Proyecto Final

Integrantes:
Luis Alfonso Bravo Flores
Gonzalez Lopez Roberto Alejandro
Navarro Osorio Armando
Cruz Galván Alberto Israel

26/Mayo/2020

Introducción

El proyecto final consistió en crear una base de datos de una papelería que buscaba innovar la manera en que almacenará su información y que tuviera los siguientes requerimientos:

Se deseaba tener almacenados datos como la razón social, domicilio, nombre y teléfonos de los proveedores, razon social, nombre, domicilio y al menos un email de los clientes. Se necesitaba tener un inventario de los productos que se venden, en el que debe guardarse el código de barras, precio al que fue comprado el producto, fecha de compra y cantidad de ejemplares en la bodega (stock). Se deseaban guardar la marca, descripcion y precio de los regalos, artculos de papelera, impresiones y recargas, siempre y cuando se tenga su correspondiente registro en el inventario. Se debían también guardarse el número de venta, fecha de venta y la cantidad total a pagar de la venta, así como la cantidad de cada artículo y precio total a pagar por artículo.

Además, se requería que:

- Al recibir el código de barras de un producto, se regresará la utilidad.
- Cada que haya la venta de un artculo, se debía decrementar el stock por la cantidad vendida de ese artículo. Si el valor llegaba a cero, se abortaba la transacción. Si hay menos de 3, se emitía un mensaje.
- Dada una fecha, o una fecha de inicio y fecha de n, se regresaba la cantidad total que se vendió en esa fecha/periodo.
- Se obtuviera el nombre de aquellos productos de los cuales hay menos de 3 en stock. De manera automática se generará una vista que contenga información necesaria para asemejarse a una factura de una compra.
- Se creará al menos, un índice, del tipo que se prefiera y donde se prefiera.

Plan de Trabajo

1. La primera actividad del proyecto consistió en analizar detalladamente el problema presentado, para ello fue necesario que cada integrante del equipo diera su punto de vista de cual modelo entidad relación fuera el correcto, para así construirlo conjuntamente. Se definieron así tanto las entidades como los atributos y las relaciones correspondientes con cada uno de ellos. Finalmente con el diseño construido, el siguiente paso fue construir el modelo relacional que definiría las tablas de nuestra base de datos.

2. La siguiente actividad consistio en conformar el Development team que se encargaría de la programación del proyecto. Esté se conformo de la siguiente manera:

   (a) Alberto tomo el rol de desarrollo del backend que consistio en la programación en PostgresSQL de las relaciones propuestas así como de las funciones que serían utilizadas en la creación del proyecto.

   (b) Alejandro tomo el rol de desarrollo del Frontend que concierne a la conexión entre la base de datos y la página web, así como la programación de la interfaz gráfica.

   (c) Alfonso tomo el rol de desarrollador de la documentación en LATEX así como la integración de cada una de las partes del proyecto.

3. La última actividad del proyecto consistió en juntar cada una de las partes del proyecto y hacer las pruebas correspondietes de su funcionamiento.

Diseño

Como todo diseño, existen pasos a desarrollar y, lo dividiremos en tres fases:

1. **Diseño conceptual**
2. **Diseño lógico**
3. **Diseño físico**

El Diseño Conceptual
Para conseguir alcanzar el objetivo planteado, como equipo, primeramente definiremos la estructura de los datos que deben de tener nuestro sistema de información. Consistirá describir a alto nivel la estructura que tendrá nuestra base de datos. En él, se describió la información al almacenar en nuestro destino final, quye es nuestra base de datos llamada **PROYECTO_PAPELERIA**.

