**Clase 02 - Introducción a las Bases de datos Relacionales**

Concepto Base de Datos Relacional

Las bases de datos relacionales se basan en el Modelo Relacional usandotablas para representar los datos y las relaciones entre ellos.

Base de datos plana

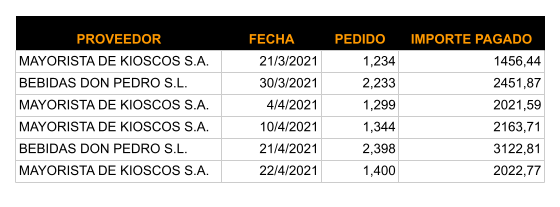
Es una DB con una sola tabla; es normal encontrar pequeños comercios que usan a las planillas de cálculo como un sistema de almacenamiento de datos.

Los datos alojados en las diferentes hojas de datos no suelen tener relación entre sí.

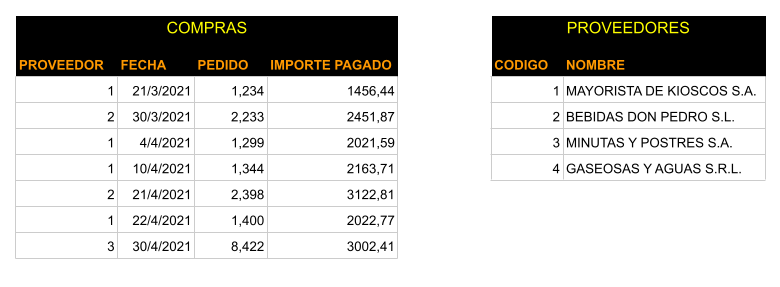
En los casos donde se da esta situación, el contenido de la información sufre de lo que se denomina ***defecto grave en términos de eficiencia***. Esto ocurre porque sus datos se tornan repetitivos; donde por ejemplo: se repite el nombre de un proveedor en cada una de las celdas donde debe figurar la denominación social de éste.

Con el tiempo y, sobre todo, en los casos donde la información que se carga es constante; una base de datos plana crecerá de forma significativa por la repetición de caracteres constante.

La base de datos se tornará lenta cuando consultemos la información almacenada y ésta tenga miles o decenas de miles de registros con información repetida.

Ejemplo DB Plana

Ahora bien, convertir la DB plana en una DB relacional nos lleva a tener que plantear más de una tabla. Separar por un lado los datos del PROVEEDOR de los datos de cada COMPRA eso evitará todos los problemas que trae la redundancia, o la pérdida de información cuando se borre el último registro de un proveedor, o que no se pueda registrar información de un proveedor que aún no ha realizado compras.

Ejemplo normalización de datos. Pasamos de una DB plana a una BD relacional.

Ventajas DB Relacional

Evitar datos duplicados

Optimiza el espacio de almacenamiento

Reduce la complejidad del uso de las tablas

Modelo Relacional

Concepto

El Modelo relacional en el cual se apoyan las **Bases de Datos Relacionales** almacenan la información en un **conjunto de tablas**, y a su vez, las aprovechan para representar tanto los ***datos*** como también las ***relaciones*** entre cada una de ellas

# Tabla

Tipo de elemento que permite guardar la información en registros y cumple con el paradigma de modelo relacional.

Cada tabla creada debe tener un nombre único en cada DB, haciéndola accesible mediante su nombre.Además si atomizamos cada uno de los datos o registros guardados en una tabla a un elemento independiente del resto, debemos tener presente que en cada columna de la tabla todos los campos son del mismo tipo, es una característica que define al modelo de datos relacional.

Las tablas en las DB se componen de dos estructuras:

* **FILAS**: la primera fila de la tabla corresponde al nombre de los campos que la integran y se denomina **esquema de tabla o cabecera**. El resto de las filas de la tabla son los registros de datos
* **COLUMNAS**: Corresponde al nombre del campo y a todos los datos que almacena para cada registro en dicha posición.

.

IMPORTANTE: En la definición de cada campo, debe existir un nombre único, con su tipo de dato correspondiente. Esto es útil a la hora de manejar varios campos en la tabla, ya que cada nombre de campo debe ser distinto entre sí.

A los campos se les puede asignar, además, propiedades especiales que afectan a los registros insertados, propiedades del estilo que deben tomar valores únicos en dicha columna, o que pueden ser campos opcionales (es decir que puede contener valores nulos), etc.

