

Tecnológico Nacional de México

Instituto Tecnológico de Culiacán



Proyecto 1 MySQL

Unidad 1

Perspectiva de la administración de base de datos

Materia:

Administración de Bases de Datos

Docente:

M.C. Daniel Esparza Soto

Integrantes:

Astorga Ochoa César Alfredo

Medrano Iturrios Angel Raul

Grupo:

6:00 pm - 7:00 pm

Fecha: viernes 11 de febrero de 2022

Contenido:

1. Resumen de su historia	3
2. Versiones	4
3. Características	5
4. Manual del proceso de instalación	6
5. Manual de como realizar una conexión al servidor	7
6. Manual de como ejecutar instrucciones SQL	8
7. Script para cargar la base de datos Northwind	9
8. Script para la familia de vistas de la base de datos Northwind	10
9. Script con procedimiento almacenado para insertar y modificar la tabla Territories	11
10. Script con un trigger que no permita eliminar la tabla Territories	12

1. Resumen de su historia

La concepción de **MySQL** se remonta a los años 90 's siendo más específicos en el año 1995. Y es aquí donde una compañía de origen sueca llamada **MySQL AB** decide desarrollar esta plataforma, con la finalidad de ofrecer un sistema gestor de bases de datos (**DBMS**) para usuarios domésticos y profesionales. El desarrollo de la plataforma estuvo a cargo de **Michael Widenius** y dos de sus compañeros; **David Axmark** y **Allan Larsson**.

Al primero de ellos es a quien se le atribuye el mérito de haber construido este DBMS y esto es debido a que **Widenius** comenzó a utilizar **mSQL** para conectar tablas mientras se encontraba desarrollando aplicaciones con **BASIC** y también con rutinas ISAM de bajo nivel. De esta manera fue donde se dio cuenta que mSQL no era lo suficientemente rápido ni flexible para sus necesidades por lo cual, acudió a implementar nuevas funciones trayendo consigo mismas MySQL.

Sobre el origen de MySQL se desconoce de donde proviene, pero se sabe que las librerías han incluido el prefijo "my" desde hace más de diez años y por otra parte, se especula que el nombre es debido a que la hija de uno de los 3 desarrollos llevaba como nombre My.

En 2008, MySQL fue adquirido por Sun Microsystems, quien en 2010 fue comprado por Oracle Corporation..

2. Versiones

Gráfico donde se muestran los años en que fueron lanzadas las versiones de MySQL hasta 2015

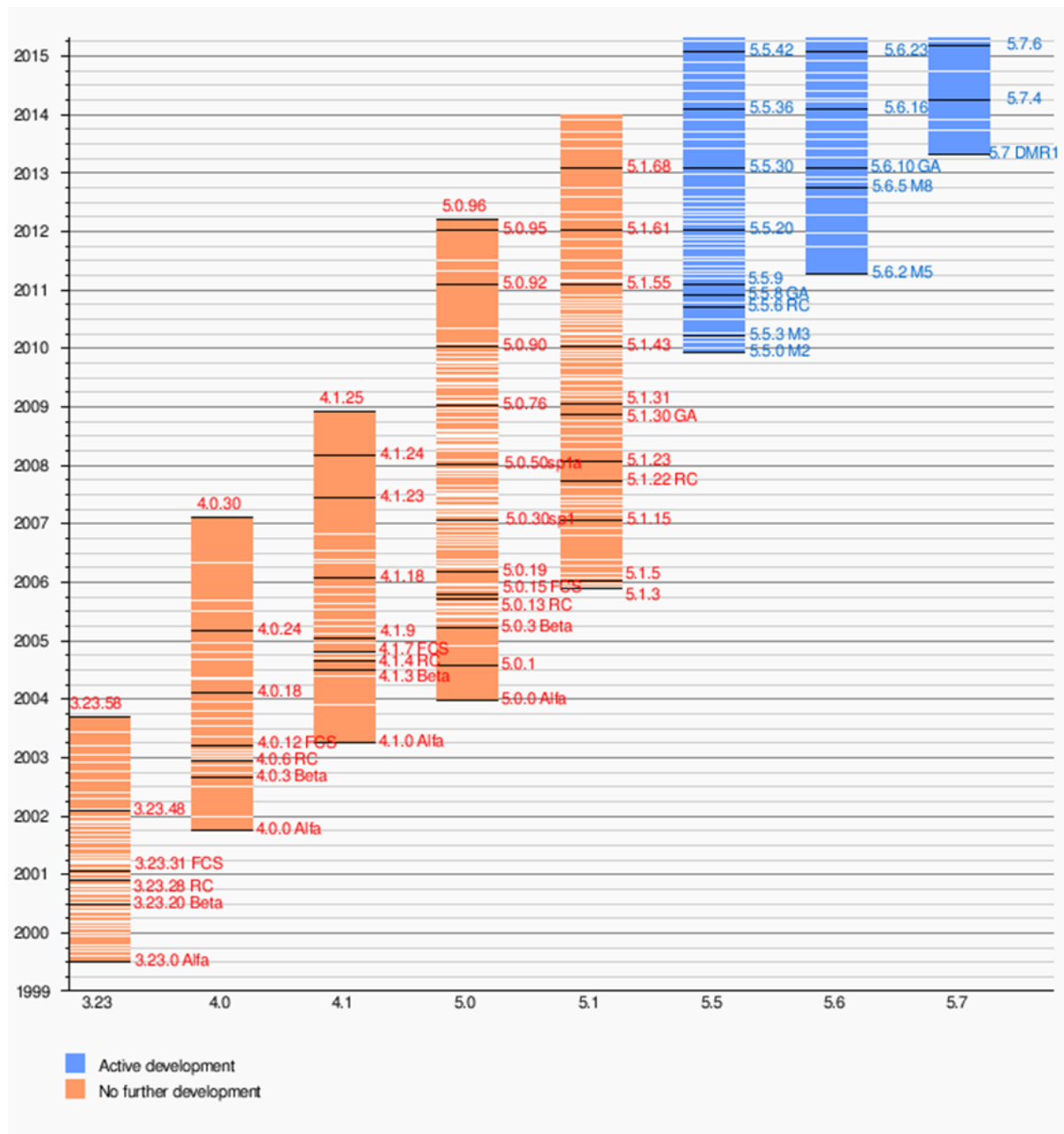


Tabla con algunas de las implementaciones que se fueron dando en cada actualización hasta la versión 5.1

Función	Versión
Replicación	3.23
Búsqueda de texto completo en tablas MyISAM	3.23
Transacciones con tablas BDB	3.23.34
Transacciones con tablas InnoDB	3.23.34
Integridad referencial para tablas InnoDB	3.23.34
<i>ELIMINAR</i> y <i>ELIMINAR</i> en varias mesas	4.0
<i>ACTUALIZAR</i> en varias mesas	4.0
<i>UNIÓN</i> (unir varios <i>SELECCIONE</i> resultados)	4.0
Caché de consultas (acelera los comandos SQL repetidos)	4.0
Biblioteca MySQL incrustada	4.0
Conexiones encriptadas (Secure Socket Layer, SSL)	4.0
Copia de seguridad en caliente para InnoDB (producto complementario comercial)	4.0
GPL para bibliotecas cliente (anteriormente LGPL)	4.0

SubSELECCIONAR s	4.1
Soporte Unicode (UTF8 y UCS2 = UTF16)	4.1
Soporte GIS (<i>GEOMETRÍA</i> tipo de datos, índice de árbol R)	4.1
Declaraciones preparadas (comandos SQL con parámetros)	4.1
<i>ENROLLAR</i> extensión para <i>AGRUPAR POR</i>	4.1
Mejor cifrado de contraseña en la tabla mysql.user	4.1
Archivos de espacio de tabla InnoDB individuales para cada tabla	4.1
<i>VARCHAR</i> columnas con más de 255 caracteres	5,0
Auténtico <i>UN POCO</i> tipo de datos	5,0
Procedimientos almacenados (SP)	5,0
Triggers (ejecución automática de código SQL)	5,0 / 5,1
Puntos de vista	5,0
Formato de espacio de tabla InnoDB que ahorra espacio	5,0 / 5,1
Nueva gestión de esquemas (diccionario de datos, base de datos INFORMATION_SCHEMA)	5,0
<i>UNIÓN EXTERIOR COMPLETA</i>	5.1

Después de la versión 5 pasaron algunas cosas con MySQL, esta es una lista con los acontecimientos que pasaron después de esa versión.

- Sun Microsystems adquirió MySQL AB en 2008.
- Versión 5.1: lanzamiento de producción el 27 de noviembre de 2008 (planificador de eventos, particionamiento, API de complementos, replicación basada en filas, tablas de registro del servidor)
- MySQL 5.1 y 6.0-alpha mostraron un bajo rendimiento cuando se usaban para el almacenamiento de datos, en parte debido a su incapacidad para utilizar múltiples CPU núcleos para procesar una sola consulta.
- MySQL Server 6.0.11-alpha se anunció el 22 de mayo de 2009 como la última versión de la línea 6.0. El desarrollo futuro del servidor MySQL utiliza un nuevo modelo de lanzamiento. Las características desarrolladas para 6.0 se están incorporando en futuras versiones.
- Oracle adquirió Sun Microsystems el 27 de enero de 2010.
- El trabajo en la versión 6 se detuvo después de la adquisición de Sun Microsystems. El producto MySQL Cluster utiliza la versión 7. Se tomó la decisión de saltar a la versión 8 como el próximo número de versión principal.

3. Características

Una de las principales características de MySQL es que **puede utilizarse en diferentes plataformas** al ser software libre es compatible Windows, Linux, Mac OS...

Cuenta con **múltiples motores de almacenamiento** para adaptarse a cada entorno, **es rápido** y soporta una gran cantidad de tipos de datos.