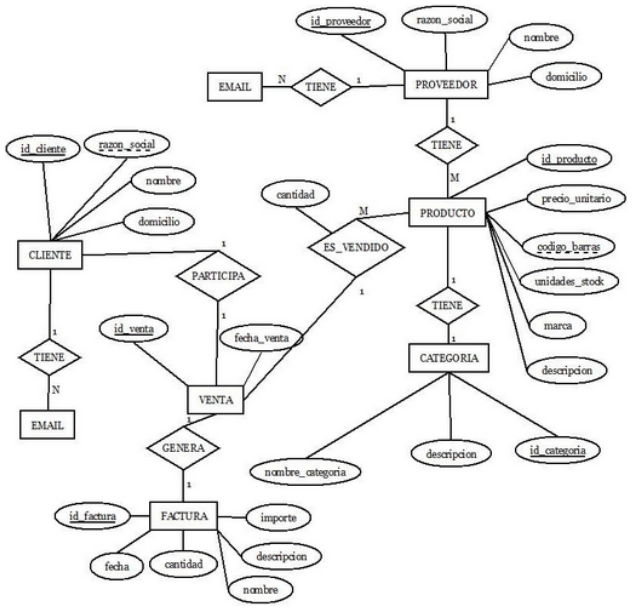

Figure 1: Diagrama Entidad - Relación

El Diseño lógico

Una vez obtenido el modelo entidad relación (MER), se comienza a crear, a partir de él, el modelo relacional (MR). Comenzamos replicando las entidades a tablas y los atributos de las entidades, fueron sustituidos por campos en dichas tablas, respetando, por obviedad, las claves primarias.
Se analizaron cada una de las relaciones que hubiera en el MER, se convirtieron a tablas y, se propagaron las llaves foráneas correspondientes.

```
CATEGORIA[id_categoria](PK),nombre_categoria, descripcion
--TABLA_1
PROVEEDORES[id_proveedor](PK),razon_social, domicilio,nombre
--TABLA_2
TELEFONO_PROVEEDORES[id_proveedor(FK),telefono](PK) --TABLA_3
PRODUCTOS[id_producto,id_precio_unitario,codigo_barras](PK),
[id_categoria,id_proveedor](FK),unidades_stock,marca,
descripcion--TABLA_4
CLIENTES[id_cliente (PK)],razon_social,nombre,domicilio
--TABLA_5
EMAIL_CLIENTES[id_cliente, email](PK) --TABLA_6
VENTAS[id_venta (PK)],id_cliente (FK), fecha_venta --TABLA_7
VENTA_DETALLES[id_venta (PK)](FK),id_producto(FK),
precio_unitario(FK),cantidad --TABLA_8
VENTA_DETALLES[id_factura(PK)],id_venta(FK),razon_social,
fecha,cantidad,nombre,desdescripcion,importe --TABLA_9
```

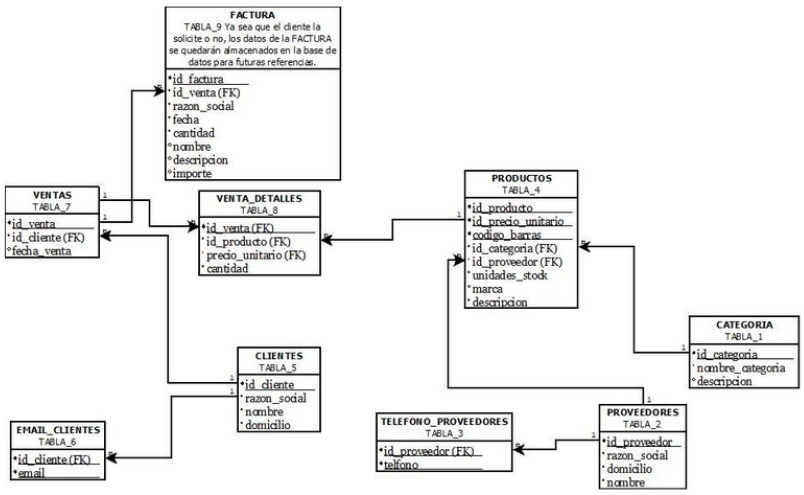

Figure 2: Modelo Entidad Relación

El Diseño físico
Evidentemente, el diseño físico, varía ligeramente de acuerdo al sistema generador de base de datos. El objetivo, será implementar físicamente (en memoria del servidor), del diseño lógico que se ha implementado, esto, con ayuda del lenguaje de consulta estructurado **SQL**.
El siguiente paso consistio en hacer el desarrollo, el cual lo dividimos en dos partes:

BACKEND

En la parte del servidor se crearon la base de datos, las tablas y se insertaròn los datos correspondientes. Primeramanete se creo la base de datos:

```
-----------------------------------------------------------------------
--AUTORES: solucionesTI_AAAA
--BD:PROYECTO_2020
--DESCRIPCIÒN: CREACIÒN DE LA BASE DE DATOS
--FECHA DE CREACIÒN
-----------------------------------------------------------------------
CREATE DATABASE "PROYECTO_PAPELERIA"
WITH
OWNER = postgres
ENCODING = 'UTF8'
CONNECTION LIMIT = -1;
```

La siguiente parte fue crear cada una de las tablas, con sus respectivas llaves, tanto primarias como foraneas, constrains, tipos de datos, etc. Lo anterior lo hicimos con la siguiente funció́n:

```
CREATE OR REPLACE FUNCTION fn_diseño_fisico() RETURNS VOID
LANGUAGE SQL
AS
$$
--=============================================================
--AUTORES: solucionesTI_AAAA
--BD:PROYECTO_PAPELERIA
--DESCRIPCIÓN: Diseño físico de la base de datos.
--FECHA DECREACIÓN
--=============================================================
-----------------------------------------------------------------------
CREATE TABLE CATEGORIA ( --TABLA_1
```

```
id_categoria INT GENERATED ALWAYS AS IDENTITY NOT NULL
,nombre_categoria VARCHAR(60) NOT NULL
,descripcion VARCHAR(100)
,CONSTRAINT PK_CATEGORIA PRIMARY KEY(id_categoria)
);
-------------------------------------------------------------------------
CREATE TABLE PROVEEDORES(--TABLA_2
id_proveedor INT GENERATED ALWAYS AS IDENTITY NOT NULL
,razon_social VARCHAR(100) NOT NULL
,domicilio VARCHAR(300) NOT NULL
,nombre VARCHAR(100) NOT NULL
,CONSTRAINT PK_PROVEEDORES PRIMARY KEY(id_proveedor)
);
-------------------------------------------------------------------------
CREATE TABLE TELEFONO_PROVEEDORES(--TABLA_3
id_proveedor INT NOT NULL
,telefono VARCHAR(48) NOT NULL
,CONSTRAINT PK_TELEFONO_PROVEEDORES PRIMARY
KEY(id_proveedor,telefono)
,CONSTRAINT FK_TELEFONO_PROOVEDORES FOREIGN
KEY(id_proveedor)
REFERENCES PROVEEDORES(id_proveedor)
);
-------------------------------------------------------------------------
CREATE TABLE PRODUCTOS( --TABLA_4
id_producto INT GENERATED ALWAYS AS IDENTITY NOT
NULL
,nombre_producto VARCHAR(50) NOT NULL
,precio_unitario DECIMAL NOT NULL
,id_categoria INT NOT NULL
,id_proveedor INT NOT NULL
,codigo_barras VARCHAR(100) NOT NULL
,unidades_stock INT NOT NULL
,marca VARCHAR(50) --PUEDE SER NULL
,descripcion VARCHAR(100) --PUEDE SER NULL
,CONSTRAINT PK_PRODUCTOS PRIMARY
KEY(id_producto,precio_unitario)
,CONSTRAINT FK_PRODUCTOS_CATEGORIA FOREIGN
KEY(id_categoria)
REFERENCES CATEGORIA(id_categoria)
,CONSTRAINT FK_PRODUCTOS_PROVEEDORES FOREIGN
KEY(id_proveedor)
REFERENCES PROVEEDORES(id_proveedor)
);
-------------------------------------------------------------------------
CREATE TABLE CLIENTES(--TABLA_5
```

```
id_cliente INT GENERATED ALWAYS AS IDENTITY NOT NULL
,razon_social VARCHAR(100) NOT NULL
,domicilio VARCHAR(300) NOT NULL
,nombre VARCHAR(100) NOT NULL
,CONSTRAINT PK_CLIENTE PRIMARY KEY(id_cliente)
);
-----------------------------------------------------------------------
CREATE TABLE EMAIL_CLIENTES(--TABLA_6
id_cliente INT NOT NULL
,email VARCHAR(50) NOT NULL
,CONSTRAINT PK_EMAIL_LIENTES PRIMARY KEY(id_cliente,email)
,CONSTRAINT FK_EMAIL_CLIENTES FOREIGN KEY(id_cliente)
REFERENCES CLIENTES(id_cliente)
);
-----------------------------------------------------------------------
CREATE TABLE VENTAS(--TABLA_7
id_venta VARCHAR(50) NOT NULL
,id_cliente INT NOT NULL
,fecha_venta DATE NOT NULL
,CONSTRAINT PK_VENTAS PRIMARY KEY(id_venta)
,CONSTRAINT FK_VENTAS_CLIENTES FOREIGN KEY(id_cliente)
REFERENCES CLIENTES(id_cliente)
);
-----------------------------------------------------------------------
CREATE TABLE VENTA_DETALLES(--TABLA_8
id_venta VARCHAR(50) NOT NULL
,id_producto INT NOT NULL
,precio_unitario DECIMAL NOT NULL
,cantidad INT NOT NULL
,CONSTRAINT PK_VENTA_DETALLES PRIMARY KEY(id_venta)
,CONSTRAINT FK_VENTA_DETALLES_VENTAS FOREIGN
KEY(id_venta)
REFERENCES VENTAS(id_venta)
,CONSTRAINT FK_VENTA_DETALLES_PRODUCTOS FOREIGN
KEY(id_producto,precio_unitario)
REFERENCES PRODUCTOS(id_producto,precio_unitario)
);
-----------------------------------------------------------------------
$$;
```