**Registro**

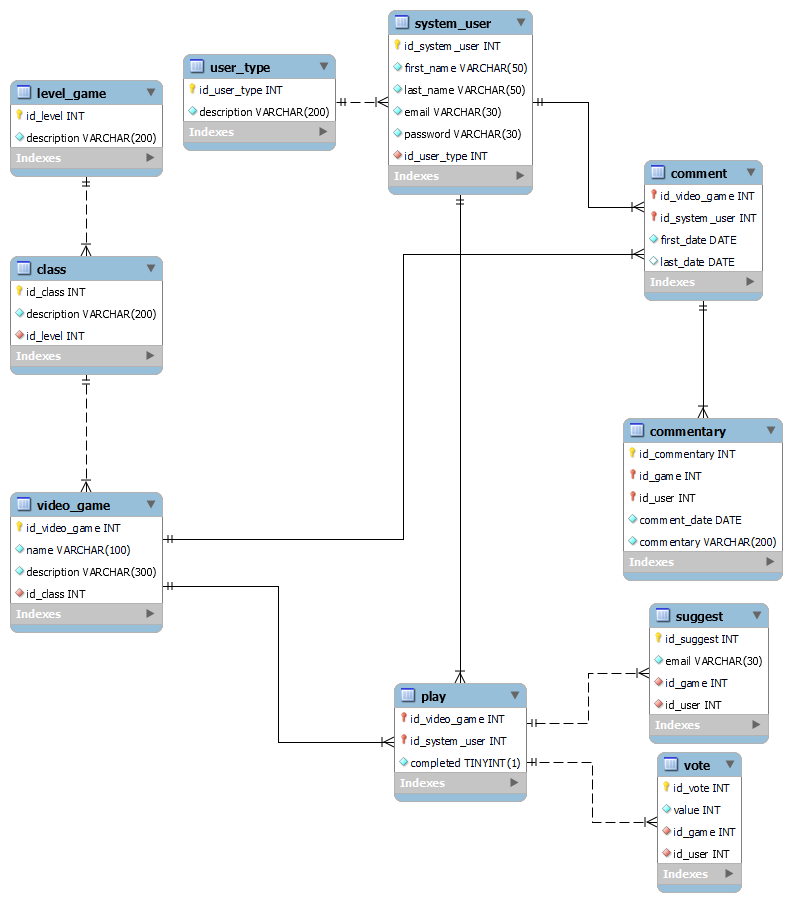
El registro o tupla, representa un objeto único de datos, implícitamente estructurado dentro de una tabla. Cada nuevo dato que vamos agregando en la tabla, conforma un nuevo registro y el conjunto de estos, termina conformando el total de datos que la tabla contiene.

Toda esta información es consistente de forma individual y, en determinados casos, también de forma cruzada.

Uno de los requisitos del modelo relacional es que en una tabla, no debiera haber registros duplicados. Los datos deben ser diferentes en, al menos, uno de los campos de la fila o registro para cumplir con una de las reglas del modelo relacional.

**Estructura de tablas**

Es una **estructura de tablas**, las cuales a su vez **se relacionan con otras** tablas.

Su principal característica es **no poseer información repetida** de forma innecesaria, lo que permite adicionar más información sin llegar a afectar la otra almacenada.

## 

## Ventajas Modelo relacional

Favorece el proceso de **normalización**, el cual permite eliminar la redundancia **evitando la duplicidad** de información en los registros guardados.

Permite realizar **consultas** y obtener **reportes** de **forma ágil y rápida** por medio de SQL u otro lenguaje de base de datos estructurado.

Se pueden crear **una o varias** relaciones entre las tablas.

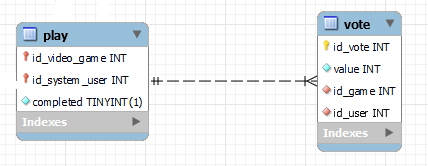
Garantiza la **integridad referencial;** si un **registro se relaciona con otro registro de otra tabla**, **no permite** **que** el mismo **sea eliminado**.

Asimismo, si se quiere borrar, también pasará con todos los datos relacionados

# Tipos de relaciones

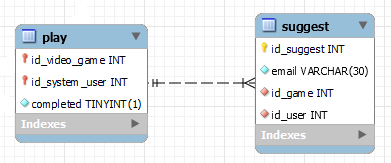
**Uno a Uno**

Cada registro, en cada tabla, aparece solo una vez, y tienen una relación unívoca.

En el ejemplo se muestra que para cada juego que juega un usuario puede realizar su voto una única vez.

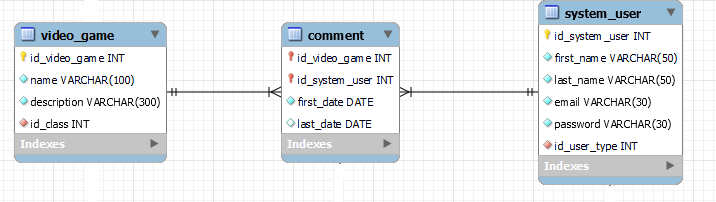
**Uno a Muchos**

Un registro en una tabla puede tener relación con varios elementos de otra tabla.