En seguridad MySQL **tiene un sistema de contraseñas** que permite verificaciones basadas en host por lo cual lo vuelve confiable.

Cuenta también con una **gran comunidad** y es **escalable** y **fácil de aprender**.

Inicialmente MySQL no soportaba algunos de los elementos más importantes de las bases de datos relacionales tales como integridad referencial y transacciones pero esto ha quedado atrás. Una desventaja es que la mayoría de las utilidades no están documentadas. Por otro lado, recopilando características y ventajas podemos decir que MySQL es:

- Rápido y con gran rendimiento
- Seguro
- Fácil de aprender.
- Escalable
- Compatible entre sistemas operativos.

4. Manual del proceso de instalación

Primero instalaremos XAMPP que es donde se aloja nuestra base de datos para MySQL, para ello nos dirigiremos a la pagina de descarga de XAMPP

<https://www.apachefriends.org/es/index.html>, ya dentro de la página damos clic en la parte de descargar




Y nos mandara a la página de descarga donde nos ofrece XAMPP para diferentes sistemas operativos, en este caso descargamos el de Windows y descargamos la última versión disponible

← → ↻ <https://www.apachefriends.org/es/download.html>

Descargar

XAMPP es una distribución de Apache fácil de instalar que contiene MariaDB, PHP y Perl. Simplemente descarga y ejecuta el instalador. ¡Es así de fácil!

 **XAMPP para Windows 7.4.27, 8.0.15 & 8.1.2**

Versión	¿Qué está incluido?	Suma de comprobación		Tamaño
7.4.27 / PHP 7.4.27	¿Qué está incluido?	md5 sha1	Descargar (64 bit)	160 Mb
8.0.15 / PHP 8.0.15	¿Qué está incluido?	md5 sha1	Descargar (64 bit)	161 Mb
8.1.2 / PHP 8.1.2	¿Qué está incluido?	md5 sha1	Descargar (64 bit)	164 Mb

Documentación/FAQs

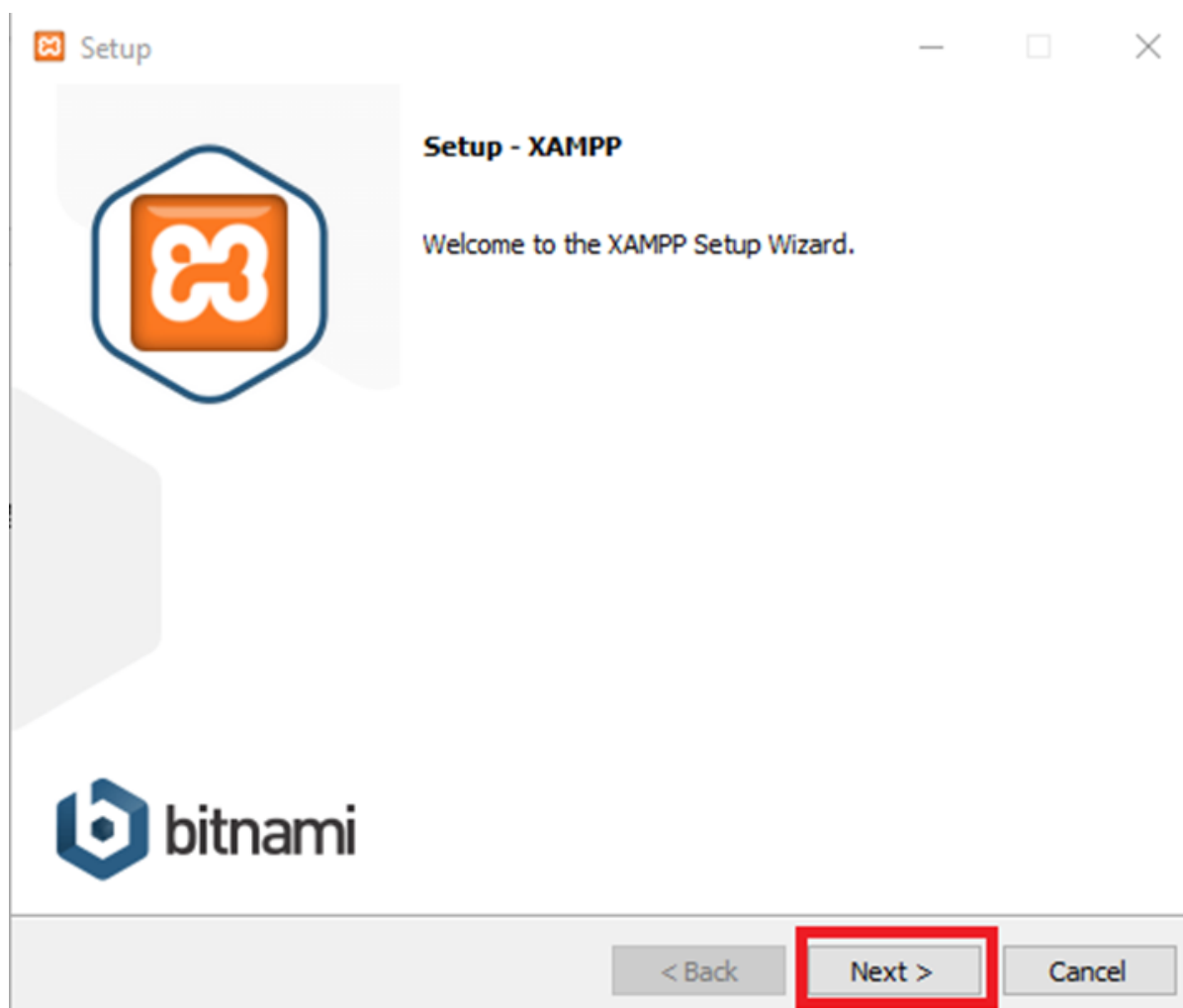
No hay un manual para XAMPP. Escribimos la documentación en forma de preguntas frecuentes (FAQs). ¿Tienes una pregunta que no está respondida? Prueba los Foros de Stack Overflow.

- [Linux Preguntas frecuentes](#)
- [Windows Preguntas frecuentes](#)
- [OS X Preguntas frecuentes](#)
- [OS X XAMPP-VM Preguntas frecuentes](#)

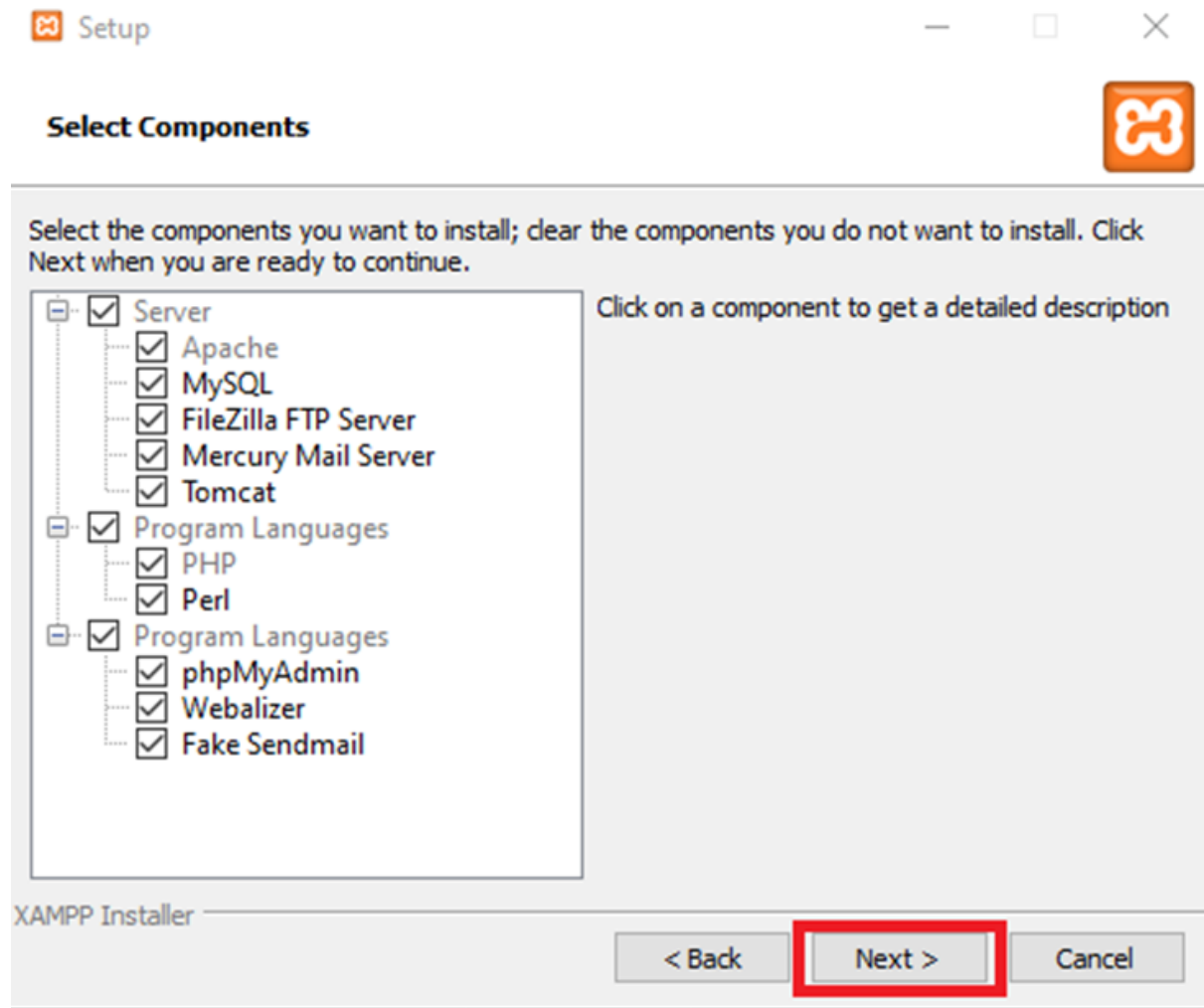
Complementos

Y damos clic en descargar.