La siguiente parte fue insertar los datos utilizados a cada una de nuestras tablas. Igual que en el paso anterior, hicimos las insercciones mediante una función:

```
CREATE OR REPLACE FUNCTION fn_inserta_datos() RETURNS VOID
```

```
LANGUAGE SQL
AS $$
--============================================================
--AUTORES: solucionesTI_AAAA --BD:PROYECTO_PAPELERIA
--DESCRIPCIÓN:Insercion de datos.  --FECHA DECREACIÓN
--============================================================
------------------------------------------------------------
--TABLA_1 CATEGORIA
INSERT INTO CATEGORIA(nombre_categoria,descripcion)
VALUES('Papeleria','Articulos escolares y de oficina');
INSERT INTO CATEGORIA(nombre_categoria,descripcion)
VALUES('Regalos','Articulos relacionados a regalos (regalos y
envolturas)');
INSERT INTO CATEGORIA(nombre_categoria,descripcion)
VALUES('Servicio de impresiones','Imp.  B/N, color y en
diferentes tamaños');
INSERT INTO CATEGORIA(nombre_categoria,descripcion)
VALUES('Servicio de recargas','Recargas telefónicas a
todas las compañias');
--SELECT * FROM CATEGORIA;
------------------------------------------------------------
--TABLA_2 PROVEEDORES(ATRIBUTO DOMICILIO COMO
Estado|C.P.|Col.|Calle y No.|nombre)
INSERT INTO PROVEEDORES(razon_social,domicilio,nombre)
VALUES('Papeleria_Mesones','CDMX, 06080, Centro, Mesones
32','Papeleria Mesones');
INSERT INTO PROVEEDORES(razon_social,domicilio,nombre)
VALUES('Tony_Superpapeleria_Mesones','CDMX, 06090, Centro,
Mesones 160-B','Tony Superpapeleria Mesones');
INSERT INTO PROVEEDORES(razon_social,domicilio,nombre)
VALUES('La_Reyna_De_Mesones_SA_de_CV','CDMX, 06090,
Centro, Jesús María 118','La Reyna');
INSERT INTO PROVEEDORES(razon_social,domicilio,nombre)
VALUES('Mobil_Mex_SA_de_CV,','CDMX, 11570, Lomas de
Chapultepec, Monte Elbruz 132','Mobil Mex');
INSERT INTO PROVEEDORES(razon_social,domicilio,nombre)
VALUES('Grupo_Fila Dixon','EDOMEX, 54940, Tultitlan,Autopista
Mexico-Queretaro 104 Int C','Grupo Dixon');
INSERT INTO PROVEEDORES(razon_social,domicilio,nombre)
VALUES('Papel_S.A.','CDMX, 06720, Doctores, Doctor Andrade
78','Papeles en tamaños');
INSERT INTO PROVEEDORES(razon_social,domicilio,nombre)
VALUES('Fantasías_Miguel_SA_de_CV','EDOMEX, 53100,
Satelite, Cto Cirujanos 5','Fantasías Miguel');
INSERT INTO PROVEEDORES(razon_social,domicilio,nombre)
VALUES('Comercializadora_de_impresoras_Angel_SA_CV',
```

```
'CDMX, 06090,Centro,Mesones 55, Papeleria
Mesones','Angel_impresoras');
--SELECT * FROM PROVEEDORES;
.
.
.
.
.
.
.
.
```

