Bases de Datos

1. **¿Qué es una base de datos?**

Una base de datos o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. En este sentido; una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta.



Una base de datos es un conjunto de información organizada de manera que pueda ser utilizada eficientemente.

Un directorio telefónico, un diccionario, un calendario o un libro de recetas son ejemplos de bases de datos.

1. **Elementos básicos de una base de datos**

* Registro: la forma en la que se organizan los datos dentro de la base de datos.

Cada registro contiene toda la información sobre una persona o un elemento de la base. Por ejemplo, cada registro en el directorio telefónico contiene el nombre, dirección y número telefónico de una persona.

* Campos: describe información particular, que en conjunto con otros campos conforman un registro.

Cada campo tiene un tipo que identifica la clase de información que puede almacenar: números, fechas, caracteres alfanuméricos y otros.

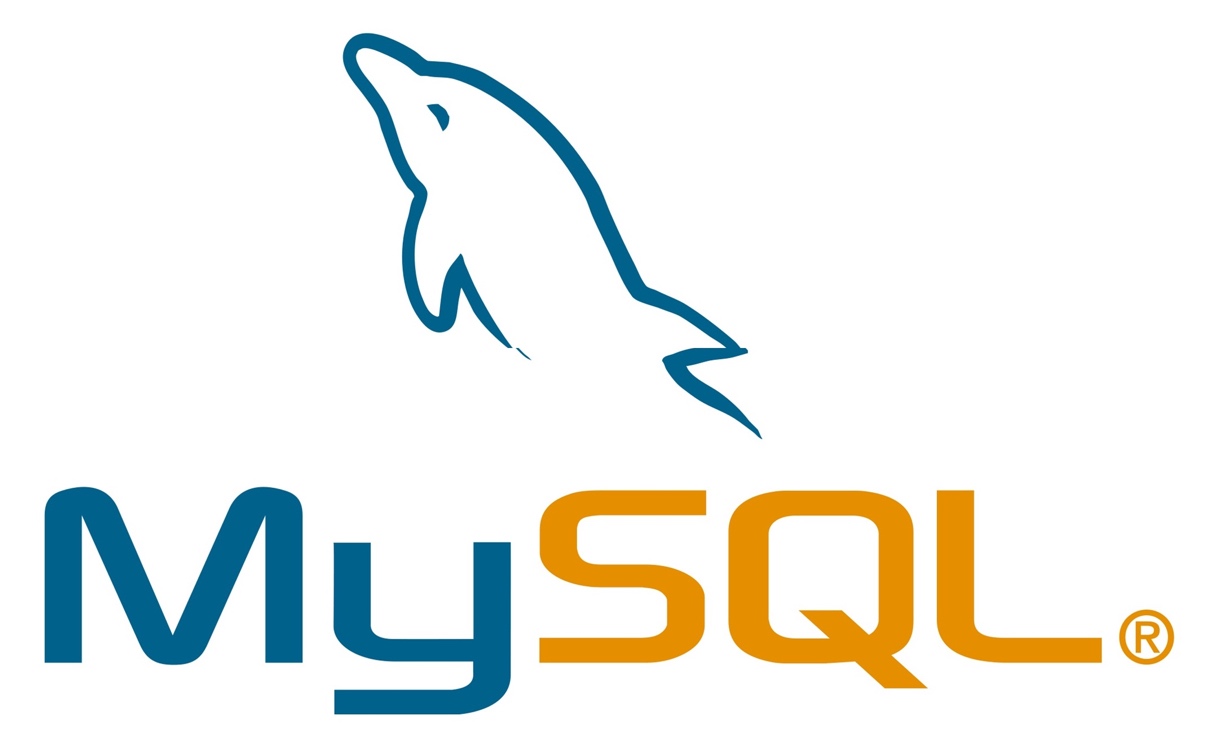
* Tablas : en un conjunto de registros. Una base de datos puede contener muchas tablas.