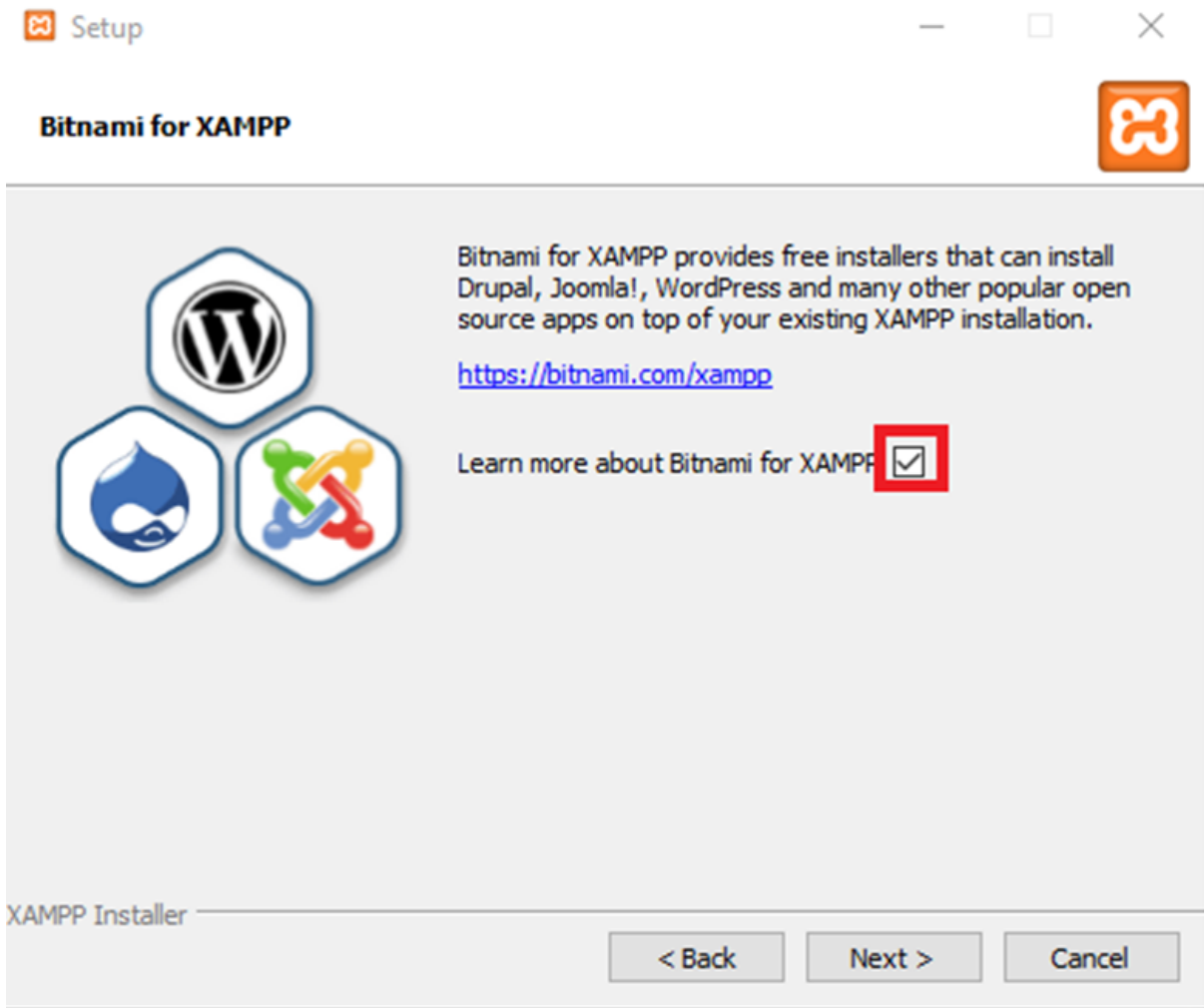
Ya descargado procederemos a la instalación, ejecutamos el archivo descargado, una vez ejecutado nos aparecerá una ventana en la que daremos next



Después nos aparecerá una pantalla en donde nos muestra que se va a instalar, la podemos dejar justo como esta o podemos elegir lo que nosotros queramos instalar, y luego volvemos a dar next



Seguiremos dando next hasta que nos aparezca esta pantalla en la que nos pregunta si queremos saber mas sobre una biblioteca de instaladores, ahí le damos clic en la flecha para desmarcarla y damos clic en next



En la siguiente pestaña volvemos a dar next y empezará la instalación de XAMPP, una vez terminada la instalación damos clic en finalizar y listo, ya tenemos XAMPP instalado

Ahora pasaremos a instalar MySQL Workbench y para ello nos dirigiremos al siguiente enlace <https://dev.mysql.com/downloads/workbench/> una vez ahí daremos clic en download

General Availability (GA) Releases Archives

MySQL Workbench 8.0.28

Select Operating System:
Microsoft Windows

Recommended Download:

MySQL Installer for Windows

All MySQL Products. For All Windows Platforms. In One Package.

Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.

Windows (x86, 32 & 64-bit), MySQL Installer MSI [Go to Download Page >](#)

Other Downloads:

Windows (x86, 64-bit), MSI Installer (mysql-workbench-community-8.0.28-winx64.msi)	8.0.28	42.7M	Download
			MD5: c3db3ca9810964283b0821bcf021be59 Signature

En la siguiente página damos clic en no gracias

MySQL Community Downloads

Login Now or Sign Up for a free account.

An Oracle Web Account provides you with the following advantages:

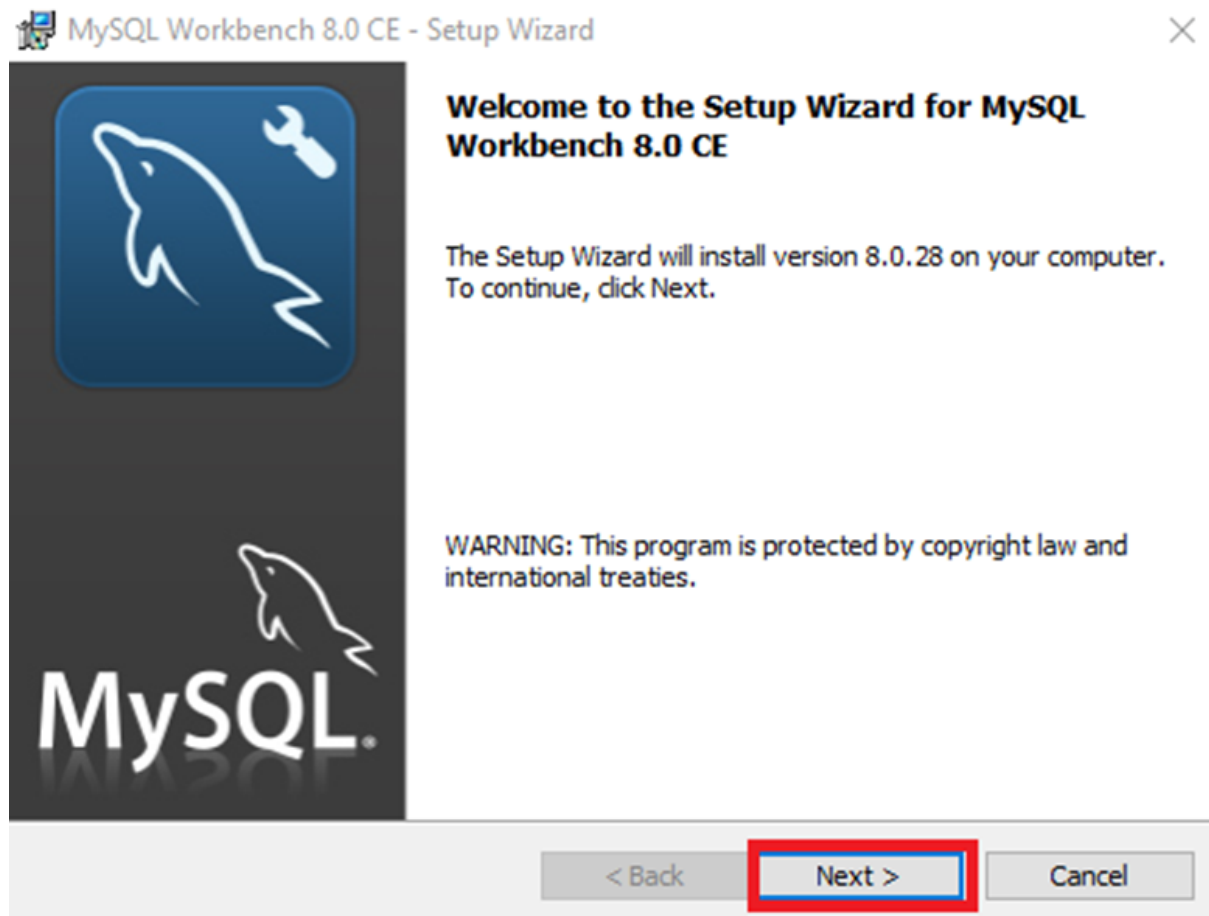
- Fast access to MySQL software downloads
- Download technical White Papers and Presentations
- Post messages in the MySQL Discussion Forums
- Report and track bugs in the MySQL bug system

[Login »](#) using my Oracle Web account [Sign Up »](#) for an Oracle Web account

MySQL.com is using Oracle SSO for authentication. If you already have an Oracle Web account, click the Login link. Otherwise, you can signup for a free account by clicking the Sign Up link and following the instructions.