FRONTEND

Implementación

REQUERIMIENTOS

1. Al recibir el código de barras de un producto, regrese la utilidad.

```
CREATE OR REPLACE FUNCTION fn_utilidad(ps_codigo_barras
VARCHAR(50)) RETURNS NUMERIC LANGUAGE plpgsql
--============================================================
--AUTORES: solucionesTI_AAAA --BD:PROYECTO_PAPELERIA
--DESCRIPCIÓN: Función que retorna la utilidad de un
producto
al recibir el código de barras --FECHA DE CREACIÓN
--============================================================
AS $$
DECLARE
tprecio_venta NUMERIC;
costo_adquisicion NUMERIC;
utilidad NUMERIC;
ganancia NUMERIC;
BEGIN
SELECT precio_unitario
FROM PRODUCTOS
WHERE codigo_barras=ps_codigo_barras
INTO precio_venta; --Obtenemos el precio de venta del
producto deseado (del catálogo).
SELECT 0.30*precio_venta
INTO ganancia; --Obtenemos la ganancia del articulo.
costo_adquisicion:=precio_venta-ganancia;--costo_
adquisicion=precio_venta-ganancia.
utilidad:=precio_venta-costo_adquisicion;--Utilidad es
resultado de esta resta.
RETURN utilidad;
END
--SELECT fn_utilidad('A-0100-R') Ejemplo de invocación de la
utilidad de la pluma.
--select * from productos;Para ver el catálogo.
```

```
$$;
```

2. Cada que haya la venta de un artículo, debería decrementarse el stock por la cantidad vendida de ese artículo. Si el valor llega a cero, abortar la transacción. Si hay menos de 3, emitir un mensaje.

```
CREATE OR REPLACE FUNCTION stock()
RETURNS trigger AS $$
BEGIN
IF unidades_stock<=0 THEN
ROLLBACK;
RAISE NOTICE 'No hay productos en almacen';
ELSE
--BEGIN
UPDATE productos SET
unidades_stock=unidades_stock-cantidad
WHERE productos.precio_unitario=venta_detalles.
precio_unitario and
productos.id_producto=venta_detalles.id_producto;
--COMMIT
IF unidades_stock<0 THEN
ROLLBACK;
RAISE NOTICE 'No se puede hacer la venta';
ELSIF unidades_stock<3 THEN
RAISE NOTICE 'Hay menos de 3 productos en
almacen';
END IF;
END IF;
--RETURN NULL;
END
$$ LANGUAGE plpgsql;
CREATE TRIGGER stock
BEFORE INSERT
ON VENTA_DETALLES
FOR EACH ROW
EXECUTE PROCEDURE stock();
```