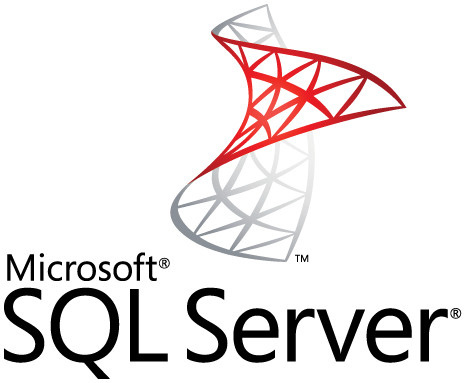
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CODIGO | NOMBRE | APELLIDO\_P | APELLIDO\_M | SUELDO |
| 1 | DANIEL | JIMENEZ | GARCIA | 8,000 |
| 1 | JUAN | LOPEZ | MARTINEZ | 13,000 |
| 3 | MARIA | PEREZ | GUITIERREZ | 5,000 |
| 4 | CRISTIAN | JUAREZ | HERNANDEZ | 2,000 |
| 5 | SARA | ROSALES | VAZQUEZ | 7,000 |

1. **Gestor de Base de Datos**

Un Sistema Gestor de Bases de Datos es un conjunto de programas no visibles que administran y gestionan la información que contiene una base de datos. A través de él se maneja todo acceso a la base de datos con el objetivo de servir de interfaz entre ésta, el usuario y las aplicaciones.

Básicamente, el gestor controla cualquier operación ejecutada por el usuario contra la base de datos. Para ello, se utilizan herramientas específicas, como sistemas de búsqueda y de generación de informes, además de distintas aplicaciones.









1. **Lenguaje para manejar – administrar una base de datos**

Al igual que en la programación estructurada u orientada a objetos, para poder manejar o administrar una base de datos se utiliza un lenguaje denominado de consulta; este lenguaje, como su nombre lo indica nos permite consultar información especifica de la base de datos.

Existen varios lenguajes de consulta para utilizarse con una base de datos, pero el mas utilizado hoy en día es:

* SQL: (Structured Query Language): Lenguaje de Consultas Estructurado

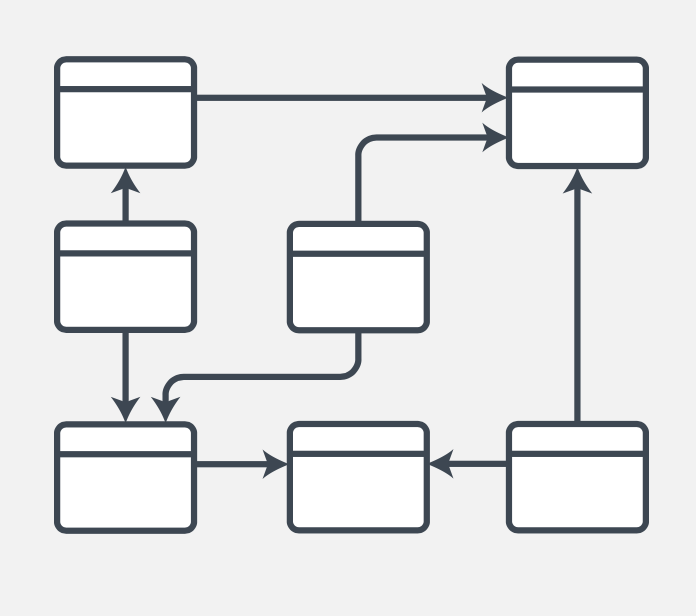


Consultas: Una consulta es un objeto que proporciona una visión personal de los datos almacenados en las tablas ya creadas.

Existen varios tipos de consultas para seleccionar, actualizar, borrar datos, etc., pero la mayoría de las veces se utilizan para extraer de las tablas los datos que cumplen ciertas condiciones.