[No thanks, just start my download.](#)

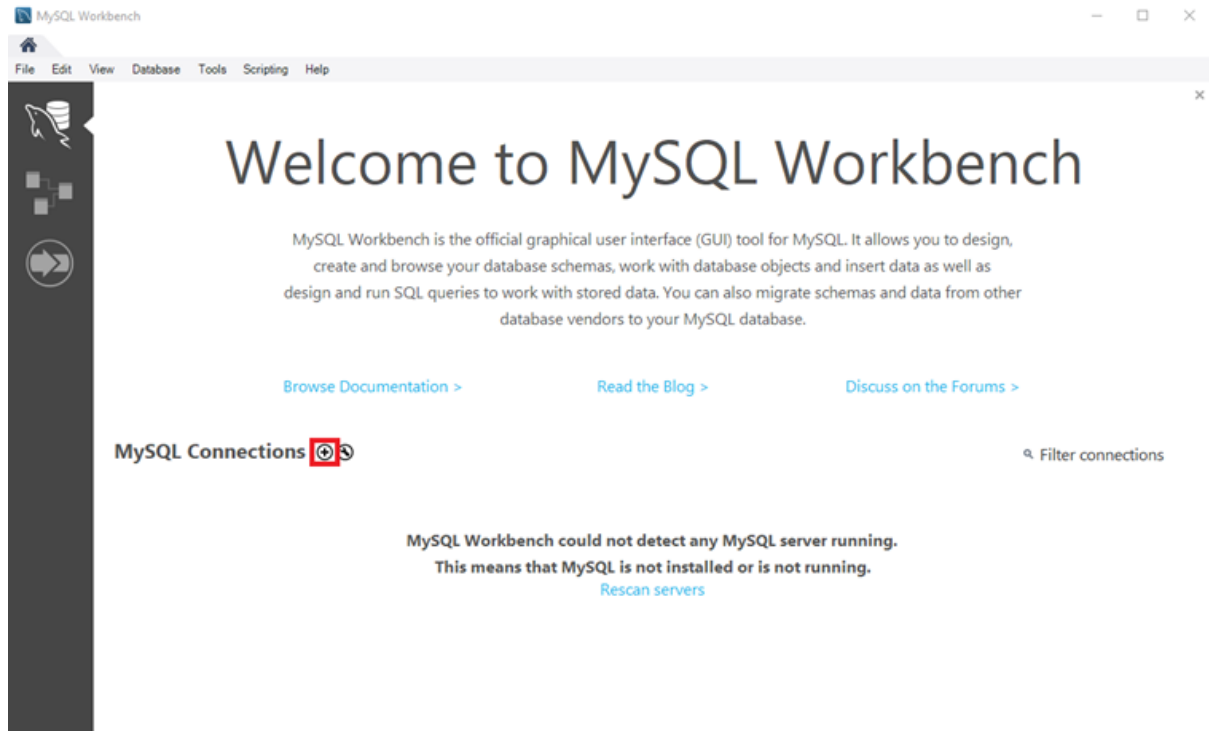
Y después comenzará la descarga, una vez terminada la descarga abrimos el instalador dándole clic, nos aparecerá una ventana en la que daremos clic a next



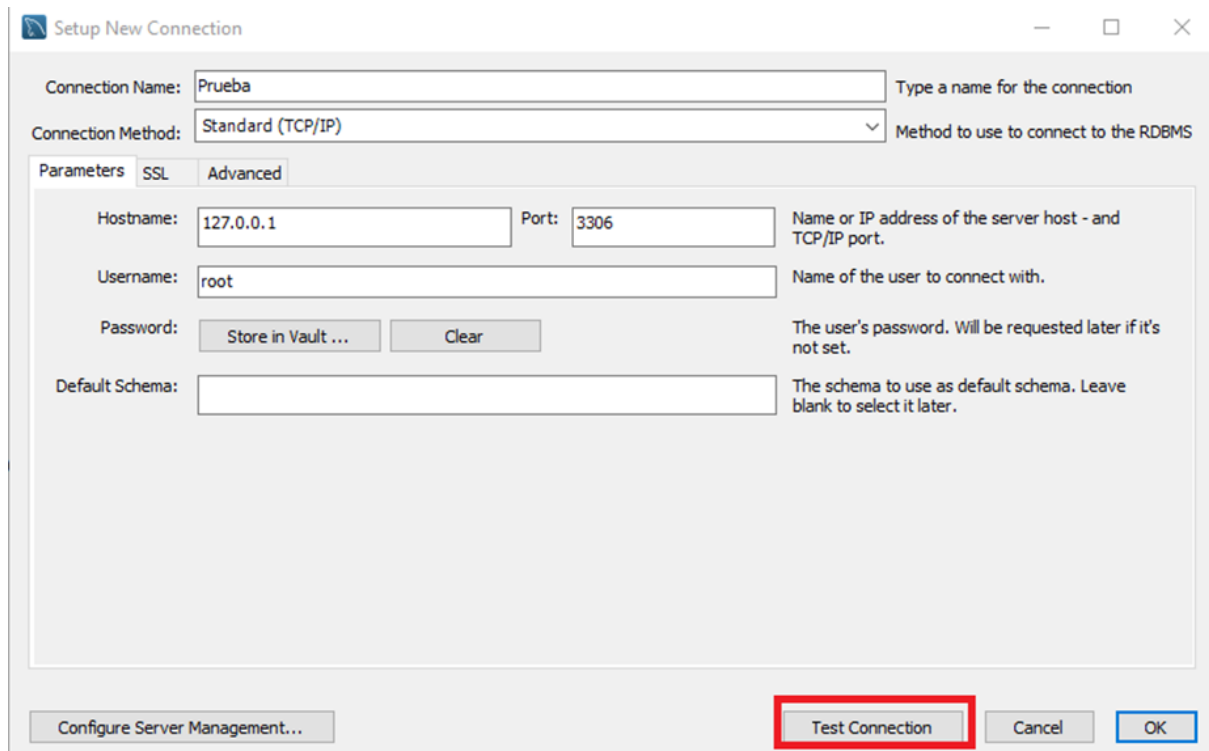
Después de eso le daremos clic en next hasta que nos aparezca una solicitud de permiso en la que daremos clic en sí y después procederá a instalarse el MySQL Workberhch.

Una vez instalado procederemos a abrir XAMPP y después MySQL, ya dentro por lo general aparece una conexión, pero en caso de que no salga nada se deberá hacer los siguiente.

Primero le daremos clic al más en MySQL connections



Después nos aparecerá una ventana en la que nos pedirá primero el nombre de la conexión, ahí le podemos el nombre que nosotros queramos y le damos clic a test connection



Nos aparecerán dos ventanas en la primera damos continuar y la segunda nos mostrará un mensaje que nos dirá que la conexión es exitosa y daremos clic en ok, y le damos clic en ok otra vez

Setup New Connection

Connection Name: Type a name for the connection

Connection Method: Method to use to connect to the RDBMS

Parameters SSL Advanced

Hostname: Port: Name or IP address of the server host - and TCP/IP port.

Username: Name of the user to connect with.

Password: The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

Y listo ya tenemos todo para utilizar MySQL.

5. Manual de como realizar una conexión al servidor

Para realizar una conexión al servidor realice las siguientes indicaciones:

Primero, busque la aplicación **XAMPP Control Panel** fig. 1 y ejecute la.

A continuación, comience el servicio de MySQL pulsando el botón de **"Start"**

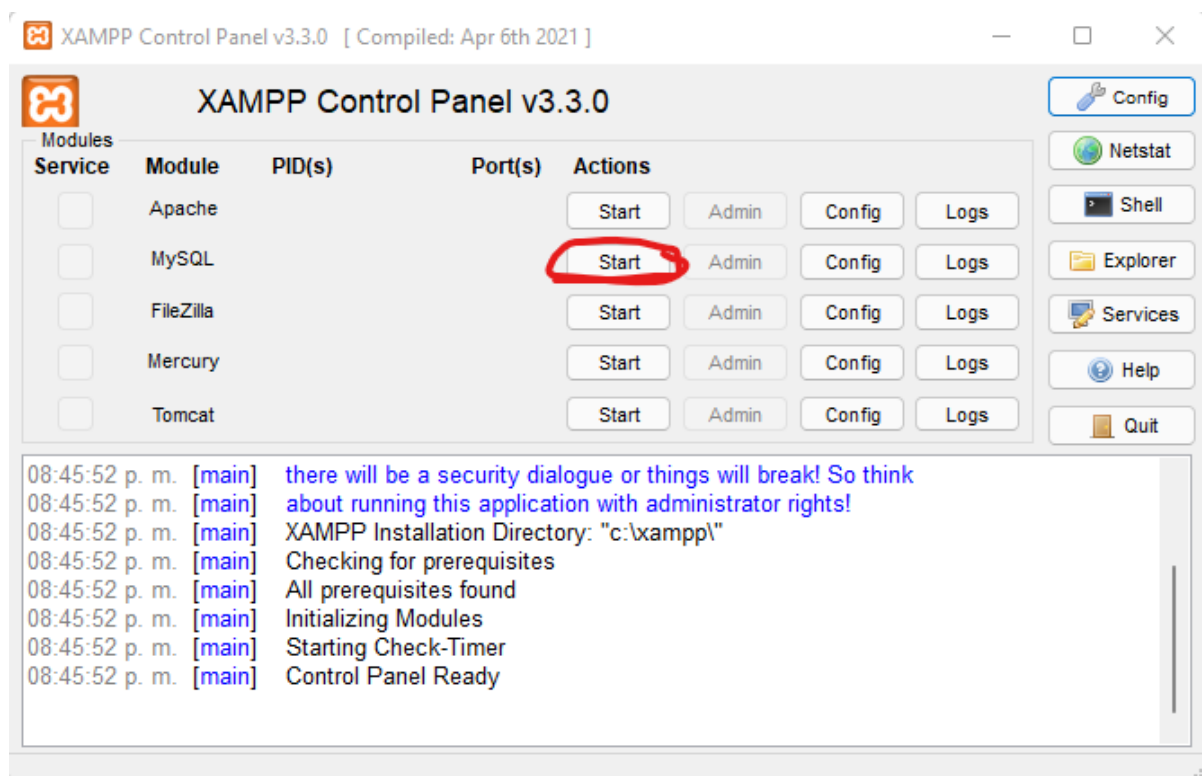


fig. 1. XAMPP Control Panel

Enseguida, notará que el servicio ha comenzado y se tornará a un color verde, Después de esto, puede cerrar esta ventana.