En el ejemplo que se presenta se observa que un usuario de un sistema de juegos en línea es de un tipo (USER\_TYPE) y a su vez pueden haber varios usuarios del mismo tipo.

**Muchos a Muchos**

Uno o más registros en una tabla pueden tener relación con uno o más elementos de otra tabla.



# Claves en las tablas

Lógicas

*PK* **Clave primaria PK**: también llamada llave primaria o primary key, hace que el **registro** sea **unívoco** y obligatoriamente **no nulo.**

Una clave primaria comprende una [columna](https://es.wikipedia.org/wiki/Columna_(base_de_datos)) o conjunto de columnas. No puede haber dos filas en una tabla que tengan la misma clave primaria.

Ejemplos de claves primarias: [DNI](https://es.wikipedia.org/wiki/Documento_de_identidad) (asociado a una [persona](https://es.wikipedia.org/wiki/Persona)) o [ISBN](https://es.wikipedia.org/wiki/ISBN) (asociado a un [libro](https://es.wikipedia.org/wiki/Libro)).

*FK* **Clave foránea FK**: también llamada foreign key, clave secundaria o clave externa, puede ser -o no- una clave primaria dentro de la tabla. Su característica es que es el punto de enlace con otra tabla donde ésta es primary key.

La clave foránea identifica una [columna](https://es.wikipedia.org/wiki/Columna_(base_de_datos)) o grupo de columnas en una [tabla](https://es.wikipedia.org/wiki/Tabla_(base_de_datos)) (*tabla hija o referendo*) que se refiere a una columna o grupo de columnas en otra tabla (*tabla maestra o referenciada*). Las columnas en la tabla referendo deben ser la [clave primaria](https://es.wikipedia.org/wiki/Clave_primaria) u otra clave candidata en la tabla referenciada.

Los valores en una fila de las columnas referendo deben existir solo en una fila en la tabla referenciada. Así, una fila en la tabla referendo no puede contener valores que no existen en la tabla referenciada. De esta forma, las referencias pueden ser creadas para vincular o relacionar información. Esto es una parte esencial de la normalización de base de datos. Múltiples filas en la tabla referendo pueden hacer referencia, vincularse o relacionarse a la misma fila en la tabla referenciada. Mayormente esto se ve reflejado en una relación uno (*tabla maestra o referenciada*) a muchos (*tabla hija o referendo*).

*INDEX* **Clave índice:** es un campo que **facilita la búsqueda** dentro de una tabla. Generalmente son campos primary key.

Se los define como una [estructura de datos](https://es.wikipedia.org/wiki/Estructura_de_datos) que mejora la velocidad de las operaciones, por medio de un identificador único de cada fila de una tabla, permitiendo un rápido acceso a los [registros](https://es.wikipedia.org/wiki/Registro_(base_de_datos)) de una [tabla](https://es.wikipedia.org/wiki/Tabla_(base_de_datos)) en una base de datos.

El índice tiene un funcionamiento similar al [índice](https://es.wikipedia.org/wiki/%C3%8Dndice) de un [libro](https://es.wikipedia.org/wiki/Libro), guardando parejas de elementos: el elemento que se desea indexar y su posición en la base de datos. Para buscar un elemento que esté indexado, sólo hay que buscar en el índice dicho elemento para, una vez encontrado, devolver un registro que se encuentre en la posición marcada por el índice.

Conceptuales

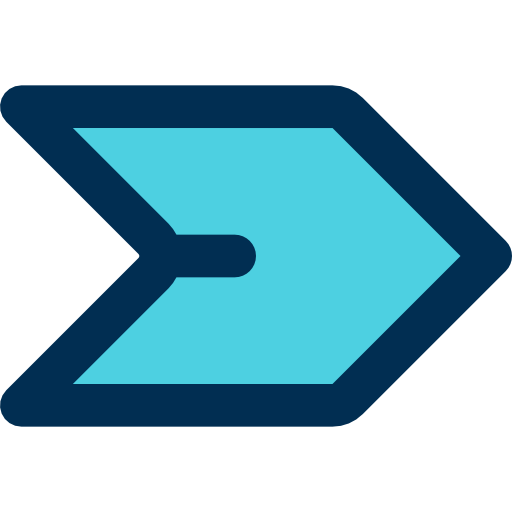
**Clave concatenada CK**: ayuda a encontrar la singularidad en una tabla **combinando dos campos**, ya que no hay una llave primaria.

