CURSO PROGRAMACIÓN WEB FULL-STACK

NIVEL 2

MÓDULO 4 – Bases de Datos

Ejercicio 1:

Crear un diagrama de Entidad Relación partiendo de las siguientes entidades:

- ALUMNO (cod_matricula, nombre, dni, fecha_nac, email)
- CURSO (cod_curso, nombre)
- PROFESOR (profesor_id, nombre, especialidad, email)

Teniendo en cuenta que:

- ✔ Un alumno puede estar inscripto en uno o varios cursos.
- ✔ Un curso es impartido por al menos uno o varios profesores.
- ✔ Un profesor podrá impartir varios cursos.

Ejercicio 2:

Crear un diagrama de Entidad Relación partiendo de las siguientes entidades:

- PAIS (pais_id, nombre_pais)
- PROVINCIA (provincia_id, nombre_provincia)
- LOCALIDAD (codigo_localidad, nombre, codigo_postal)
- EMPLEADO (empleado_id, dni, nombre, telefono, email, fecha_alta)

Se requiere almacenar los datos de cada uno de los empleados, para ello:

- ✔ Un empleado vive en una sola localidad.
- ✔ Cada localidad pertenece a una única provincia.
- ✔ Cada provincia pertenece a un país.
- ✔ Se pueden repetir los nombres de las provincias y localidades pero no de los países.

Ejercicio 3:

Se desea diseñar una base de datos sobre la información de las reservas de una empresa dedicada al alquiler de automóviles teniendo en cuenta que:

- ✔ Un determinado cliente puede tener en un momento dado una o varias reservas.
- ✔ De cada cliente se desea almacenar su DNI, nombre, dirección y teléfono.
- ✔ Además los clientes se diferencian entre sí por un código único.
- ✔ De cada reserva es importante registrar su número de identificación, la fecha de inicio y final de la reserva, el precio total.
- ✔ De cada coche se requiere la matrícula, el modelo, el color y la marca. Cada coche tiene un precio de alquiler por hora.

- ✔ Además en una reserva se pueden incluir varios coches de alquiler. Queremos saber los coches que incluye cada reserva y los litros de gasolina en el depósito en el momento de realizar la reserva, pues se cobrarán aparte.

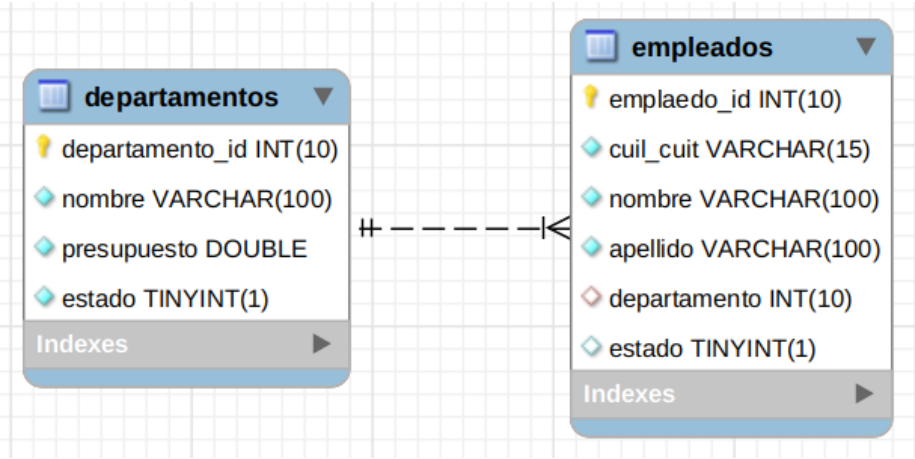
Ejercicio 4:

A partir de los enunciados del “Ejercicio 1” y “Ejercicio 2”, crear un script SQL con su nombre “ejercicio1.sql” y “ejercicio2.sql”, capaz de:

- Crear las tablas con los campos.
- Crear las relaciones entre las tablas. Tener en cuenta que las relaciones se dan por que existe al menos una PK y una FK
- Almacenar como mínimo 3 valores por cada tabla conservando las sentencias realizadas.

Ejercicio 5:

Partiendo del modelo entidad relación, ejecutar el script y resolver las consultas. Luego guardar las consultas en un archivo llamado “ejercicio5.sql”




```
CREATE TABLE departamentos (
departamento_id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
nombre VARCHAR(100) NOT NULL,
presupuesto DOUBLE UNSIGNED NOT NULL,
estado boolean NOT NULL
);

CREATE TABLE empleados (
emplaedo_id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
cuil_cuit VARCHAR(15) NOT NULL UNIQUE,
nombre VARCHAR(100) NOT NULL,
apellido VARCHAR(100) NOT NULL,
departamento INT UNSIGNED,
estado BOOLEAN,
FOREIGN KEY (departamento) REFERENCES departamentos(departamento_id));
```

```
INSERT INTO departamentos VALUES(1, 'Desarrollo', 120000, true);
INSERT INTO departamentos VALUES(2, 'Sistemas', 150000, true);
INSERT INTO departamentos VALUES(3, 'Recursos Humanos', 280000, true);
INSERT INTO departamentos VALUES(4, 'Contabilidad', 110000, true);
INSERT INTO departamentos VALUES(5, 'I+D', 375000, true);
INSERT INTO departamentos VALUES(6, 'Proyectos', 0,true );
INSERT INTO departamentos VALUES(7, 'Publicidad', 0, true);
INSERT INTO departamentos VALUES(8, 'Comercial', 0, false);

INSERT INTO empleados VALUES(1, '27-32481596-3', 'Aarón', 'Rivero', 1, true);
INSERT INTO empleados VALUES(2, '27-29557532-1', 'Adela', 'Salas', 2, true);
INSERT INTO empleados VALUES(3, '20-36970642-1', 'Adolfo', 'Rubio', 3, true);
INSERT INTO empleados VALUES(4, '20-41705545-1', 'Adrián', 'Suárez', 4, true);
INSERT INTO empleados VALUES(5, '20-17087203-3', 'Marcos', 'Loyola', 5, true);
INSERT INTO empleados VALUES(6, '27-38382980-3', 'María', 'Santana', 1, true);
INSERT INTO empleados VALUES(7, '23-80576669-1', 'Pilar', 'Ruiz', 2, true);
INSERT INTO empleados VALUES(8, '24-71651431-3', 'Pepe', 'Ruiz', 3, true);
INSERT INTO empleados VALUES(9, '25-36399183-3', 'Juan', 'Gómez', 2, true);
INSERT INTO empleados VALUES(10, '20-34638446-3', 'Diego','Flores', 5, true);
INSERT INTO empleados VALUES(11, '27-36738983-3', 'Marta','Herrera', 1, true);
INSERT INTO empleados VALUES(12, '27-44123836-1', 'Irene','Salas', NULL,
false);
INSERT INTO empleados VALUES(13, '20-38265162-1', 'Juan', 'Antonio', NULL,
true);
```

Resolver las siguientes consultas utilizando la sintaxis SQL

a) Lista el apellido de todos los empleados.
b) Lista el apellido de los empleados eliminando los apellidos que estén repetidos.
c) Lista todas las columnas de la tabla empleados.
d) Lista el nombre y apellido de todos los empleados.
e) Lista el cuit/cuil de los departamentos de los empleados que aparecen en la tabla empleados.
f) Lista el nombre y apellido de los empleados en una única columna.
g) Lista el nombre y apellido de los empleados en una única columna, convirtiendo todos los caracteres en mayúscula.
h) Lista el nombre y apellido de los empleados en una única columna, convirtiendo todos los caracteres en minúscula.
i) Lista el nombre de los departamentos y el valor del presupuesto actual ordenado de forma ascendente.
j) Lista el nombre de todos los departamentos ordenados de forma ascendente.
k) Lista el nombre de todos los departamentos ordenados de forma descendente.
l) Lista el apellido y el nombre de todos los empleados, ordenados de forma alfabética tendiendo en cuenta en primer lugar su apellido y luego su nombre.
m) Devuelve una lista con el nombre y el presupuesto, de los 3 departamentos que tienen mayor presupuesto.
n) Devuelve una lista con el nombre y el presupuesto, de los 3 departamentos que tienen menor presupuesto.
o) Devuelve una lista con el nombre de los departamentos y el presupuesto, de aquellos que tienen un presupuesto mayor o igual a $150000.

p) Devuelve una lista con el nombre de los departamentos y el presupuesto, de aquellos que tienen un presupuesto entre $100000 y $200000. Sin utilizar el operador BETWEEN.

q) Devuelve una lista con el nombre de los departamentos que no tienen un presupuesto entre $100000 y $200000. Sin utilizar el operador BETWEEN.

r) Devuelve una lista con el nombre de los departamentos que tienen un presupuesto entre $100000 y $200000. Utilizando el operador BETWEEN.

s) Devuelve un listado con los empleados y los datos de los departamentos donde trabaja cada uno.

t) Devuelve un listado con los empleados y los datos de los departamentos donde trabaja cada uno. Ordena el resultado, en primer lugar por el nombre del departamento (en orden alfabético) y en segundo lugar por apellido y el nombre de los empleados.

u) Devuelve un listado con el código y el nombre del departamento, solamente de aquellos departamentos que tienen empleados.

v) Devuelve el nombre del departamento donde trabaja el empleado que tiene el cuit 27-38382980-3.

w) Devuelve el nombre del departamento donde trabaja el empleado Pepe Ruiz.

x) Devuelve un listado con los datos de los empleados que trabajan en el departamento de I+D. Ordena el resultado alfabéticamente.

y) Devuelve un listado con los datos de los empleados que trabajan en el departamento de Sistemas, Contabilidad o I+D. Ordena el resultado alfabéticamente.

z) Devuelve una lista con el nombre de los empleados que tienen los departamentos que no tienen un presupuesto entre $100000 y $200000.

Ejercicio 6:

Subir al repositorio público creado en el TP N.º 3 los siguientes archivos:

I. Las imágenes o capturas de pantallas de los modelos confeccionados hasta el ejercicio 3.

II. Los scripts pertenecientes al ejercicio 4 y 5 (ejercicio1.sql, ejercicio2.sql y ejercicio5.sql).

Bibliografía:

- https://erdplus.com/
- https://www.w3schools.com/sql/default.asp