Ahora es necesario ejecutar la aplicación de **MySQL Workbench** fig. 2 para continuar con el proceso.

Una vez iniciada la aplicación antes mencionada, encontrará una conexión ya definida de click sobre la misma.

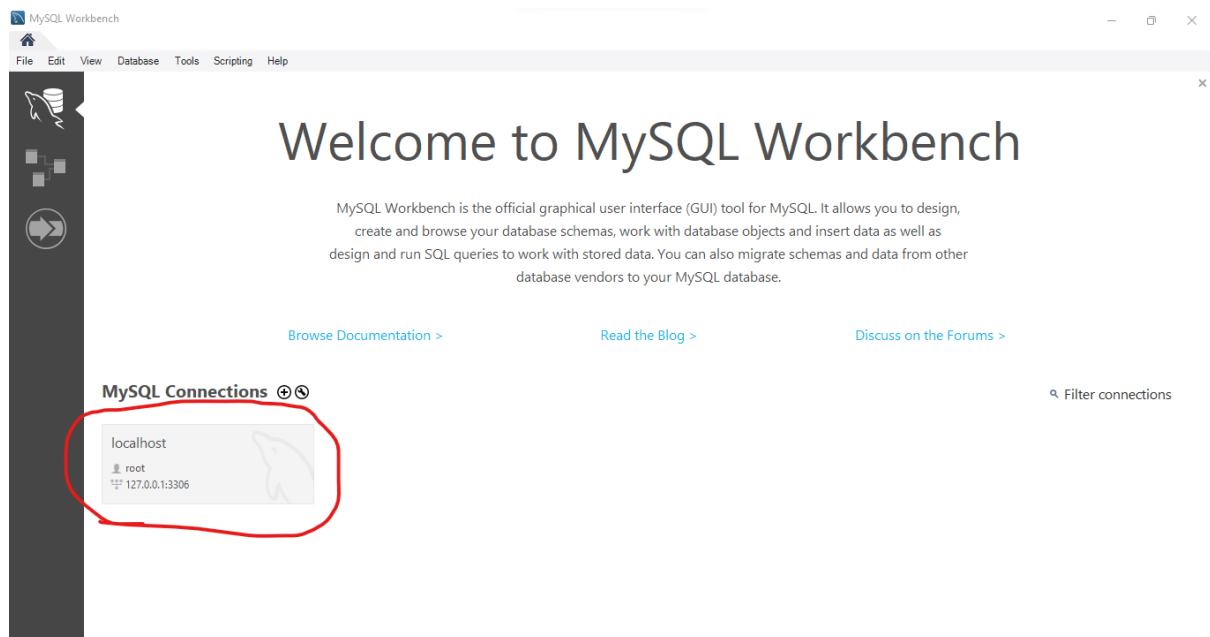
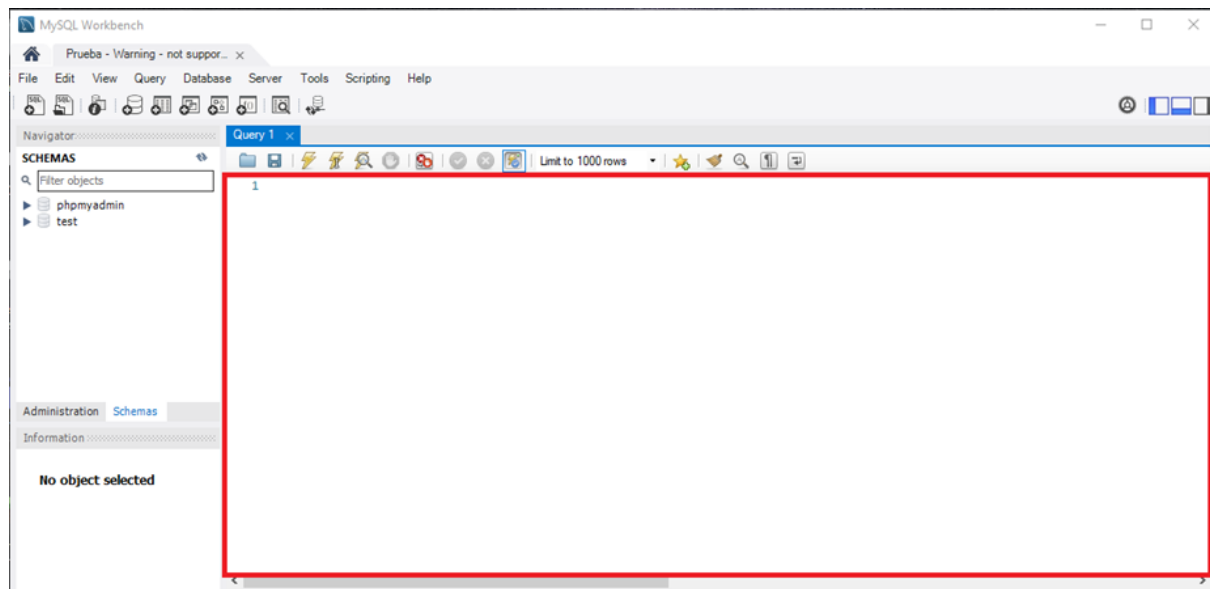


fig. 2. MySQL Workbench

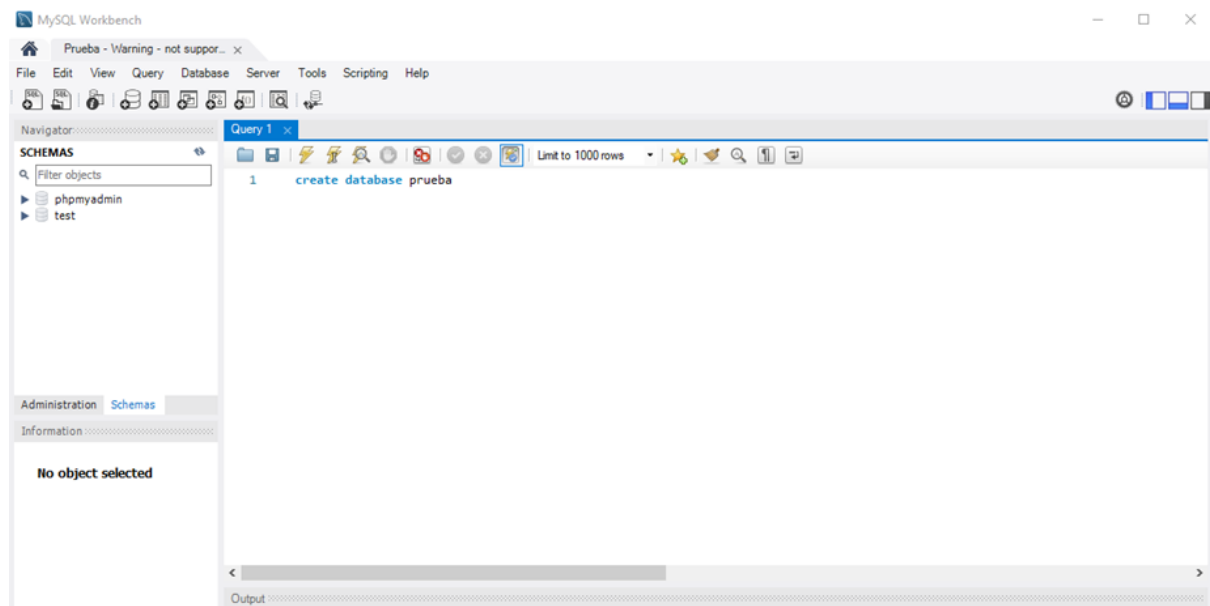
De esta manera lograra entablar una conexión con el servidor y podrá ejecutar instrucciones SQL, dirijase a la sección 6 para obtener información acerca de esto.