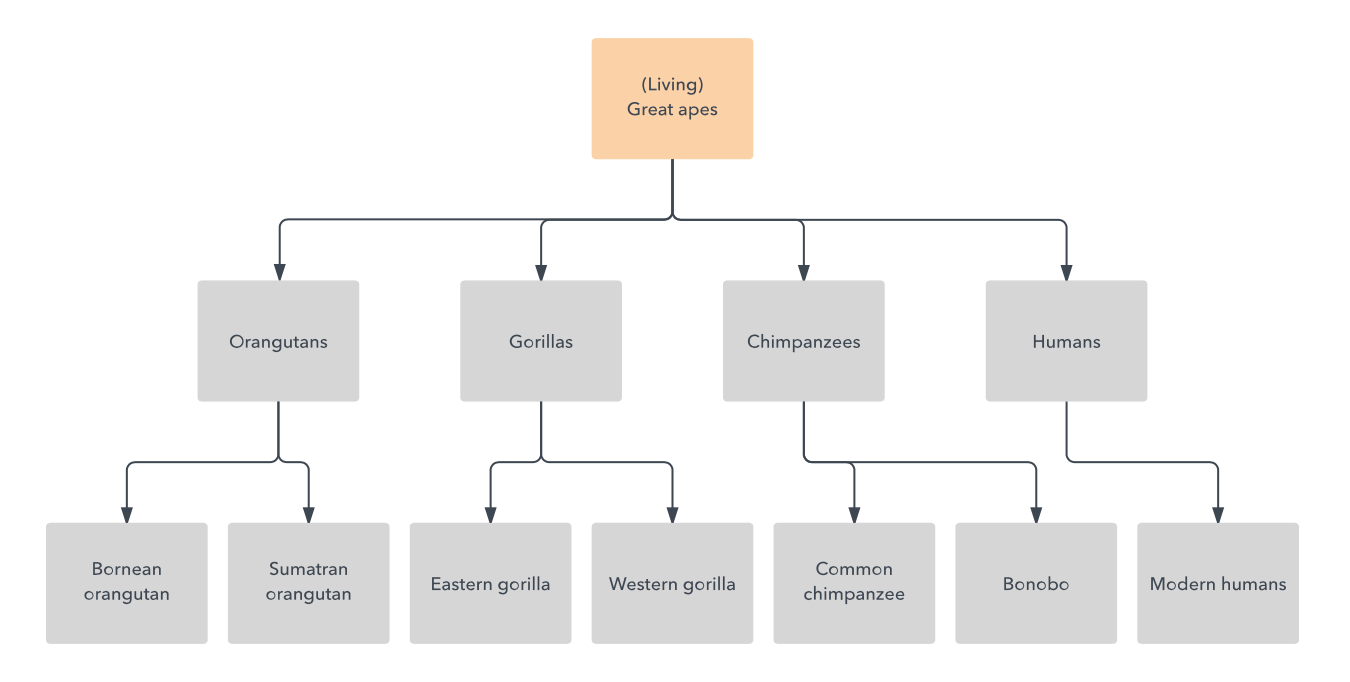
**MODELO DE BASE DE DATOS**

Un modelo de datos es una representación, usualmente grafica, de estructuras de datos.

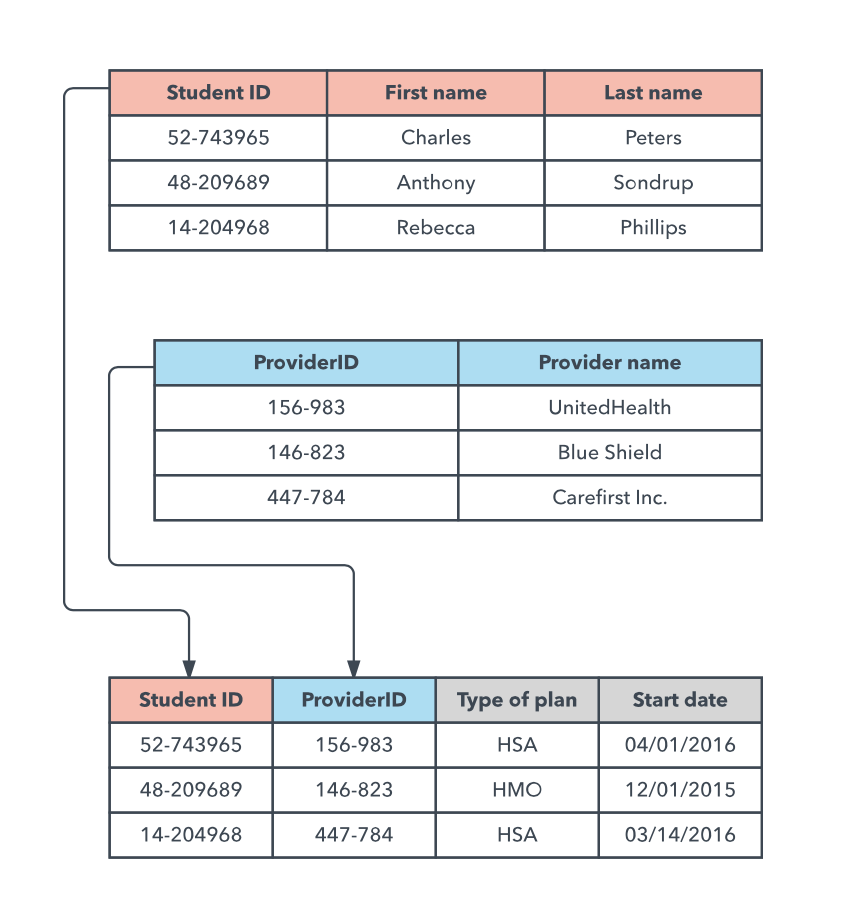


Un modelo de base de datos muestra la estructura lógica de la base, incluidas las relaciones y limitaciones que determinan cómo se almacenan los datos y cómo se accede a ellos.