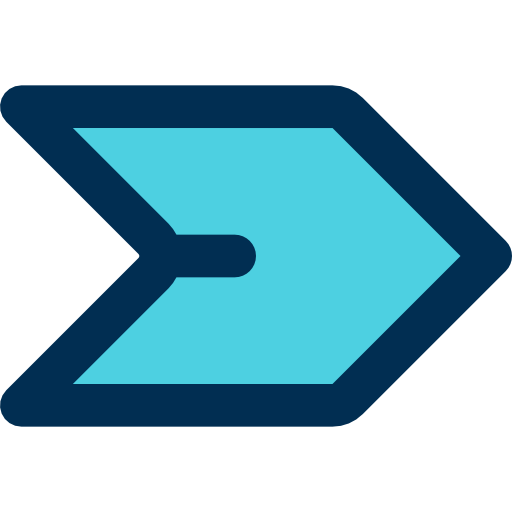
**Claves candidatas**: cuando existe más de una clave primaria dentro de la tabla, por ejemplo legajo y dni.

Modelo Entidad-Relación

# Modelo de datos Entidad-Relación

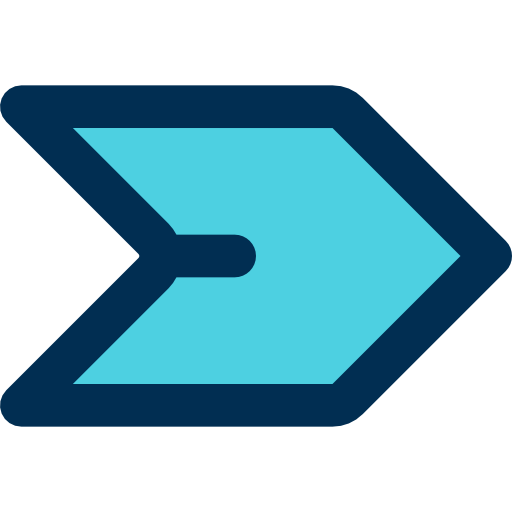
#### ¿De qué se trata?

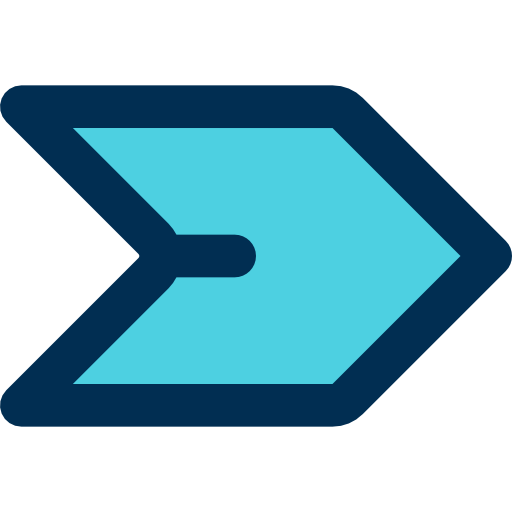
El modelo de datos entidad-relación (E-R) se basa en la percepción del mundo real que consiste en un **conjunto de objetos básicos**, denominados **entidades**, y de las **relaciones entre esos objetos**.

Cuando hablamos de **Entidad**, nos referimos puntualmente a un objeto o concepto del mundo real que se diferencia de otro objeto, aún cuando ambos sean del mismo tipo, o de una misma entidad. Un ejemplo de esto serían las personas. En sí, una persona es única por su ADN. No se parece a ninguna otra persona incluso en los casos donde encontramos gemelos o trillizos. Cada una de estas personas seguirá teniendo su propio ADN que lo identifica como individuo.

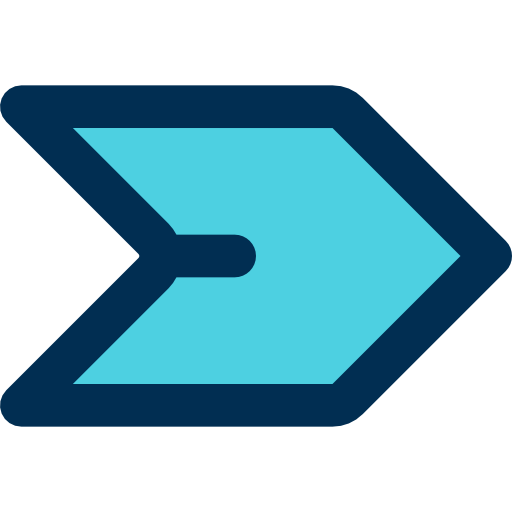
Una casa puede ser otro claro ejemplo. Aún cuando veamos un barrio que fue construido con casas exactamente iguales, con la misma distribución y elementos que la componen, cada una de las casas será diferente a la otra simplemente porque su dirección difiere del resto.

