null  
-> );_
```



Si la tabla fue creada con éxito el servidor lo notificará con un mensaje como el siguiente:

```
Query OK, 0 rows affected (0.19 sec)
```



El mensaje indica que la consulta se ejecutó con éxito, sin filas afectadas y en 0.19 segundos.

Consultar la estructura de una tabla

1. Para examinar la estructura de la tabla recién creada se puede utilizar el comando `DESCRIBE nombretabla;` de la siguiente forma:

DESCRIBE autor;

2. El resultado se muestra a continuación:

```
mysql> DESCRIBE autor;
```

Field	Type	Null	Key	Default	Extra
idautor	int(10) unsigned	NO	PRI	NULL	auto_increment
nombreautor	char(30)	NO			
apellidautor	char(30)	NO			
nacionalidad	char(25)	NO			

```
4 rows in set (0.15 sec)
```

Eliminar una tabla

1. Para eliminar tablas de una base de datos seleccionada con USE puede utilizar el comando:

```
DROP TABLE nombre_tabla;
```

1. Al ejecutar DROP TABLE se eliminan todos los registros de la tabla y la estructura completa de la misma.
2. Cuando lo que desee sea únicamente vaciar el contenido de la tabla conservando su estructura entonces utilice el comando

```
TRUNCATE TABLE nombre_tabla;
```

Manipulación de datos

1. Los gestores de bases de datos proporcionan una serie de **sentencias SQL** para **manipular registros** existentes en una base de datos o para crear nuevos registros.
2. Las **sentencias SQL**, relacionadas con la **manipulación de registros** son:
 - 📌 INSERT: Permite adicionar registros nuevos en las tablas.
 - 📌 SELECT: Permite obtener registros existentes de las tablas.
 - 📌 UPDATE: Permite actualizar/modificar los campos de los registros de las tablas.
 - 📌 DELETE: Permite eliminar registros de las tablas.

Insertar registros en las tablas

1. Para poder insertar nuevos registros en las tablas, debe utilizarse una sentencia **INSERT INTO** que debe indicar **la tabla de la base de datos** en la que se insertará el registro y **los nombres de los campos y los valores** que se colocarán en estos.
2. Existen varias sintaxis admitidas para una sentencia INSERT en MySQL, usted deberá decidir cuál es la más conveniente a la hora de realizar una inserción de registro.

Insertar registros en las tablas

1. Las distintas formas de insertar registros son las siguientes:

```
INSERT INTO tabla (campo1, campo2, ... , campoN)  
VALUES (val1, val2, ... , valN);
```

```
INSERT INTO tabla VALUES (val1, val2, ... , valN);
```

```
INSERT INTO tabla SET campo1=val1, campo2=val2, ...  
    , campoN=valN;
```

--INSERT Extendido

```
INSERT INTO tabla (campo1, campo2, ... , campoN)  
VALUES (val11, val12, ... , val1N),  
    (val21, val22, ... , val2N), ...  
    (valN1, valN2, ... , valNN);
```

Consultar registros en las tablas

1. La sentencia SELECT se utiliza para recibir registros seleccionados desde una o más tablas.
2. De hecho, MySQL incluye soporte para subconsultas. Esto es, una consulta dentro de otra consulta; es decir, sentencias SELECT anidadas.

```
SELECT lista_campos FROM nombre_tabla1 [WHERE  
condicion];
```

The diagram illustrates the parts of a SQL query using a sample query: `Select Nombre, Apellido from Clientes where Cliente_ID = 5 and Ciudad = "Buenos Aires"`.

- Select:** A red box explains that `lista_campos` are identifiers with no special meaning for SQL, so they can't be used for reserved words like `where`. It points to `Select` in the query.
- from:** A blue box explains that `nombre_tabla` can be defined as combinations of characters used for selections and comparisons, pointing to `from Clientes`.
- where:** A green box explains that `nombre_de_objeto` are values that refer to table names, column names, or other database objects, pointing to `where Cliente_ID = 5 and Ciudad = "Buenos Aires"`. Below this box, a list shows: `Nombre ; Apellido ; Clientes ;` and `Cliente_ID ; Ciudad`.
- and:** An orange box explains that `valores` are values that correspond directly to the instruction, pointing to the values `5` and `"Buenos Aires"`.

Consultar registros en las tablas

1. Existen comodines para abreviar la escritura de nombres de campo. Por ejemplo, un asterisco (*) en lista_campos indicaría que se desean obtener todas las columnas en la consulta.

```
SELECT *  
FROM nombre_tabla;
```



```
SELECT * FROM Documents, Categories  
WHERE Documents.cat = Categories.id
```

Actualizar datos de una tabla

1. Para actualizar o modificar la información almacenada en los registros de una tabla debe utilizarse una sentencia UPDATE.
2. Esta sentencia se utiliza en conjunto con una cláusula SET para establecer los nuevos valores de los campos que se volverán a definir.