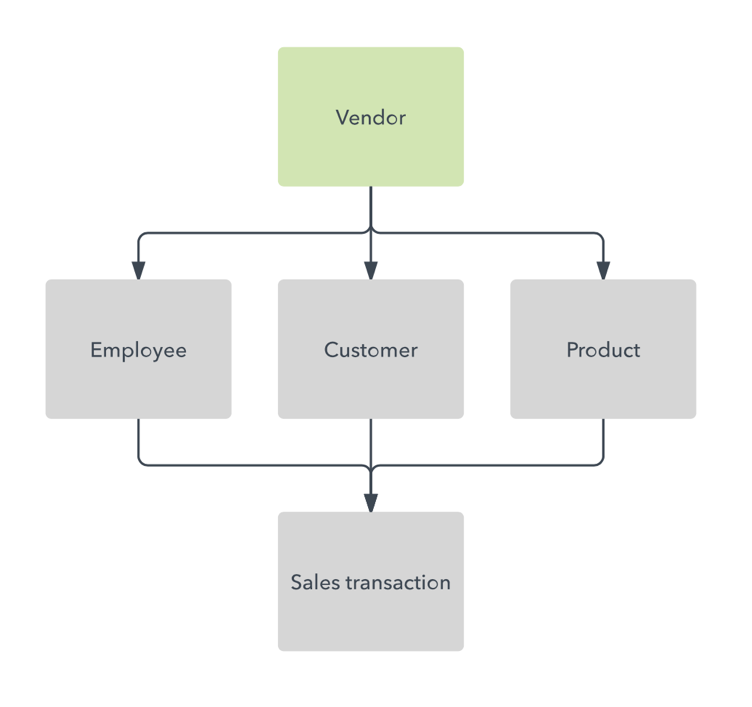
* Modelo de base de datos jerárquico: El modelo jerárquico organiza los datos en una estructura de árbol, en la que cada registro tiene un único elemento o raíz. Los registros del mismo nivel se clasifican en un orden específico.



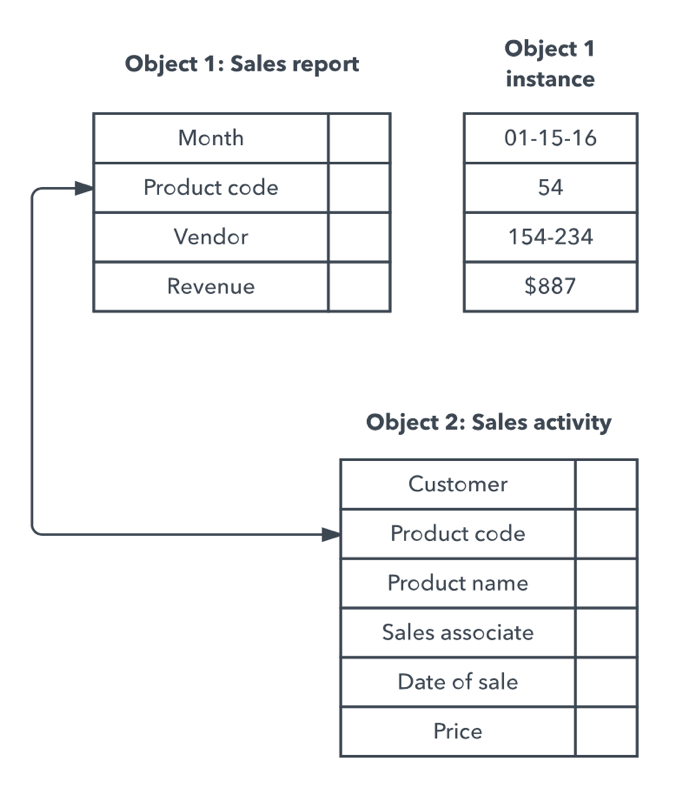
* **Modelo relacional:** Siendo el modelo más común, el modelo relacional ordena los datos en tablas, también conocidas como relaciones, cada una de las cuales se compone de columnas y filas.