6. Manual de como ejecutar instrucciones SQL

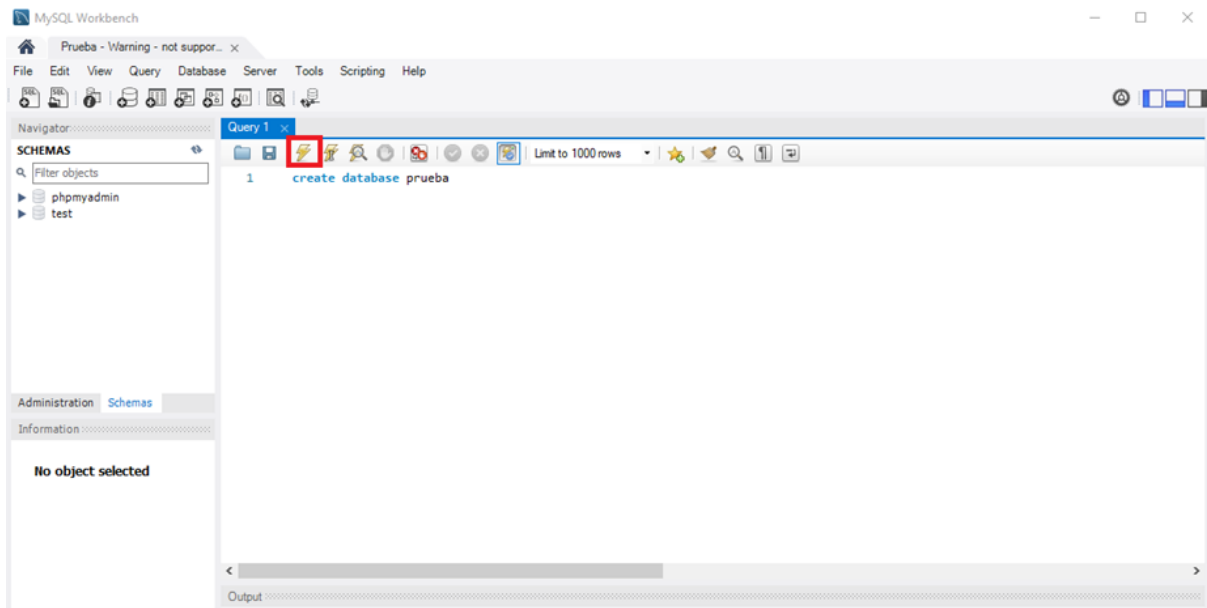
Para empezar a ejecutar instrucciones en MySQL primero nos debemos posicionar en el área del query



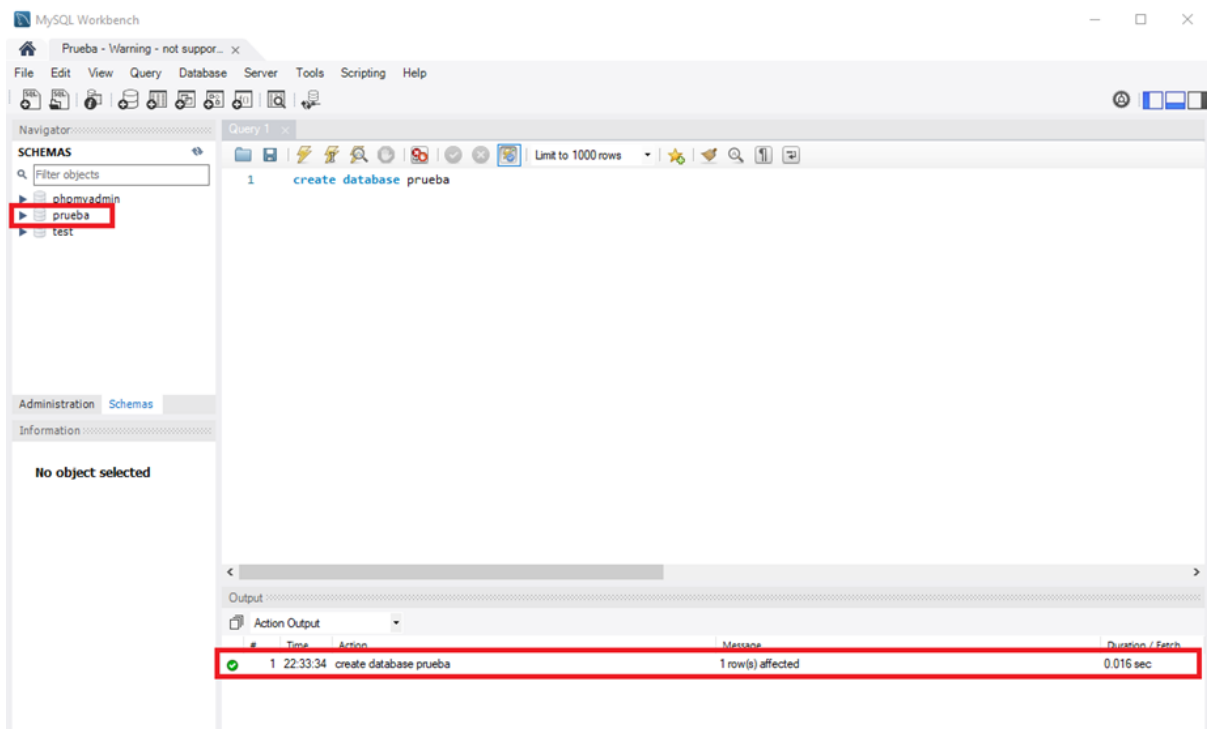
En esa área empezamos a escribir nuestras instrucciones, un ejemplo, crear una base de datos



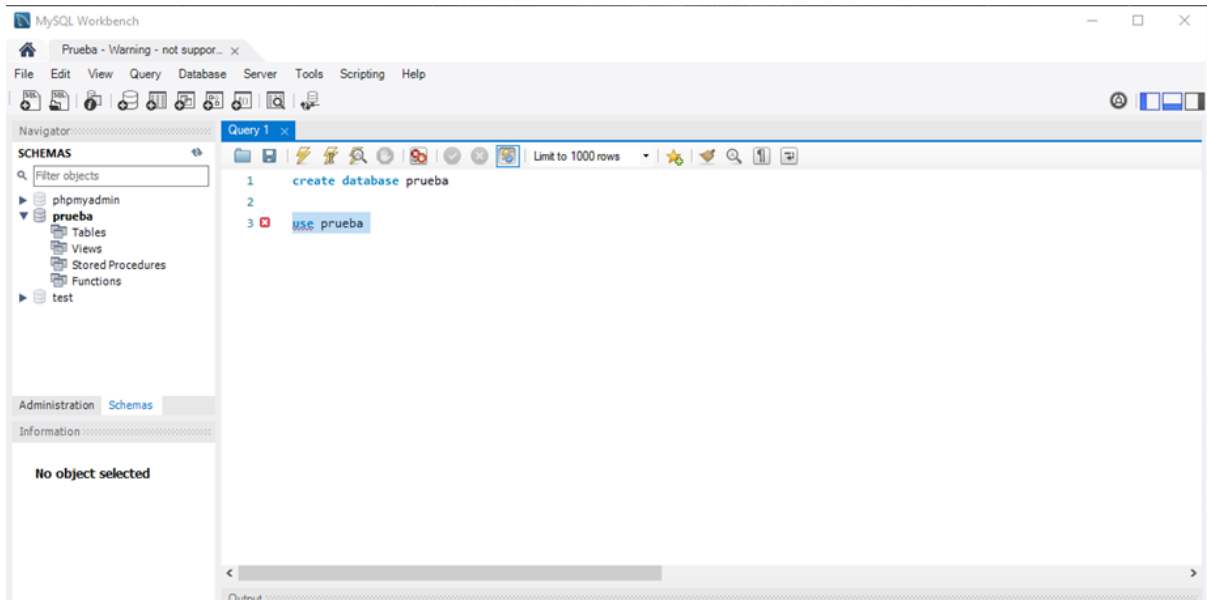
Ya escrita nuestra instrucción damos clic en ejecutar (el icono del rayo)



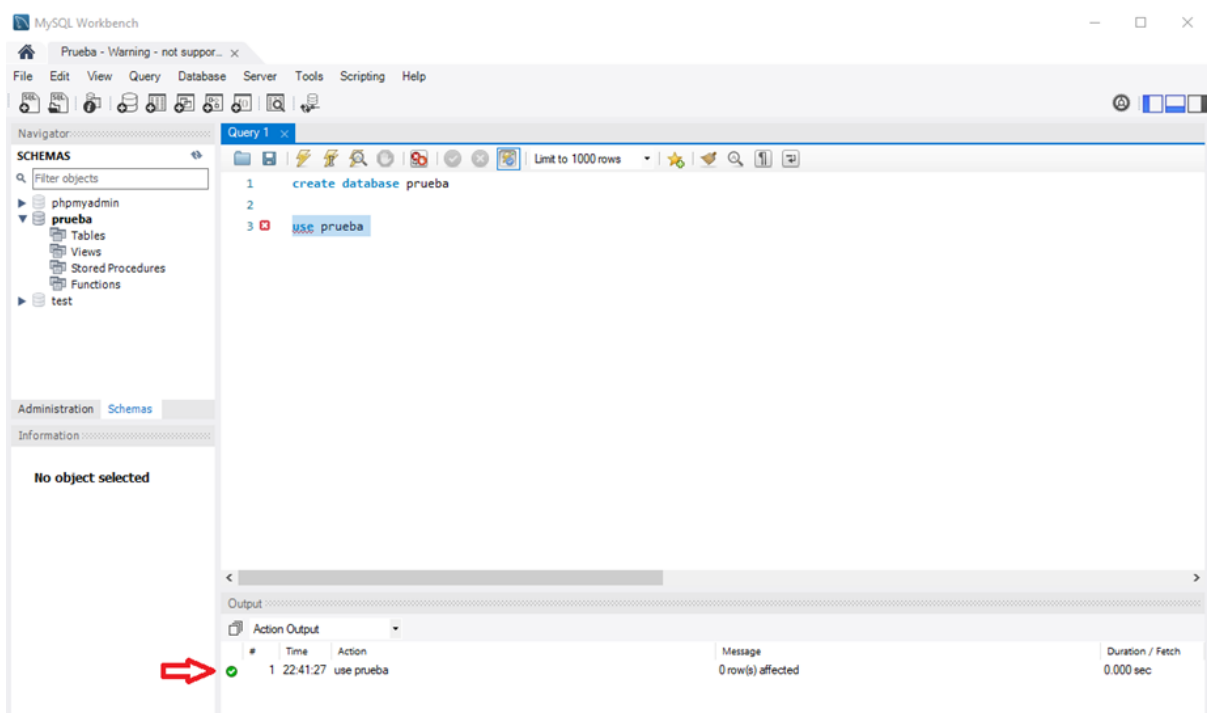
Y listo ejecutamos una instrucción que nos creó una base de datos