```
UPDATE nombre_tabla1  
SET nombre_campo1 = expresion1  
[, nombre_campo2 = expresion2, ...]  
[WHERE condicion];
```

Actualizar datos de una tabla

1. La sintaxis de una sentencia UPDATE es la siguiente:

```
UPDATE tabla SET campo1='valor1', campo2='valor2', ...,  
campoN='valorN';
```

1. La cláusula SET permite indicar qué campos serán modificados o actualizados.
2. Además, se puede utilizar una cláusula WHERE que especifica los registros que deberán ser actualizados. Si no está presente se actualizan todos los registros

```
UPDATE noticia  
SET titulonoticia='La afición del FC Barcelona nunca falla'  
WHERE idnoticia=3;
```

Eliminar registros de las tablas

1. La sentencia MySQL para eliminar registros de una tabla es la sentencia DELETE.
2. La sintaxis de una sentencia DELETE es la siguiente:

```
DELETE FROM tabla WHERE expresion;
```

1. La cláusula FROM se antepone al nombre de la tabla de la que se van a borrar registros, en tanto que WHERE se utiliza para filtrar las filas que deben ser borradas. Si no se utiliza WHERE se borrarán todos los registros de la tabla.

Transacción de datos

1. Cuando se trabaja con el motor InnoDB se puede hacer uso de las transacciones en MySQL.
2. Una transacción es, en esencia, un grupo de instrucciones que son ejecutadas como una sola, de modo que si alguna de las instrucciones del grupo falla, deben deshacerse todas las del resto del grupo que fueron ejecutadas, para retornar la base de datos a un estado consistente (conforme a ACID: Atomicidad, Consistencia, Aislamiento y Durabilidad).
3. Lo anterior significa que InnoDB da todo el soporte para transacciones con sentencias COMMIT (confirmación) y ROLLBACK (cancelación) y recuperación contra fallos.

Transacción de datos



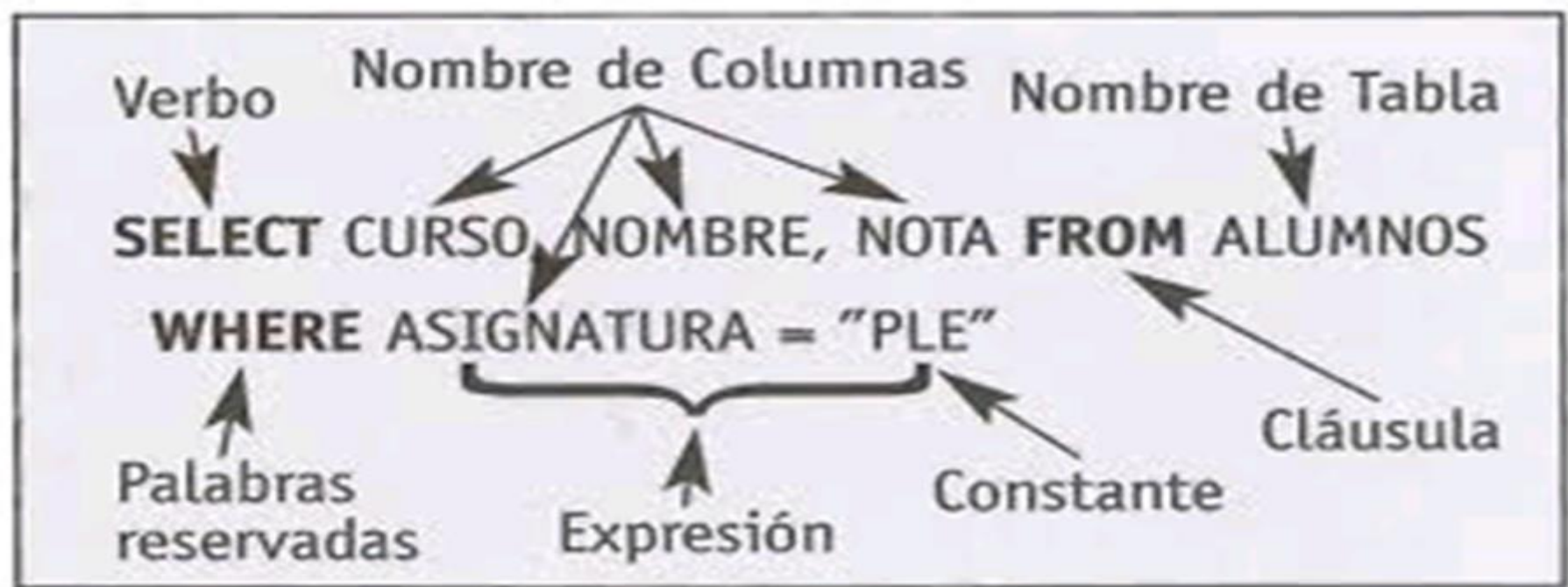
Los pasos para usar transacciones en MySQL son:

1. Iniciar una transacción con la sentencia **BEGIN**.
2. Ejecutar la sentencia DML de SQL (**INSERT**, **UPDATE** o **DELETE**).
3. Confirmar los cambios en la base de datos con el uso de la sentencia **COMMIT**. Sólo hasta ejecutar esta sentencia los cambios serán permanentes en la base de datos.
4. Si ocurriera algún problema, utilizar la sentencia **ROLLBACK** para cancelar los cambios.

Las sentencias SQL

1. Todas las **sentencias SQL** comienzan con un **verbo** que es una **palabra clave** que describe lo que la sentencia hace, como por ejemplo: CREATE, USE, SELECT, INSERT, UPDATE, DELETE, ALTER, etc.
2. Continúa con una o más **cláusulas** en las que se **puede indicar los datos sobre los que actúa la sentencia**.
3. Todas las **cláusulas comienzan con palabras claves** como FROM, WHERE, GROUP BY, HAVING, ORDER BY, etc. Algunas de estas **cláusulas son obligatorias**, otras son **opcionales**.

Las sentencias SQL



Las sentencias SQL

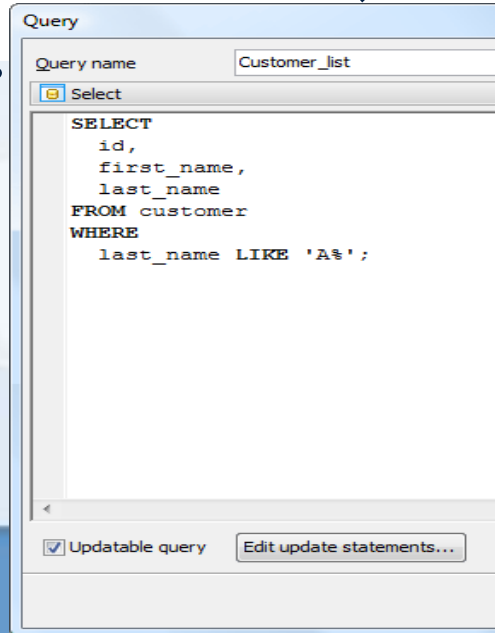
1. El **lenguaje SQL** posee además **capacidades aritméticas** haciendo uso de **operadores** para formar sentencias como: $A < B + 3$.
2. Entre los operadores que se pueden utilizar están los de **comparación**, los **aritméticos** y otros especiales como: BETWEEN (intervalo de valores), LIKE (para comparación) e IN (listar registros para comparación).
3. Además, se pueden utilizar **funciones agregadas** para calcular promedio (AVG), obtener el número de registros de una selección (COUNT), obtener la suma (SUM), obtener el valor máximo (MAX), obtener el valor más bajo (MIN), etc.

MySQL

1. Es un **Gestor de Bases de Datos Relacionales** (SGBDR) diseñado para ser utilizado como **servidor** de una interfaz de usuario distinta al servidor web.
2. Con base de datos se quiere decir que funciona como un **depósito con la información** que se **desea almacenar** a manera de una **colección de datos** organizada en **tablas**.
3. Cada **tabla** se organiza en **filas** y **columnas**, en donde **cada fila representa un registro** con información relacionada con una única ocurrencia de la entidad que representa.
4. Los **registros** están formados con **fragmentos de información** y cada **columna** de una tabla se corresponde con uno de estos fragmentos.

MySQL

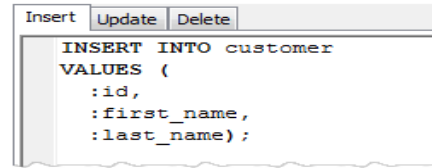
1. Que sea **Sistema de Gestión** indica que es un software que permite realizar **operaciones especiales** con los **datos**, como **insertar**, **recuperar**, **modificar** y **eliminar registros**.



The screenshot shows the MySQL Query window. The 'Query name' field is set to 'Customer_list'. The 'Select' tab is active, displaying the following SQL query:

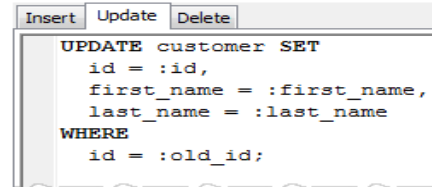
```
SELECT
  id,
  first_name,
  last_name
FROM customer
WHERE
  last_name LIKE 'A%';
```

At the bottom of the window, there is a checkbox labeled 'Updatable query' which is checked, and a button labeled 'Edit update statements...'. The 'Query' title bar is visible at the top.



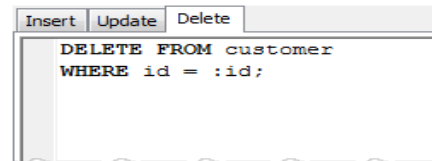
The screenshot shows the MySQL Insert window. The 'Insert' tab is active, displaying the following SQL statement:

```
INSERT INTO customer
VALUES (
  :id,
  :first_name,
  :last_name);
```



The screenshot shows the MySQL Update window. The 'Update' tab is active, displaying the following SQL statement:

```
UPDATE customer SET
  id = :id,
  first_name = :first_name,
  last_name = :last_name
WHERE
  id = :old_id;
```

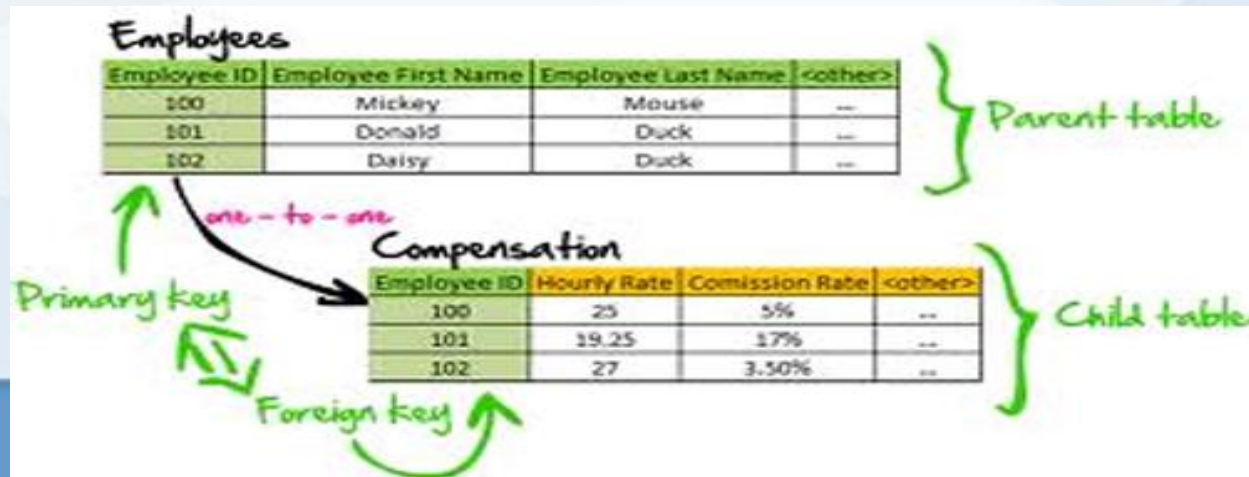


The screenshot shows the MySQL Delete window. The 'Delete' tab is active, displaying the following SQL statement:

```
DELETE FROM customer
WHERE id = :id;
```

MySQL

1. **Relacional** significa que se trata de un **tipo de Sistema de Gestión de Bases de Datos**, conocido como **Base de Datos Relacional**, en el sentido de que **relaciona o encuentra coincidencias** entre la **información almacenada en una tabla** y la que se encuentra **almacenada en otra**, buscando **elementos comunes** entre éstas.



MySQL

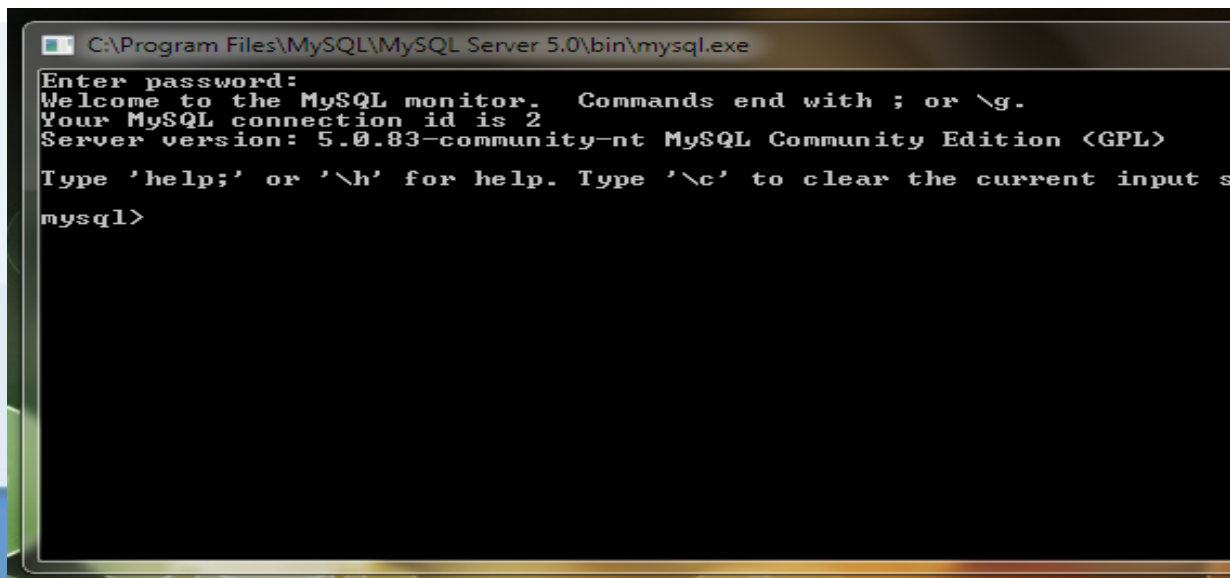
1. La potencia de un SGBDR radica en su **capacidad para extraer datos** de las tablas de forma apropiada y **unir la información de tablas relacionadas** para **solventar respuestas** a preguntas que no podrían responderse mediante una tabla individual.



The SQL component in an RDBMS provides the means to make changes to database records.

Utilizar el cliente MySQL

1. El cliente MySQL es un programa interactivo que permite conexión con un servidor de bases de datos MySQL vía comandos o vía archivos de texto con consultas que luego son ejecutadas en el servidor.






A screenshot of a Windows command prompt window titled "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysql.exe". The window shows the MySQL command-line interface. The text displayed is: "Enter password:", "Welcome to the MySQL monitor. Commands end with ; or \g.", "Your MySQL connection id is 2", "Server version: 5.0.83-community-nt MySQL Community Edition (GPL)", "Type 'help;' or '\h' for help. Type '\c' to clear the current input s", and "mysql>".


```
C:\Program Files\MySQL\MySQL Server 5.0\bin\mysql.exe
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.0.83-community-nt MySQL Community Edition (GPL)