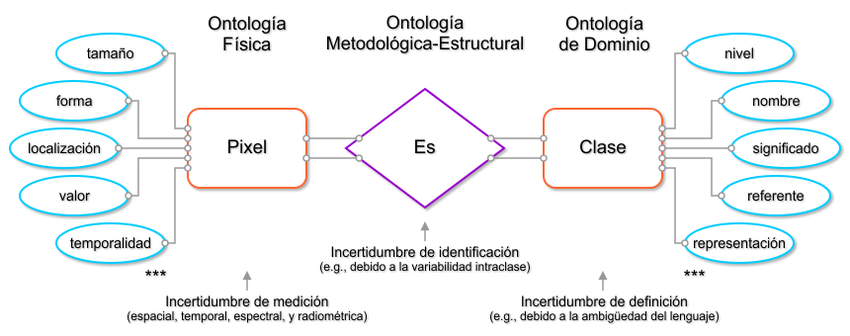
* Modelo de red: El modelo de red se basa en el modelo jerárquico, permitiendo relaciones de muchos a muchos entre registros vinculados, lo que implica registros principales múltiples.



* Modelo de base de datos orientado a objetos: Este modelo define una base de datos como una colección de objetos, o elementos de software reutilizables, con funciones y métodos relacionados.



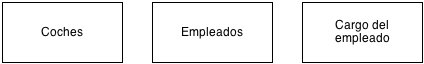
* **Modelo entidad-relación**: Este modelo capta las relaciones entre entidades del mundo real de forma muy similar al modelo de red, pero no está directamente ligado a una estructura física de la base de datos. En cambio, con frecuencia se lo usa para diseñar una base de datos conceptualmente.



**MODELO ENTIDAD-RELACION**

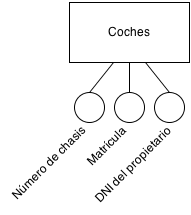
**Entidad**

Las entidades representan cosas u objetos (ya sean reales o abstractos), que se diferencian claramente entre sí.



**Atributos**

Los atributos definen o identifican las características de entidad (es el contenido de esta entidad). Cada entidad contiene distintos atributos, que dan información sobre esta entidad. Estos atributos pueden ser de distintos tipos (numéricos, texto, fecha...).



**Relación**

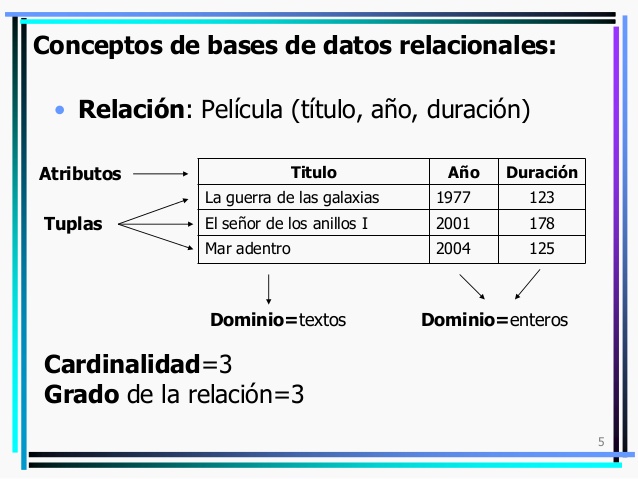
Es un vínculo que nos permite definir una dependencia entre varias entidades, es decir, nos permite exigir que varias entidades compartan ciertos atributos de forma indispensable.



**MODELO RELACIONAL**

Modelo de organización y gestión de bases de datos consistente en el almacenamiento de datos en tablas compuestas por filas, o tuplas, y columnas o campos.

* **Tabla**: es el nombre que recibe cada una de las relaciones que se establecen entre los datos almacenados; cada nueva relación da lugar a una tabla. Están formadas por filas, también llamadas tuplas, donde se describen los elementos que configuran la tabla (es decir, los elementos de la relación establecida por la tabla), columnas o campos, con los atributos y valores correspondientes, y el dominio, concepto que agrupa a todos los valores que pueden figurar en cada columna.
* **Claves**: elementos que impiden la duplicidad de registros, una de las grandes desventajas que presentan otros modelos de organización y [gestión de bases de datos](https://blog.es.logicalis.com/analytics/bid/370963/Gesti-n-de-la-base-de-datos). Existen dos grandes tipos de claves: las claves primarias y las secundarias o externas.
* **Claves** **primarias**: son los atributos (columnas) según el tipo de relación que se ha definido en la tabla. Pueden añadirse otros atributos específicos y propios.
* **Claves externas o secundarias**: son las claves que se definen para cada una de las claves primarias establecidas para los elementos o entidades de una relación.



**Primer base de datos**

**Planteamiento del problema:**

El dueño de la tiendita de la esquina necesita llevar un control de sus ventas, por lo que ha decidido implementar una base de datos para administrar su tienda. Para poder llevar un buen control de sus ventas, el dueño quiere conocer de cada producto que vendió, la marca y el precio y del cliente que lo compró su nombre y teléfono.