La relación entre estas entidades es que son del mismo tipo, a su vez, pero cada una de ellas tendrá una diferenciación del resto.

Al llevar este modelo al terreno de las bases de datos comenzamos a ver que cada una de las tablas posee **registros** (*entidades*), y que éstos se diferencian del resto de sus pares por alguna característica particular.

Los registros están relacionados entre sí dentro de una misma tabla, porque poseen una característica que los hace iguales. Por ejemplo, la tabla de **PROVEEDORES**; la misma tendrá la información de todas las entidades del tipo proveedor, que nos proveen con productos o servicios. Pero, a su vez, cada proveedor es unívoco por su identificador, nombre o razón social, domicilio, teléfonos de contacto, y el tipo de servicio o producto que éste comercializa.

De todos los datos que conforman a cada entidad proveedor, habrá al menos uno que diferencia a éste, del resto de los proveedores. Más allá de esto, los proveedores estarán relacionados entre sí porque comparten una misma entidad o categoría (*la de proveedor*).

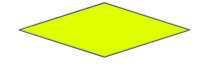
En el ecosistema de DB cada componente posee una forma de identificarse del resto. Además, la forma en la que se relacionan, se basa en un modelo claro que permite la comprensión de la interacción entre cada parte.

# Componentes de base de datos

Estos componentes se ocupan de representar cada una de las figuras que nos permitirán definir un diagrama Entidad-Relación de la forma más clara posible; como paso previo a la construcción de una base de datos con cada **Entidad** clara y asociada con el resto de las entidades por uno o más puntos en común, los cuales llamamos **Relación**.

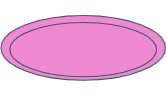
• **Rectángulos**: representan conjuntos de entidades.

• **Elipses**: representan atributos, en la mayoría de los casos, la clave candidata.

• **Rombos**: representan acciones que permiten relaciones entre tablas.

• **Líneas**: unen atributos a conjuntos de entidades y conjuntos de entidades a conjuntos de relaciones. Representan la unión entre acciones y entidades 

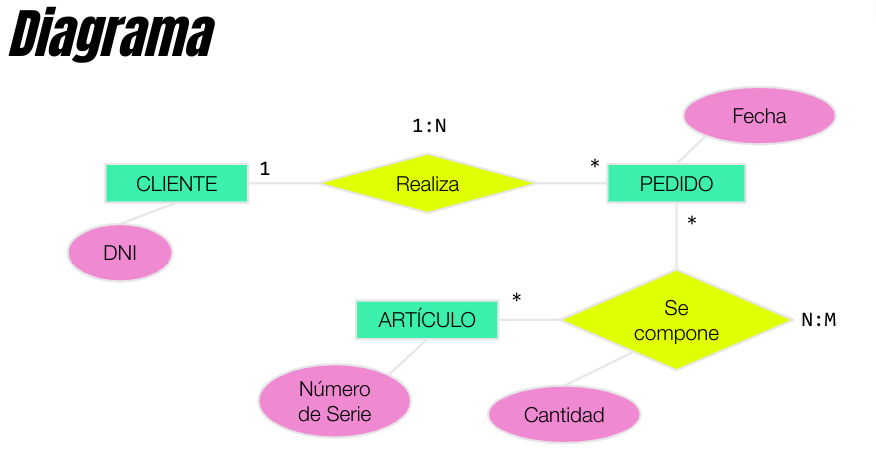
Además existen otras combinaciones más complejas, que utilizan su simbología. Las dejamos a continuación y serán abordadas más adelante en este mismo curso.

• **Elipses dobles**: representan atributos multivalorados. 

• **Elipses discontinuas**: que denotan atributos derivados.

• **Líneas dobles**: indican la participación total de una entidad en un conjunto de relaciones.

• **Rectángulos dobles**: representan conjuntos de entidades débiles.

Ejemplos diagrama

MySQL

## Un poco de historia

Es un **sistema de gestión de Base de datos Relacional**.

**ORACLE** es su dueño actual, pero mantiene una **licencia dual**, que le permite tener una opción **bajo el estándar OPEN SOURCE** (*Código Abierto*).

En el mundo open source, es la base de datos más popular entre sus pares.

MySQL fue inicialmente desarrollada por [MySQL AB](https://es.wikipedia.org/wiki/MySQL_AB) (*empresa fundada por* [*David Axmark*](https://es.wikipedia.org/w/index.php?title=David_Axmark&action=edit&redlink=1)*,* [*Allan Larsson*](https://es.wikipedia.org/w/index.php?title=Allan_Larsson&action=edit&redlink=1) *y* [*Michael Widenius*](https://es.wikipedia.org/wiki/Michael_Widenius)). MySQL AB fue adquirida por [Sun Microsystems](https://es.wikipedia.org/wiki/Sun_Microsystems) en 2008 y ésta, a su vez, fue comprada por [Oracle Corporation](https://es.wikipedia.org/wiki/Oracle_Corporation) en 2010, la cual ya era dueña desde 2005 de [Innobase Oy](https://es.wikipedia.org/w/index.php?title=Innobase_Oy&action=edit&redlink=1), empresa [finlandesa](https://es.wikipedia.org/wiki/Finlandia) desarrolladora del motor [InnoDB](https://es.wikipedia.org/wiki/InnoDB) para MySQL.