Type 'help;' or '\h' for help. Type '\c' to clear the current input s
mysql>
```


Utilizar el cliente MySQL

1. Para realizar la **conexión al servidor** se necesitará **una cuenta** para poder conectarse y crear bases de datos, tablas y **realizar transacciones** en la base de datos.
2. Para poder crear una cuenta diferente de **root** para un usuario, deberá **iniciar sesión**, precisamente como administrador, haciendo uso del usuario root y una contraseña, que por defecto será nula. Pero esto puede ser diferente, de manera que si tiene contraseña, es preciso que la conozca para poder seguir.

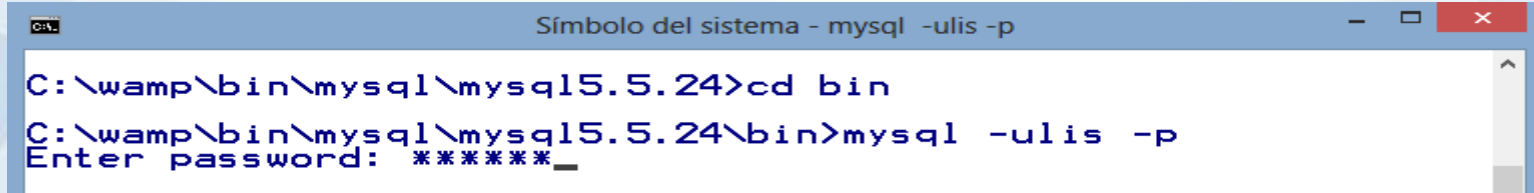
	User	Host	Password	Global privileges ¹	Grant	
<input type="checkbox"/>	debian-sys-maint	localhost	Yes	ALL PRIVILEGES	Yes	
<input type="checkbox"/>	phpmyadmin	localhost	Yes	USAGE	No	
<input type="checkbox"/>	root	127.0.0.1	Yes	ALL PRIVILEGES	Yes	
<input type="checkbox"/>	root	localhost	Yes	ALL PRIVILEGES	Yes	
<input type="checkbox"/>	root	mysqlTest	Yes	ALL PRIVILEGES	Yes	

 [Check All / Uncheck All](#)

 [Add a new User](#)

Iniciar sesión en MySQL

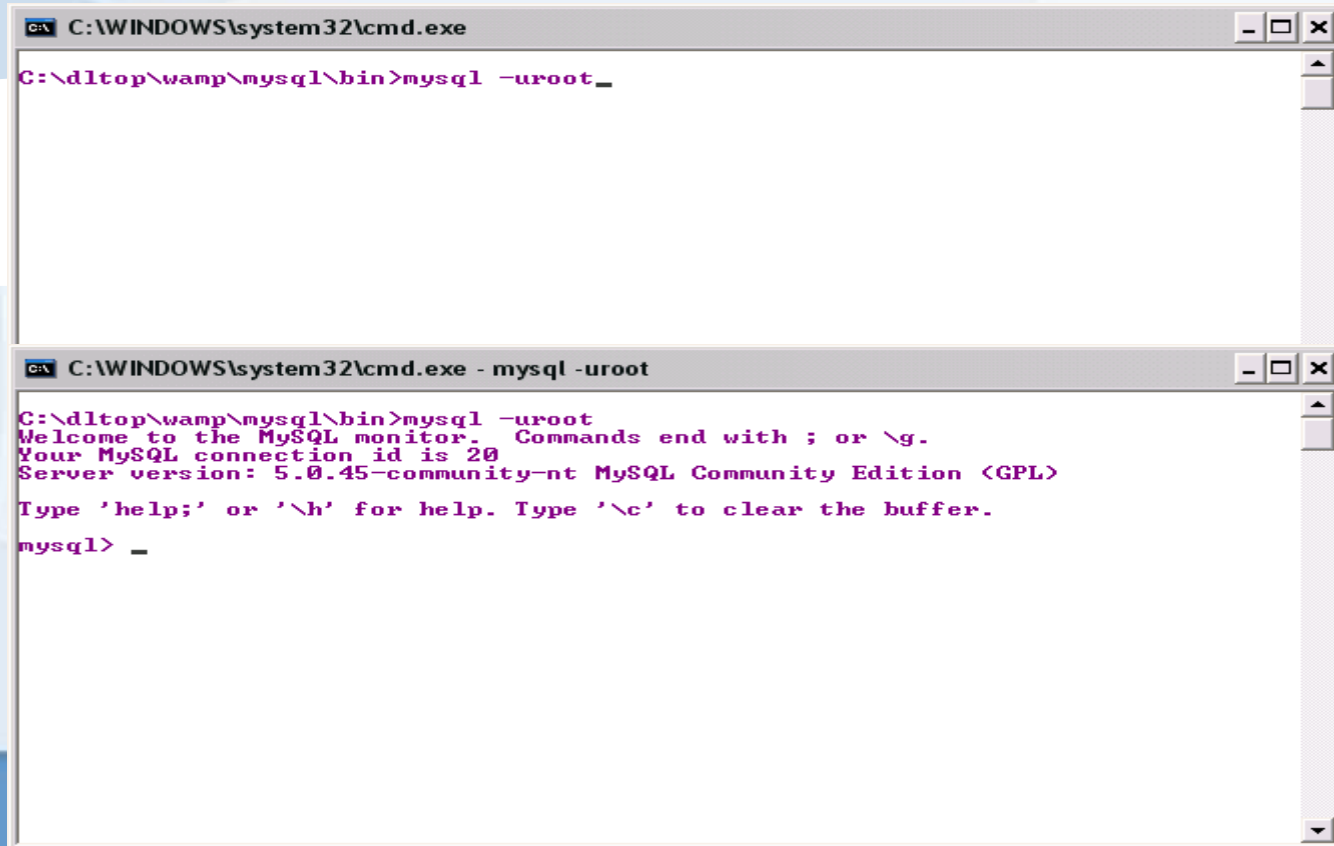
1. Iniciar el cliente de MySQL para conectarse al servidor es fácil utilizando la consola de Windows.
2. Puede utilizar el siguiente comando para hacerlo:
`mysql -u[username] -p[password]`
1. El comando anterior debe ser digitado directamente en la línea de comandos del sistema operativo.