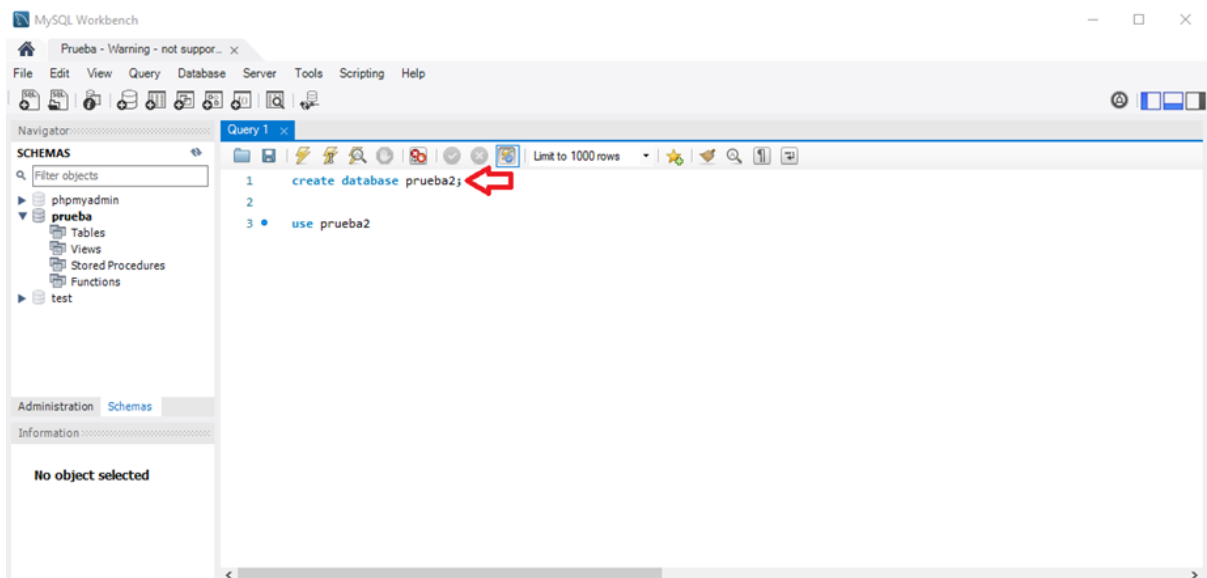
Ahora si por ejemplo tenemos más de una instrucción, es posible ejecutarlas de manera separada o en conjunto, si se desea ejecutar de manera separa lo que debemos hacer es seleccionar la instrucción que queremos ejecutar primero, por ejemplo



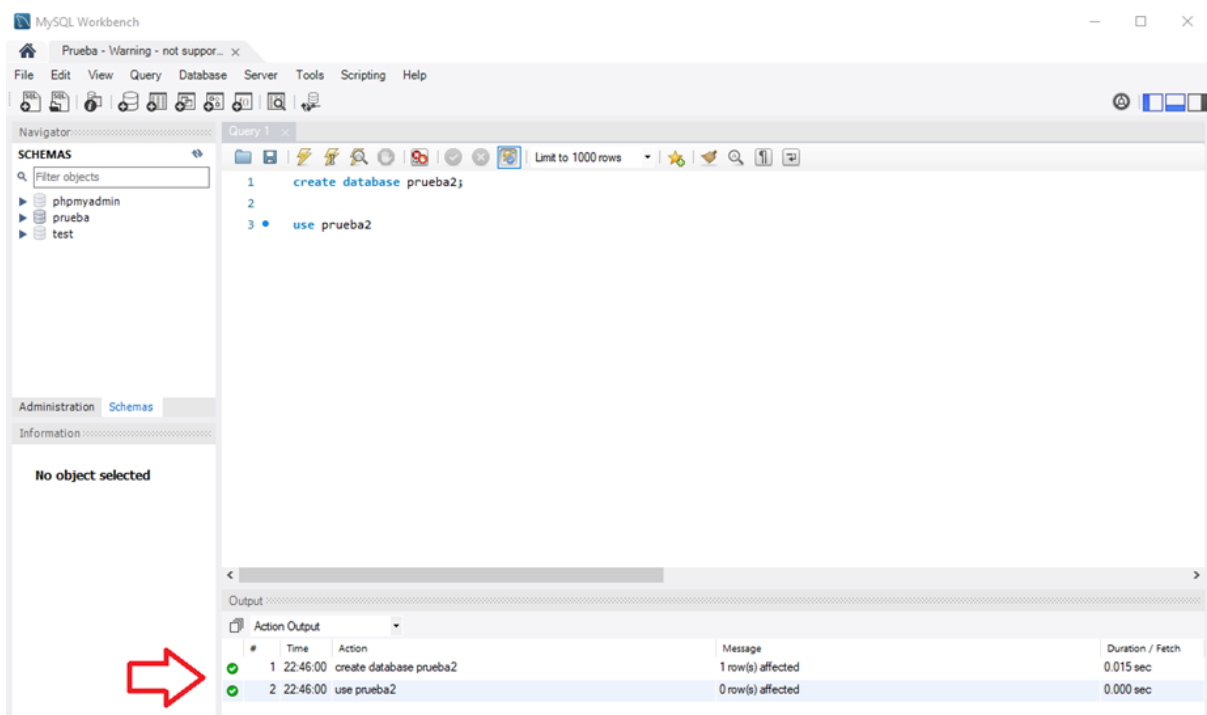
De esta manera al momento de darle clic a ejecutar solamente se ejecutará la instrucción subrayada



Si se quiere ejecutar varias instrucciones al mismo tiempo se debe poner un punto y coma al final de cada instrucción de esta manera



Así nada mas damos clic a ejecutar y se ejecutarán las dos instrucciones



7. Script para cargar la base de datos Northwind

-- Para ejecutar sin problemas el query se sustituyó la palabra GO por el punto y coma, en cada tabla se hizo un

-- cambio, la tabla categories tenía el campo identity el cual fue cambiado por el auto_increment de MySQL, lo mismo

-- paso para las demás tablas con identity

-- A todas las tablas se les cambió la línea CONSTRAINT "PK_Categories" PRIMARY KEY CLUSTERED ("Llave primaria de la tabla") por

-- PRIMARY KEY(Llave primaria de la tabla)

-- A la tabla employees se le cambio CONSTRAINT "CK_Birthdate" CHECK (BirthDate < getdate()) por CONSTRAINT CK_Birthdate CHECK (BirthDate < '2007-01-01')

- Los tipos de datos que se cambiaron fueron:

ntext por text

money por decimal

```
create database NorthwindMY;
```

```
use NorthwindMY;
```

```
CREATE TABLE Categories (  
  
    CategoryID int auto_increment NOT NULL,  
  
    CategoryName nvarchar (15) NOT NULL,  
  
    Description text NULL,  
  
    PRIMARY KEY(CategoryID)  
  
);
```



```

CREATE TABLE Suppliers (

    SupplierID int auto_increment NOT NULL ,

    CompanyName nvarchar (40) NOT NULL ,

    ContactName nvarchar (30) NULL ,

    ContactTitle nvarchar (30) NULL ,

    Address nvarchar (60) NULL ,

    City nvarchar (15) NULL ,

    Region nvarchar (15) NULL ,

    PostalCode nvarchar (10) NULL ,

    Country nvarchar (15) NULL ,

    Phone nvarchar (24) NULL ,

    Fax nvarchar (24) NULL ,

    HomePage text NULL,

    PRIMARY KEY(SupplierID)

);

```

```

CREATE TABLE Products (

    ProductID int auto_increment NOT NULL ,

    ProductName nvarchar (40) NOT NULL ,

    SupplierID int NULL ,

    CategoryID int NULL ,

    QuantityPerUnit nvarchar (20) NULL ,

```

```

UnitPrice decimal(7,2) NULL DEFAULT (0),

UnitsInStock smallint NULL DEFAULT (0),

UnitsOnOrder smallint NULL DEFAULT (0),

ReorderLevel smallint NULL DEFAULT (0),

Discontinued bit NOT NULL DEFAULT (0),

PRIMARY KEY(ProductID),


CONSTRAINT FK_Products_Categories FOREIGN KEY

(

    CategoryID

) REFERENCES Categories (

    CategoryID

),

CONSTRAINT FK_Products_Suppliers FOREIGN KEY

(

    SupplierID

) REFERENCES Suppliers (

    SupplierID

),

CONSTRAINT CK_Products_UnitPrice CHECK (UnitPrice >= 0),

CONSTRAINT CK_ReorderLevel CHECK (ReorderLevel >= 0),

CONSTRAINT CK_UnitsInStock CHECK (UnitsInStock >= 0),

```

```
CONSTRAINT CK_UnitsOnOrder CHECK (UnitsOnOrder >= 0)

);
```

```
CREATE TABLE Employees (

    EmployeeID int auto_increment NOT NULL ,

    LastName nvarchar (20) NOT NULL ,

    FirstName nvarchar (10) NOT NULL ,

    Title nvarchar (30) NULL ,

    TitleOfCourtesy nvarchar (25) NULL ,

    BirthDate datetime NULL ,

    HireDate datetime NULL ,

    Address nvarchar (60) NULL ,

    City nvarchar (15) NULL ,

    Region nvarchar (15) NULL ,

    PostalCode nvarchar (10) NULL ,

    Country nvarchar (15) NULL ,

    HomePhone nvarchar (24) NULL ,

    Extension nvarchar (4) NULL ,

    Notes text NULL ,

    ReportsTo int NULL ,

    PhotoPath nvarchar (255) NULL ,
```

```

PRIMARY KEY(EmployeeID),

CONSTRAINT FK_Employees_Employees FOREIGN KEY

(
    ReportsTo
) REFERENCES Employees (
    EmployeeID
),

CONSTRAINT CK_Birthdate CHECK ( BirthDate < '2007-01-01' )

);