Al contrario de proyectos como [Apache](https://es.wikipedia.org/wiki/Servidor_HTTP_Apache), donde el software es desarrollado por una comunidad pública y los [derechos de autor](https://es.wikipedia.org/wiki/Derechos_de_autor) del código están en poder del autor individual, MySQL es patrocinado por una [empresa privada](https://es.wikipedia.org/wiki/Empresa_privada), que posee el copyright de la mayor parte del código.

Esto es lo que posibilita el esquema de doble licenciamiento anteriormente mencionado. La base de datos se distribuye en varias versiones, una *Community*, distribuida bajo la Licencia pública general de [GNU](https://es.wikipedia.org/wiki/GNU), versión 2, y varias versiones *Enterprise*, para aquellas empresas que quieran incorporarlo en productos privativos. Las versiones *Enterprise* incluyen productos o servicios adicionales tales como herramientas de [monitorización](https://es.wikipedia.org/w/index.php?title=Monitorizaci%C3%B3n_de_sistemas&action=edit&redlink=1) y [asistencia técnica](https://es.wikipedia.org/wiki/Soporte_t%C3%A9cnico) oficial.

En [2009](https://es.wikipedia.org/wiki/2009) se creó un [*fork*](https://es.wikipedia.org/wiki/Bifurcaci%C3%B3n_(desarrollo_de_software)) denominado [MariaDB](https://es.wikipedia.org/wiki/MariaDB) por algunos desarrolladores (*incluido algunos desarrolladores originales de MySQL*) descontentos con el modelo de desarrollo y el hecho de que una misma empresa controle a la vez los productos MySQL y [Oracle Database](https://es.wikipedia.org/wiki/Oracle_Database).

Fuente: *Wikipedia* (<https://es.wikipedia.org/wiki/MySQL>)

## Modelo cliente-servidor

Fue concebida para usarse bajo el **modelo cliente-servidor**. Debe instalarse en un servidor para que los clientes se conecten y operen con la información de la base de datos.

Soporta múltiples bases de datos bajo un único motor y, para una respuesta efectiva, dependerá del hardware de procesamiento del servidor donde se ejecute.

El servidor MySQL y los clientes soportan los principales sistemas operativos existentes, entre los cuales se encuentran los siguientes:

* La familia de sistemas operativos Windows (Windows 95, 98, Me, NT, 2000 y XP). MySQL no soporta MS-DOS o Windows 3.1
* La familia UNIX y derivados entre los cuales se encuentran: Los sistemas BSD (ejemplo: FreeBSD, OpenBSD, NetBSD, etc.), el sistema operativo MacOS X, System V, Solaris, HP-UX, entre otros, la familia Linux (como Fedora, RedHat, SuSE, Debian, Mandrake, Gentoo, Ubuntu, etc.)
* y Novell Netware 6.5 y superior.

Debido a que MySQL trabaja en un ambiente de Red (*es decir entre computadores conectados en Red o al internet*), el servidor MySQL y los clientes pueden inter-operar en diferentes sistemas operativos. Por ejemplo, podemos tener instalado el servidor MySQL en un sistema operativo tipo Unix (*como Linux*) y podemos acceder a la información a través de la red con un ordenador con sistema operativo tipo Windows a través de las aplicaciones clientes.

Fuente: *Wikipedia* (<https://es.wikipedia.org/wiki/MySQL>)

# Componentes Mysql

¡ATENCIÓN! El uso de comandos está más relacionado a ambientes como ser servidores, a los cuales accedes remotamente por clientes como SSH o similares, y que no cuentan con una interfaz gráfica de administración de DB para ahorrar recursos de procesamiento, entre otros casos.