```
Símbolo del sistema - mysql -ulis -p

C:\wamp\bin\mysql\mysql5.5.24>cd bin
C:\wamp\bin\mysql\mysql5.5.24\bin>mysql -ulis -p
Enter password: *****_
```

Conectarse con el servidor MySQL



The image shows two overlapping Windows command prompt windows. The top window has a title bar that reads "C:\WINDOWS\system32\cmd.exe". Its content shows the command `C:\dltop\wamp\mysql\bin>mysql -uroot _` being entered. The bottom window has a title bar that reads "C:\WINDOWS\system32\cmd.exe - mysql -uroot". Its content shows the command `C:\dltop\wamp\mysql\bin>mysql -uroot` being entered, followed by the MySQL server's welcome message: "Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection id is 20. Server version: 5.0.45-community-nt MySQL Community Edition (GPL)". It also shows the prompt `mysql> _`.

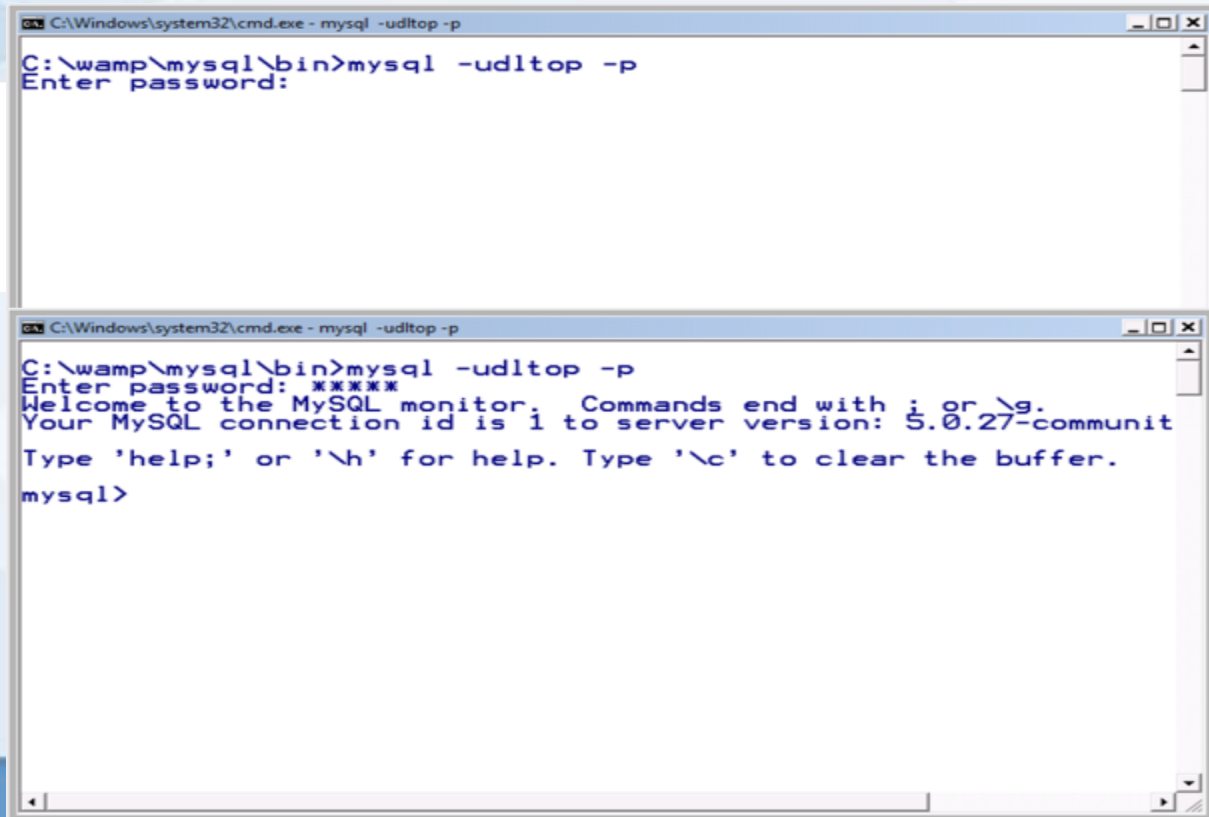
```
C:\WINDOWS\system32\cmd.exe
C:\dltop\wamp\mysql\bin>mysql -uroot _

C:\WINDOWS\system32\cmd.exe - mysql -uroot
C:\dltop\wamp\mysql\bin>mysql -uroot
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 20
Server version: 5.0.45-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> _
```

Conectarse con el servidor MySQL



The image displays two sequential screenshots of a Windows command prompt window, illustrating the process of connecting to a MySQL server. The window's title bar reads "C:\Windows\system32\cmd.exe - mysql -udltop -p".

The first screenshot shows the command prompt at the directory `C:\wamp\mysql\bin>`. The user has entered the command `mysql -udltop -p`, and the prompt now asks for the password: `Enter password:`.

The second screenshot shows the prompt after the password has been entered (represented by asterisks). The MySQL client displays a welcome message: `Welcome to the MySQL monitor. Commands end with ; or \g.` It also shows the connection ID as 1 and the server version as `5.0.27-community-edition`. The prompt then asks for help with `Type 'help;' or '\h' for help. Type '\c' to clear the buffer.` and shows the `mysql>` prompt.