**Obtención de elementos para crear una BD**

**Producto: marca y precio.**

**Cliente: nombre y teléfono.**

**ID - Identificadores**

Entidad Producto

* Marca
* Precio
* IDProducto

Entidad Cliente

* Nombre
* Teléfono
* IDCliente

**Modelo Entidad Relación**

Compra

Cliente

Producto

**Modelo Relacional**

|  |
| --- |
| Producto |
| IDPRODUCTO |
| MARCA |
| PRECIO |

|  |
| --- |
| Compra |
| IDPRODUCTO |
| IDCLIENTE |
| MARCA |
| PRECIO |
| NOMBRE |
| TELEFONO |

|  |
| --- |
| Cliente |
| IDCLIENTE |
| NOMBRE |
| TELEFONO |

**Introducción a SQL**

Consultas -> Crear y manipular directamente la BD

**Componentes**

* Comandos
* Clausulas
* Operadores
* Funciones de agregado

**COMANDOS**

DLL -> Crean y definen nuevas Bases de Datos, campos, índices, etc.

DML -> Generar las consultas para ordenar, filtrar, extraer, etc.

**DLL**

CREATE: Crear nuevas tablas, campos e índices

DROP: Eliminar tablas e índices

ALTER: Modificar tablas, en especifico sus campos

**DML**

SELECT: Consultar registros de la BD

INSERT: Agregar datos a nuestra BD

UPDATE: Modificar valores de campos y registros

DELETE: Eliminar registros de una tabla

**CLAUSULAS**

FROM: Especificar la tabla que va a ser utilizada

WHERE: Especifica condiciones que deben reunir los registros que se van a seleccionar

GRUP BY: Separar los registros seleccionados en grupos específicos

HAVING: Expresa la condición que debe satisfacer cada grupo

ORDER BY: Ordena los registros seleccionados de acuerdo con un orden especifico

**OPERADORES LOGICOS**

AND “Y” lógico: TRUE -> ambos lados de la condición son verdaderos

OR “O” lógico: TRUE -> alguno de los lados de la condición sea verdadero

NOT -> negación lógica -> exactamente lo contrario

**OPERADORES DE COMPARACION**

< Menor que

> Mayor que

<> Distinto de

<= Menor o igual

>= Mayor o igual

BETWEEN: Especificar un intervalo de valores

LIKE: Comparación de un modelo

IN: Especificar registros de una BD

**FUNCIONES DE AGREGADO**

AVG: Calcular el promedio de los valores de un grupo determinado

COUNT: Devuelve el numero de registros de la selección utilizada

SUM: Sumar todos los valores de un campo determinado

MAX: Devolver el valor mas alto de un campo especifico

MIN: Devolver el valor mas bajo de un campo especifico

**CREAR UNA BASE DE DATOS**

CREATE DATA BASE NombreBD;

USE NombreBD;

DROP DATA BASE NombreBD;

**CREACION DE TABLAS EN UNA BASE DE DATOS**

CREATE TABLE NombreTabla (Campos de la tabla con sus definiciones);

CREATE TABLE NombreTabla( id\_empleado INTEGER PRIMARY KEY,

Nombre VARCHAR(20),

Sueldo INTEGER);

DROP TABLE NombreTabla;

**INSERTAR REGISTROS A UNA TABLA**

INSERT INTO NombreTabla VALUES (Valores a ingresar en la Tabla);

INSERT INTO NombreTabla VALUES ( 1, “Daniel”, 3000);

INSERT INTO NombreTabla VALUES ( 2, “Rosa”, 500);

INSERT INTO NombreTabla VALUES ( 3, Jaime, ‘2300’);

**ACTUALIZAR REGISTROS**

UPDATE NombreTabla SET Nombre = “Daniel” WHERE id\_empleado = 1;

UPDATE NombreTabla SET Sueldo = 1500 WHERE id\_empleado = 2;

**TABLA EMPLEADOS:**

|  |  |  |
| --- | --- | --- |
| Id\_empleado | Nombre | Sueldo |
| 1 | Daniel | 3000 |
| 2 | Rosa | 1500 |
| 3 | Roberto | 850 |

**SELECCIONAR REGISTROS DE UNA TABLA**

SELECT Campos FROM NombreTabla

WHERE Campo = 1;

SELECT \* FROM Empleados WHERE id\_empleado = 1;

|  |  |  |
| --- | --- | --- |
| Id\_empleado | Nombre | Sueldo |
| 1 | Daniel | 3000 |

SELECT Nombre FROM Empleados WHERE Nombre = “Roberto”;

|  |
| --- |
| Nombre |
| Roberto |

SELECT Nombre FROM Empleados WHERE id\_empleado = 2;

|  |
| --- |
| Nombre |
| Rosa |

SELECT \* FROM Empleados;

**Planteamiento del problema**

El salón de la FAMA de la FIFA requiere tener un registro de la cantidad de goles que hizo un jugador a lo largo de su vida en cada equipo que jugó. Se conoce el nombre de los equipos y de los jugadores.