**mysqld**: es el corazón de MySQL. Es el programa servidor el cual proporciona las bases de datos que se encuentran en memoria o en el disco duro. En los sistemas operativos Windows NT, 2000 o XP el servidor MySQL son los programas mysqld-nt o mysql-max-nt

**mysql:** esta interfaz de texto permite también controlar todos los aspectos del servidor pero la interfaz es solamente texto. Permite leer instrucciones del usuario por medio del teclado como también puede leer archivos que contienen instrucciones. Ésta es la aplicación más usada por los programadores para controlar la base de datos. Ésta y otras aplicaciones que veremos a continuación son aplicaciones de texto que funcionan en la consola. (command prompt o cmd en Windows y un shell sh, bash, etc. en Unix, Linux, MacOS y demás)

**mysqlimport:** permite importar datos a través de archivos de texto. Provee una interfaz de texto para los comandos LOAD, DATA, INFILE

**mysqldump:** permite hacer copias o respaldos de la información almacenada para restaurarlos en el mismo servidor o para exportarlo a otros servidores

**mysqladmin:** permite administrar el servidor con una interfaz gráfica y de una forma muy sencilla

**mysqlcheck:** permite revisar la salud de la base de datos. Permite también reparar dichas bases si fuera necesario

**myisamchk:** revisa las tablas y realiza operaciones de reparación

**myisampack:** crea versiones comprimidas de sólo-lectura de tablas MyISAM

Todos estos componentes se delinean como una aplicación de línea de comandos o Terminal. Tienen una serie de parámetros asociados, denominados también como opciones, que se combinan con otros comandos en la misma línea de ejecución. Por ejemplo, en la línea de comandos escribes:

mysql (ENTER)

E inmediatamente te solicitará el password para conectarte al motor de base de datos Mysql instalado en la computadora. Ya identificado con la clave, el prompt de la línea de comandos o Terminal cambiará por:

mysql>

A partir de ahora, ya puedes escribir otros comandos, como ser:

show databases; (ENTER)

Te invitamos a ver más ejemplos como éste, en el siguiente link: [https://blog.devart.com/how-to-connect-to-mysql-server.html](https://blog.devart.com/how-to-connect-to-mysql-server.html?gclid=EAIaIQobChMI7daGluW_8QIVbh-tBh1rlQvlEAAYAiAAEgL0QfD_BwE)

MySQL Workbench

MySQL Workbench es una herramienta visual unificada; proporciona modelado de datos, desarrollo SQL y herramientas de administración integrales para la configuración del servidor, administración de usuarios, respaldo y mucho más.

Este software integra:

**Diseño:** MySQL Workbench permite a un DBA, desarrollador o arquitecto de datos diseñar, modelar, generar y administrar bases de datos visualmente.

Incluye todo lo que un modelador de datos necesita para crear modelos ER complejos, ingeniería directa e inversa, y también ofrece características clave para realizar tareas de documentación y administración de cambios difíciles que normalmente requieren mucho tiempo y esfuerzo. <https://www.mysql.com/products/workbench/design/>

**Desarrollo:** MySQL Workbench ofrece herramientas visuales para crear, ejecutar y optimizar consultas SQL. El Editor de SQL proporciona resaltado de sintaxis de color, autocompletado, reutilización de fragmentos de código SQL e historial de ejecución de SQL. El Panel de conexiones de base de datos permite a los desarrolladores administrar fácilmente las conexiones de base de datos estándar, incluido MySQL Fabric. El Explorador de objetos proporciona acceso instantáneo al esquema y los objetos de la base de datos. <https://www.mysql.com/products/workbench/dev/>

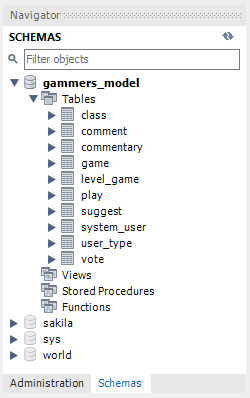
**Administración:** MySQL Workbench proporciona una consola visual para administrar fácilmente los entornos MySQL y obtener una mejor visibilidad de las bases de datos. Los desarrolladores y administradores de bases de datos pueden utilizar las herramientas visuales para configurar servidores, administrar usuarios, realizar copias de seguridad y recuperación, inspeccionar datos de auditoría y ver el estado de la base de datos. <https://www.mysql.com/products/workbench/admin/>

**Mantenimiento:** MySQL Workbench proporciona un conjunto de herramientas para mejorar el rendimiento de las aplicaciones MySQL y **migración de base de datos** una solución completa y fácil de usar para migrar Microsoft SQL Server, Microsoft Access, Sybase ASE, PostgreSQL y otras tablas, objetos y datos RDBMS a MySQL.

Fuente: [**https://www.mysql.com/products/workbench/**](https://www.mysql.com/products/workbench/)

Video en Youtube: <https://www.youtube.com/watch?v=X_umYKqKaF0>

# Panel Schemas

El **Panel Schemas** nos lista todas las bases de datos que tengamos, junto a sus objetos.

Podemos navegar cada objeto a través del **menú de árbol**, para ver sus propiedades y acciones.

Al pié, obtenemos **información general del elemento** seleccionado.