```
C:\Windows\system32\cmd.exe - mysql -udltop -p

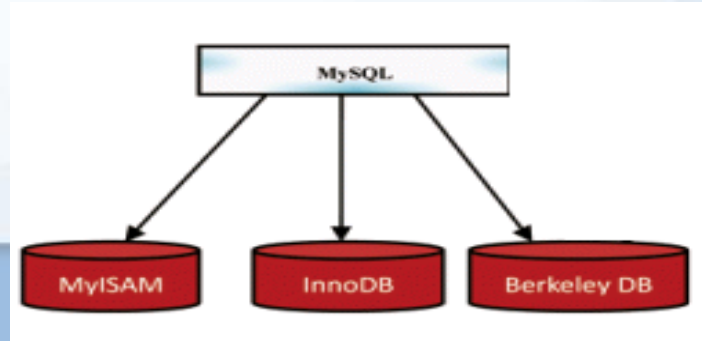
C:\wamp\mysql\bin>mysql -udltop -p
Enter password:

C:\Windows\system32\cmd.exe - mysql -udltop -p

C:\wamp\mysql\bin>mysql -udltop -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 5.0.27-community-edition
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql>
```

Motores de almacenamiento

1. MySQL posee diversos motores de almacenamiento que tratan con distintos tipos de tablas.
2. Los distintos tipos de tablas con los que trabaja MySQL son:
 1. Transaccionales (InnoDB y BDB).
 2. Colecciones de tablas (MERGE).
 3. No transaccionales (MyISAM).
 4. En memoria (MEMORY).
 5. Particionadas (NDB Cluster).



Motor de almacenamiento MyISAM

1. MyISAM es el motor de almacenamiento por defecto en MySQL. Se basa en el código ISAM (primer motor de almacenamiento para la primera versión de mSQL) pero con muchas extensiones útiles.
2. Para cada tabla MySQL se almacenan en disco tres archivos.
3. Los nombres de estos archivos comienzan con el nombre de la tabla y tienen una extensión que indica el tipo de archivo de que se trata: .frm (estructura de la tabla), .MYD (MyData con los datos de la tabla) y .MYI (MyIndex con los índices definidos para la tabla).

Creación de tablas MyISAM

1. Ya se ha mencionado que por defecto, el motor de almacenamiento de MySQL es MyISAM, lo que significa que si no se indica específicamente un motor de almacenamiento, MySQL creará una tabla MyISAM.
2. Para indicar explícitamente que una tabla será MyISAM debe añadir luego de la definición de la tabla la opción ENGINE.
3. De la siguiente forma:

```
CREATE TABLE table_name (  
    field_name type, ...  
) ENGINE = MyISAM;
```


Características de MyISAM

1. Todos los datos se almacenan con el byte menor primero, lo que garantiza la portabilidad, al hacerlos independientes de la máquina y el sistema operativo.
2. Archivos grandes (hasta de 63 bits de longitud), lo cual es soportado en sistemas de archivos y sistemas operativos que soportan archivos grandes.
3. Registros de tamaño dinámico que se fragmentan mucho menos cuando se mezclan registros borrados con actualizaciones e inserciones.
4. Número máximo de índices por tabla 64 y máximo número de columnas por índice 16.

Características de MyISAM

1. Longitud máxima para una clave es 1000 bytes. Para el caso de que la clave sea mayor a 250 bytes se utiliza un tamaño de bloque mayor de 1024 bytes.
2. Los campos de tipo BLOB y TEXT pueden indexarse.
3. Los valores NULL se permiten en columnas indexadas.
4. El tratamiento interno de una columna AUTO_INCREMENT en tablas MyISAM, actualiza automáticamente esta columna para operaciones INSERT y UPDATE.
5. Un valor de columna AUTO_INCREMENT puede cambiarse con ALTER TABLE.
6. Etc.

Motor de almacenamiento InnoDB

1. InnoDB es un motor de almacenamiento transaccional con capacidades de confirmación de transacción (COMMIT), cancelación (ROLLBACK) y recuperación de fallas.
2. InnoDB soporta integridad referencial mediante la utilización de restricciones de llave foránea.
3. InnoDB realiza bloqueos a nivel de fila y también proporciona funciones de lectura consistente sin bloqueo al estilo Oracle en sentencias SELECT.
4. InnoDB se diseñó para obtener el máximo rendimiento al procesar grandes volúmenes de datos. Se dice que ningún otro motor de bases de datos relacionales en disco iguala su eficiencia en el uso de CPU.

Motor de almacenamiento InnoDB

1. InnoDB ofrece una confiabilidad y consistencia superior a MyISAM.
2. InnoDB crea un archivo que se corresponde directamente con una determinada tabla. Este archivo tiene extensión .frm y posee el mismo nombre de la tabla.
3. El archivo .frm se almacena en la carpeta de la base de datos a la que pertenece la tabla.



Motor de almacenamiento InnoDB

1. Los contenidos de las tablas se representan utilizando espacios de tablas, de una de dos maneras:
 1. Espacio de tabla compartido (único archivo específico de la tabla es el archivo .frm)
 2. Espacios de tabla individuales (cada tabla tendrá dos archivos específicos: .frm, como es habitual y .ibd con los datos e índices de la tabla).
2. InnoDB mantiene un diccionario de datos interno que contiene información sobre cada una de sus tablas. Este diccionario se almacena en el espacio de tablas compartido, que por consiguiente, es necesario aunque utilicemos espacios de tablas individuales.

Creación de tablas InnoDB

1. Para crear una tabla InnoDB, al momento de la definición de la estructura de la tabla se debe especificar la opción ENGINE o TYPE en la sentencia SQL.
2. De la siguiente forma:

```
CREATE TABLE table_name(  
    field_name type, ...  
) ENGINE|TYPE = InnoDB;
```


Comparativa entre MyISAM e InnoDB

MyISAM

Las tablas de MyISAM son mas simples, por lo tanto, las personas principiantes en bases de datos se les recomienda el uso de MyISAM.

Bloqueo de tablas.

Permite la característica ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad), garantizando la integridad de las tablas.

Aumento de rendimiento con el uso constante del comando SELECT.

InnoDB

Integridad de datos. Cuando el contenido se modifica por sentencias INSERT, DELETE o UPDATE, la integridad de los datos puede perderse de muchas maneras. InnoDB ayuda a evitar esto.

Mayor velocidad en general a la hora de recuperar datos.

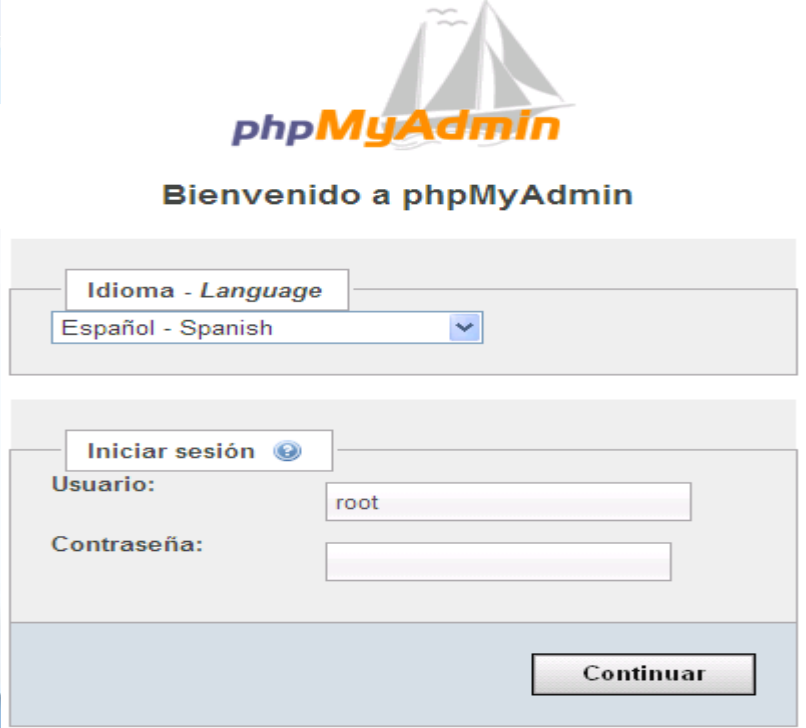
Recomendado para aplicaciones en las que se domina sentencias como SELECT ante los INSERT y UPDATE.

Ausencia de ATOMICIDAD= + velocidad
Soporte de transacciones.

Ventajas de MyISAM e InnoDB

phpMyAdmin

1. Existen aplicaciones desarrolladas en PHP que se utilizan para facilitar el trabajo con bases de datos MySQL. Una de las más utilizadas es phpMyAdmin que es una aplicación web que permite realizar todas las operaciones de administración de forma más conveniente e intuitiva.



The screenshot shows the phpMyAdmin login interface. At the top, there is a logo featuring a sailboat and the text 'phpMyAdmin'. Below the logo, it says 'Bienvenido a phpMyAdmin'. There are two main sections: a language selection section and a login section. The language section has a dropdown menu currently set to 'Español - Spanish'. The login section has a button labeled 'Iniciar sesión' with a help icon. Below this, there are input fields for 'Usuario:' (containing 'root') and 'Contraseña:'. At the bottom right, there is a 'Continuar' button.

phpMyAdmin

Bienvenido a phpMyAdmin

Idioma - Language

Español - Spanish

Iniciar sesión

Usuario: root

Contraseña:

Continuar

phpMyAdmin

1. La versión 3.5.3 de phpMyAdmin muestra significativos cambios respecto a sus versiones de la generación 2 y las primeras de la versión 3, con una interfaz de usuario.
2. Un menú más simplificado e intuitivo y un área de navegación mejor organizada en la parte izquierda que facilita el acceso a las bases de datos.
3. Al acceder a las bases de datos se muestra un fácil acceso a las operaciones que se pueden realizar con las tablas y con la base de datos misma.

phpMyAdmin

phpMyAdmin

localhost

Bases de datos SQL Estado actual Usuarios Exportar Importar Más

(Tablas recientes) ...

book_club
ci_series
db_drupal
empresa
fet
_dokeos_main
_dokeos_stats
_dokeos_user
information_schema
joomlalet2012
joomlaletudb2
libros_editorial
moodleaci
moodlefetudb
mysql
países
prensa

Configuraciones generales

Cambio de contraseña

Cotejamiento de la conexión al servidor ⓘ :
utf8_general_ci

Configuraciones de apariencia

Idioma - Language ⓘ : Español - Spanish

Tema: pmahomme

• Tamaño de fuente: 82%

Más configuraciones

Servidor de base de datos

- Servidor: MySQL host info: localhost via TCP/IP
- Programa: MySQL
- Versiones de programa: 5.1.36-community-log - MySQL Community Server (GPL)
- Versión del protocolo: 10
- Usuario: root@localhost
- Conjunto de caracteres del servidor: UTF-8 Unicode (utf8)

Servidor web

- Apache/2.2.11 (Win32) PHP/5.3.0
- Versión del cliente de base de datos: libmysql - mysqlnd 5.0.5-dev - 081106 - \$Revision: 1.3.2.27 \$
- extensión PHP: mysqli ⓘ

phpMyAdmin

FIN

Lenguajes Interpretados en el Servidor