```

```

CREATE TABLE Customers (

    CustomerID nchar (5) NOT NULL ,

    CompanyName nvarchar (40) NOT NULL ,

    ContactName nvarchar (30) NULL ,

    ContactTitle nvarchar (30) NULL ,

    Address nvarchar (60) NULL ,

    City nvarchar (15) NULL ,

    Region nvarchar (15) NULL ,

    PostalCode nvarchar (10) NULL ,

    Country nvarchar (15) NULL ,

    Phone nvarchar (24) NULL ,

```

```

        Fax nvarchar (24) NULL,

PRIMARY KEY(CustomerID)

);

CREATE TABLE Shippers (

        ShipperID int auto_increment NOT NULL ,

        CompanyName nvarchar (40) NOT NULL ,

        Phone nvarchar (24) NULL,

PRIMARY KEY(ShipperID)

);

CREATE TABLE Orders (

        OrderID int auto_increment NOT NULL ,

        CustomerID nchar (5) NULL ,

        EmployeeID int NULL ,

        OrderDate datetime NULL ,

        RequiredDate datetime NULL ,

        ShippedDate datetime NULL ,

        ShipVia int NULL ,

        Freight decimal(7,2) NULL DEFAULT (0),

        ShipName nvarchar (40) NULL ,

        ShipAddress nvarchar (60) NULL ,

```

```

ShipCity nvarchar (15) NULL ,

ShipRegion nvarchar (15) NULL ,

ShipPostalCode nvarchar (10) NULL ,

ShipCountry nvarchar (15) NULL,

PRIMARY KEY (OrderID),


CONSTRAINT FK_Orders_Customers FOREIGN KEY

(

    CustomerID

) REFERENCES Customers (

    CustomerID

),

CONSTRAINT FK_Orders_Employees FOREIGN KEY

(

    EmployeeID

) REFERENCES Employees (

    EmployeeID

),

CONSTRAINT FK_Orders_Shippers FOREIGN KEY

(

    ShipVia

) REFERENCES Shippers (

```

```

        ShipperID

    )

);

CREATE TABLE OrderDetails (

    OrderID int NOT NULL ,

    ProductID int NOT NULL,

    UnitPrice decimal(7,2) NOT NULL DEFAULT (0),

    Quantity smallint NOT NULL DEFAULT (1),

    Discount real NOT NULL DEFAULT (0),

    PRIMARY KEY(OrderID,ProductID),

    CONSTRAINT FK_Order_Details_Orders FOREIGN KEY

    (

        OrderID

    ) REFERENCES Orders (

        OrderID

    ),

    CONSTRAINT FK_Order_Details_Products FOREIGN KEY

    (

        ProductID

```

```

        ) REFERENCES Products (

            ProductID

        ),

        CONSTRAINT CK_Discount CHECK (Discount >= 0 and (Discount <= 1)),

        CONSTRAINT CK_Quantity CHECK (Quantity > 0),

        CONSTRAINT CK_UnitPrice CHECK (UnitPrice >= 0)

    );

/* The following adds stored procedures */

-- A estas tablas se le quitaron los [] y el dbo., tambien se eliminaron
los GO y se cambiaron por el punto y coma

CREATE TABLE Region

    ( RegionID int NOT NULL ,

        RegionDescription nchar (50) NOT NULL

    );

CREATE TABLE Territories

    (TerritoryID nvarchar (20) NOT NULL ,

        TerritoryDescription nchar (50) NOT NULL ,

        RegionID int NOT NULL

    );

```



```

CREATE TABLE EmployeeTerritories

    (EmployeeID int NOT NULL,

    TerritoryID nvarchar (20) NOT NULL

);

ALTER TABLE Region

    ADD CONSTRAINT PK_Region PRIMARY KEY

    (

        RegionID

    );

ALTER TABLE Territories

    ADD CONSTRAINT PK_Territories PRIMARY KEY

    (

        TerritoryID

    );

ALTER TABLE Territories

    ADD CONSTRAINT FK_Territories_Region FOREIGN KEY

    (

        RegionID

    ) REFERENCES Region (

```

```
        RegionID
    );
```

```
ALTER TABLE EmployeeTerritories
```

```
ADD CONSTRAINT PK_EmployeeTerritories PRIMARY KEY
(
    EmployeeID,
    TerritoryID
);
```

```
ALTER TABLE EmployeeTerritories
```

```
ADD CONSTRAINT FK_EmployeeTerritories_Employees FOREIGN KEY
(
    EmployeeID
) REFERENCES Employees (
    EmployeeID
);
```

```
ALTER TABLE EmployeeTerritories
```

```
ADD CONSTRAINT FK_EmployeeTerritories_Territories FOREIGN KEY
(
    TerritoryID
```

```

        ) REFERENCES Territories (

            TerritoryID

        );

-- The following adds constraints to the Northwindm database

CREATE TABLE CustomerCustomerDemo

    (CustomerID nchar (5) NOT NULL,

    CustomerTypeID nchar (10) NOT NULL

);

CREATE TABLE CustomerDemographics

    (CustomerTypeID nchar (10) NOT NULL ,

    CustomerDesc text NULL

);

ALTER TABLE CustomerCustomerDemo

    ADD CONSTRAINT PK_CustomerCustomerDemo PRIMARY KEY

    (

        CustomerID,

        CustomerTypeID

    );

```

```
ALTER TABLE CustomerDemographics
```

```
ADD CONSTRAINT PK_CustomerDemographics PRIMARY KEY
```

```
(  
  
    CustomerTypeID  
  
);
```

```
ALTER TABLE CustomerCustomerDemo
```

```
ADD CONSTRAINT FK_CustomerCustomerDemo FOREIGN KEY
```

```
(  
  
    CustomerTypeID  
  
) REFERENCES CustomerDemographics (  
  
    CustomerTypeID  
  
);
```

```
ALTER TABLE CustomerCustomerDemo
```

```
ADD CONSTRAINT FK_CustomerCustomerDemo_Customers FOREIGN KEY
```

```
(  
  
    CustomerID  
  
) REFERENCES Customers (  
  
    CustomerID  
  
);
```

8. Script para la familia de vistas de la base de datos Northwind

```
use northwindmy;
```

```
CREATE VIEW vw_Products AS
```

```
SELECT p.ProductID, p.ProductName, p.QuantityPerUnit, p.UnitPrice AS ProductUnitPrice,
```

```
p.UnitsInStock, p.UnitsOnOrder, p.ReorderLevel, p.Discontinued,
```

```
s.SupplierID, s.CompanyName, s.ContactName AS ContactName, s.ContactTitle, s.Address,
```

```
s.City, s.Region, s.PostalCode, s.Country, s.Phone, s.Fax, s.HomePage,
```

```
c.CategoryID, c.CategoryName, c.Description
```

```
FROM Products p
```

```
INNER JOIN Suppliers s ON p.SupplierID = s.SupplierID
```

```
INNER JOIN Categories c ON p.CategoryID = c.CategoryID;
```

```
CREATE VIEW vw_Orders AS
```

```
SELECT
```

```
o.OrderID, o.OrderDate, o.RequiredDate, o.ShippedDate, o.Freight, o.ShipName,
```

```
o.ShipAddress, o.ShipCity, o.ShipRegion, o.ShipPostalCode, o.ShipCountry,
```

```
s.ShipperID, s.CompanyName AS ShipCompanyName, s.Phone AS ShipPhone,
```

```
c.CustomerID, c.CompanyName AS CusCompanyName, c.ContactName AS CusContactName, c.ContactTitle AS CusContactTitle,
```

c.Address AS CusAddress, c.City AS CusCity,

c.Region AS CusRegion, c.PostalCode AS CusPostalCode, c.Country AS CusCountry, c.Phone AS CusPhone, c.Fax AS CusFax,

e.EmployeeID, e.LastName, e.FirstName, e.Title, e.TitleOfCourtesy, e.BirthDate,

e.HireDate, e.Address AS EmpAddress, e.City AS EmpCity, e.Region AS EmpRegion, e.PostalCode AS EmpPostalCode, e.Country AS EmpCountry, e.HomePhone, e.Extension,

e.Notes, e.ReportsTo, e.PhotoPath

FROM Orders o

INNER JOIN Employees e ON o.EmployeeID = e.EmployeeID

INNER JOIN Shippers s ON o.ShipVia = s.ShipperID

INNER JOIN Customers c ON o.CustomerID = c.CustomerID;

CREATE VIEW vw_OrderDetails AS

SELECT

x.Quantity, x.Discount, x.UnitPrice, p. 'o.'

FROM OrderDetails x

INNER JOIN vw_Orders o ON o.OrderID = x.OrderID

INNER JOIN vw_Products p ON x.ProductID = p.ProductID;

9. Script con procedimiento almacenado para insertar y modificar la tabla Territories

```
-- Procedimiento almacenado que inserta y modifica la tabla territories
--

delimiter $$

create procedure sp_Territorios(inout terrid int, in terrDescription
nchar(50), in tRegion int)

begin

    if exists(select*from territories where territoryid = terrid)

    then

        update territories set territoryid = terrid,
territorydescription = terrDescription, regionid = tRegion where
territoryid = terrid;

    else

        select terrid = max(territoryid)+1 from territories;

        insert into territories(territoryid, territorydescription,
regionid) values (terrId, terrDescription, tRegion);

    end if;
end $$
```

10. Script con un trigger que no permita eliminar la tabla Territories

```
use northwindmy;
```

```
DELIMITER //
```

```
CREATE TRIGGER tr_Territories
```

```
BEFORE DELETE ON territories
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    SIGNAL sqlstate '45000' SET message_text = 'No se pueden eliminar  
territorios';
```

```
END//
```