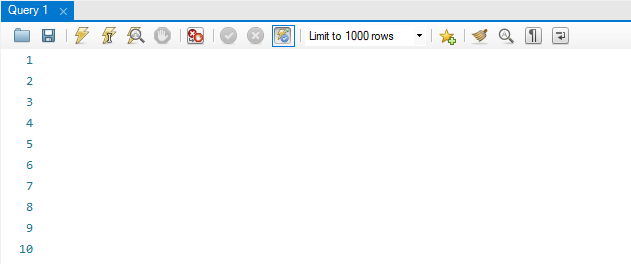
# Barra de Herramientas Superior



Incluye botones para iniciar cualquiera de las principales tareas tales como crear una base de datos, abrir una ventana de script, crear una tabla, una función, un stored procedures y otras opciones más.

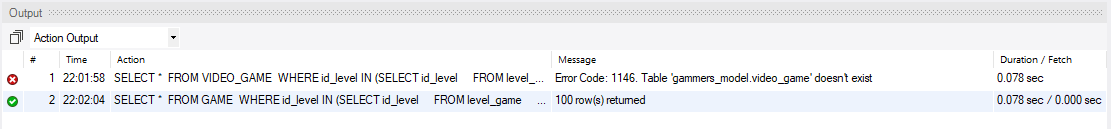
Al pié se encuentran **las pestañas de acción** para navegar entre los objetos en uso.

# Panel central



Permite visualizar diferentes pestañas para ver o escribir comandos SQL, junto a la ventana de resultados al pie de este panel

# Pestaña de Acciones



Al pié de la ventana está la **pestaña de acciones**, donde veremos el resultado de cada consulta o comando SQL que se dispare; la confirmación de ejecución, los errores, y demás mensajes del sistema.

Glosario Técnico

* **Base de datos plana:** La base de datos plana, a diferencia de la relacional, puede almacenar información general y depender de una o dos tablas no relacionadas entre sí. Usualmente este tipo de base de datos se utiliza para almacenar LOGs de una base de datos relacional más potente.
* **Modelo relacional:** Las **Bases de Datos Relacionales** almacenan la información en un **conjunto de tablas**, y a su vez, las aprovechan para representar tanto los ***datos*** como también las ***relaciones*** entre cada una de ellas
* **Diagrama Entidad-Relación:** Es un tipo de diagrama de flujo que ilustra cómo las "entidades", como personas, objetos o conceptos, se relacionan entre sí dentro de un sistema.
* **SGBD:** Siglas correspondientes a Sistema de Gestión de Base de Datos. Hacen referencia a la aplicación de software que permite la gestión total de la información de las bases de datos. Usualmente estos sistemas son aplicaciones visuales que facilitan la interacción con los diferentes objetos de una DB.
* **Mysql:** Es un **sistema de gestión de Base de datos Relacional**.
* **Mysql Workbench:** es una herramienta visual unificada.
* **Clave primaria:** Es la columna o grupo de columnas de una tabla que identifica de forma exclusiva cada fila de la tabla. Impiden que los datos de dicho registro sean del tipo Nulo y agilizan la búsqueda de la información utilizando mecanismos acordes.
* **Clave foránea:** Es una clave específica que permite referenciar a dos o más tablas de una base de datos. La misma debe estar conformada por una clave primaria o clave candidata en la tabla de referencia.
* **Índice único:** es un campo que **facilita la búsqueda** dentro de una tabla.
* **Relación uno a uno:** Se denomina relación uno a uno al vínculo de información entre dos tablas. Cada registro de cada tabla aparecerá solo una vez en éstas.
* **Relación uno a muchos:** Se denomina relación de uno a muchos cuando un registro de una tabla se asocia con uno o varios registros de otra tabla. Por lo tanto, la clave primera de una tabla se enlazará a la clave foránea de otra tabla.
* **Relación muchos a muchos:** Se denomina relación de muchos a muchos cuando uno o varios registros de una tabla se asocian con uno o varios registros de otra tabla. Para que esto pueda resolverse en SQL se deberá optar por incluir una tercera tabla que se ocupe de dicha relación.
* **Tablas:** Las tablas son el objeto en una base de datos que permite almacenar registros, teniendo como premisa una estructura definida por campos y registros, estableciendo una forma normalizada para el almacenamiento de dicha información.
* **Filas y columnas:** Las filas y columnas en una tabla de datos guardan relación directa, siendo las columnas quien define o estructura la forma relacional en la cual se almacenan los datos en dicha tabla, mientras que los campos hacen referencia al tipo de dato que se almacenará en cada posición.
* **Registros:** Es el conjunto de información que se almacena en las tablas de una base de datos. Cada registro es un cúmulo de datos homogéneo que representa a una entidad definida.