- JUGADOR: nombre

- EQUIPO: nombre

**Modelo Entidad Relación**

Contrato

Jugador

Equipo

**Modelo Relacional**

|  |  |  |
| --- | --- | --- |
| EQUIPO | | |
| ID\_EQUIPO (INTEGER) | **NOMBRE\_E (VARCHAR)** |  |
| 001 | **Real Madrid** |  |
| 002 | **Barcelona** |  |

|  |  |  |
| --- | --- | --- |
| CONTRATO | | |
| ID\_EQUIPO (INTEGER) | **ID\_JUGADOR (INTEGER)** | **CantGoles (INTEGER)** |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| JUGADOR | | |
| ID\_JUGADOR (INTEGER) | **NOMBRE\_J (VARCHAR)** |  |
| 001 | **Cristiano Ronaldo** |  |
| 002 | **Lionel Messi** |  |

**Creación de Base de Datos en SQL**

//creado una base de datos llamada golesfifa

CREATE DATABASE golesfifa;

//usar la base de datos goles fifa

USE DATABASE golesfifa;

//crear tabla EQUIPO

CREATE TABLE EQUIPO ( ID\_EQUIPO INTEGER PRIMARY KEY,

NOMBRE\_E VARCHAR(30) );

//crear tabla JUGADOR

CREATE TABLE JUGADOR ( ID\_JUGADOR INTEGER PRIMARY KEY,

NOMBRE\_J VARCHAR(50) );

//crear tabla CONTRATO

CREATE TABLE CONTRATO (

ID\_EQUIPO INTEGER NOT NULL,

ID\_JUGADOR INTEGER NOT NULL,

CantGoles INTEGER,

PRIMARY KEY (ID\_EQUIPO, ID\_JUGADOR),

FOREIGN KEY (ID\_EQUIPO) REFERENCES EQUIPO (ID\_EQUIPO),

FOREIGN KEY (ID\_JUGADOR) REFERENCES JUGADOR (ID\_JUGADOR) );

CREATE TABLE CONTRATO (

ID\_ EQUIPO INTEGER NOT NULL REFERENCES EQUIPO,

ID\_JUGADOR INTEGER NOT NULL REFERENCES JUGADOR,

CantGoles INTEGER,

PRIMARY KEY (ID\_EQUIPO, ID\_JUGADOR) );

**Insertar Datos en SQL**

**INSERT INTO**

INSERT INTO EQUIPO VALUES ( ID\_EQUIPO , NOMBRE\_E);

INSERT INTO EQUIPO VALUES ( 1 , “Real Madrid”);

INSERT INTO EQUIPO VALUES ( 2, “Barcelona”);

INSERT INTO EQUIPO VALUES ( 3 , “Manchester United”);

INSERT INTO EQUIPO VALUES ( 4 , “Manchester City”);

INSERT INTO EQUIPO VALUES ( 5 , “AC Milan”);

INSERT INTO JUGADOR VALUES (ID\_JUGADOR, NOMBRE\_J);

INSERT INTO JUGADOR VALUES ( 1 , “Cristiano Ronaldo”);

INSERT INTO JUGADOR VALUES ( 2 , “Lionel Messi”);

INSERT INTO JUGADOR VALUES ( 3 , “Gareth Bale”);

INSERT INTO JUGADOR VALUES ( 4 , “Neymar Jr.”);

INSERT INTO JUGADOR VALUES ( 5 , “Oribe Peralta”);

INSERT INTO CONTRATO VALUES (ID\_EQUIPO, ID\_JUGADOR, CantGoles);

INSERT INTO CONTRATO VALUES ( 1, 3, 120);

INSERT INTO CONTRATO VALUES ( 2, 5, 20);

INSERT INTO CONTRATO VALUES ( 3, 4, 80);

|  |  |
| --- | --- |
| EQUIPO | |
| ID\_EQUIPO (INTEGER) | **NOMBRE\_E (VARCHAR)** |
| 1 | **Real Madrid** |
| 2 | **Barcelona** |
| 3 | **Manchester United** |
| 4 | **Manchester City** |
| 5 | **AC Milán** |

|  |  |  |
| --- | --- | --- |
| CONTRATO | | |
| ID\_EQUIPO (INTEGER) | **ID\_JUGADOR (INTEGER)** | **CantGoles (INTEGER)** |
| 1 | **3** | **120** |
| 2 | **5** | **20** |
| 3 | **4** | **80** |

|  |  |
| --- | --- |
| JUGADOR | |
| ID\_JUGADOR (INTEGER) | **NOMBRE\_J (VARCHAR)** |
| 1 | **Cristiano Ronaldo** |
| 2 | **Lionel Messi** |
| 3 | **Gareth Bale** |
| 4 | **Neymar Jr** |
| 5 | **Oribe Peralta** |

**Consultas SQL**

* OBTENER TODOS LOS DATOS DE LA TABLA DE JUGADORES

SELECT \* FROM JUGADOR;

|  |  |
| --- | --- |
| ID\_JUGADOR | NOMBRE\_J |
| 1 | **Cristiano Ronaldo** |
| 2 | **Lionel Messi** |
| 3 | **Gareth Bale** |
| 4 | **Neymar Jr** |
| 5 | **Oribe Peralta** |

* OBTENER TODOS LOS DATOS DE LA TABLA EQUIPOS

SELECT \* FROM EQUIPO;

|  |  |
| --- | --- |
| ID\_EQUIPO | NOMBRE\_E |
| 1 | **Real Madrid** |
| 2 | **Barcelona** |
| 3 | **Manchester United** |
| 4 | **Manchester City** |
| 5 | **AC Milan** |

* OBTENER EL NOMBRE DE TODOS LOS JUGADORES

SELECT NOMBRE\_J FROM JUGADOR;

|  |
| --- |
| NOMBRE\_J |
| Cristiano Ronaldo |
| Lionel Messi |
| Gareth Bale |
| Neymar Jr |
| Oribe Peralta |

* OBTENER TODOS LOS DATOS DEL EQUIPO BARCELONA

SELECT \* FROM EQUIPO WHERE NOMBRE\_E = “Barcelona”;

|  |  |
| --- | --- |
| 2 | Barcelona |

* OBTENER EL NOMBRE DEL EQUIPO CON ID = 3

SELECT NOMBRE\_E FROM EQUIPO WHERE ID\_EQUIPO = 3;

|  |
| --- |
| NOMBRE\_E |
| Manchester United |

* OBTENER EL ID DEL EQUIPO CON NOMBRE = AC MILAN

SELECT ID\_EQUIPO FROM EQUIPO WHERE NOMBRE\_E = “AC Milan”;

|  |
| --- |
| ID\_EQUIPO |
| 5 |

* OBTENER EL EQUIPO EN EL QUE JUEGA GARETH BALE

SELECT NOMBRE\_E

FROM EQUIPO, JUGADOR, CONTRATO

WHERE EQUIPO.ID\_EQUIPO = CONTRATO.ID\_EQUIPO

AND JUGADOR.ID\_JUGADOR = CONTRATO.ID\_JUGADOR

AND JUGADOR.NOMBRE\_J = “Gareth Bale”;

|  |
| --- |
| NOMBRE\_E |
| Real Madrid |

* OBTENER CUANTOS GOLES ANOTO GARETH BALE EN EL REAL MADRID

SELECT CantGoles

FROM EQUIPO, JUGADOR, CONTRATO

WHERE EQUIPO.ID\_EQUIPO = CONTRATO.ID\_EQUIPO

AND JUGADOR.ID\_JUGADOR = CONTRATO.ID\_JUGADOR

AND JUGADOR.NOMBRE\_J = “Gareth Bale”

AND EQUIPO.NOMBRE\_E = “Real Madrid”;

|  |
| --- |
| CantGoles |
| 120 |

* OBTENER CUANTOS GOLES ANOTO NEYMAR JR EN EN MANCHESTER UNITED

SELECT CantGoles

FROM EQUIPO, JUGADOR, CONTRATO

WHERE EQUIPO.ID\_EQUIPO = CONTRATO.ID\_EQUIPO

AND JUGADOR.ID\_JUGADOR = CONTRATO.ID\_JUGADOR

AND JUGADOR.NOMBRE\_J = “Neymar Jr”

AND EQUIPO.NOMBRE\_E = “Manchester United”;

|  |
| --- |
| CantGoles |
| 80 |