3. Dada una fecha, o una fecha de inicio y fecha de fin, regresar la cantidad total que se vendió en esa fecha/periodo.

```
--============================================================
--AUTORES: solucionesTI_AAAA
--BD:PROYECTO_PAPELERIA
--DESCRIPCIÓN: Dada una fecha de inicio y una fecha de fin,
```

```
regresar la cantidad total que se vendió en esa
fecha/periodo.
============================================================
DECLARE
ln_total_vendido_por_periodo NUMERIC :=0;
ld_fecha_inicio DATE:=to_date(ps_fecha_
inicio,'YYYYMMDD');
ld_fecha_fin DATE:=to_date(ps_fecha_
fin,'YYYYMMDD');
BEGIN
SELECT SUM (precio_unitario*cantidad)
FROM VENTA_DETALLES VD
INNER JOIN VENTAS V on V.id_venta=VD.id_venta
WHERE V.fecha_venta BETWEEN ld_fecha_inicio AND
ld_fecha_fin
INTO ln_total_vendido_por_periodo;
--Mediante esta consulta, se obtienen
--la cantidad de dinero vendido en el periodo
solicitado.
RETURN ln_total_vendido_por_periodo;
--select
public.fn_cantidad_total_vendida_por_periodo
('20200507','20200507');
END
$$;
```

4. Permitir obtener el nombre de aquellos productos de los cuales hay menos de 3 en stock.

```
CREATE VIEW vw_articulos_por_agotarse AS
--============================================================
--AUTORES: solucionesTI_AAAA
--BD:PROYECTO_PAPELERIA
--DESCRIPCIÓN: Obtención de productos que estén por
agotarse,
es decir, aquellos productos cuyas unidades en stock sean
menores a 3.
--LLAMADO: SELECT * FROM vw_articulos_por_agotarse
--============================================================
SELECT nombre_producto,descripcion
FROM PRODUCTOS
WHERE unidades_stock <=2 AND unidades_stock>0;
```

5. De manera automática se genere una vista que contenga información necesaria para asemejarse a una factura de una compra.

```
CREATE OR REPLACE FUNCTION fn_genera_factura()
RETURNS TRIGGER LANGUAGE plpgsql
--============================================================
--AUTORES: solucionesTI_AAAA
--BD:PROYECTO_PAPELERIA
--DESCRIPCIÓN: Genera datos de factura, de forma automática
al generar una venta.
--FECHA DE CREACIÓN
--============================================================
AS
$$ DECLARE
li_id_venta VARCHAR(50);
ls_razon_social VARCHAR(100);
ld_fecha_date DATE:=current_date;
li_cantidad INT:=0;
ls_nombre_producto VARCHAR(50);
ls_descripcion VARCHAR(100);
li_importe NUMERIC;
BEGIN
li_id_venta:=new.id_venta;
------------------------------------------------------------------
SELECT razon_social
FROM CLIENTES Cte
INNER JOIN VENTAS V ON V.id_cliente=Cte.id_cliente
INNER JOIN VENTA_DETALLES VD ON VD.id_venta=V.id_venta
WHERE VD.id_venta=new.id_venta
INTO ls_razon_social;--Obtenemos la razon social del cliente
que compra.
------------------------------------------------------------------
li_cantidad:=new.cantidad;
------------------------------------------------------------------
SELECT nombre_producto
FROM PRODUCTOS
WHERE id_producto=new.id_producto
INTO ls_nombre_producto;
------------------------------------------------------------------
SELECT descripcion
FROM PRODUCTOS
WHERE id_producto=new.id_producto
INTO ls_descripcion;
------------------------------------------------------------------
li_importe:=new.cantidad*new.precio_unitario;
------------------------------------------------------------------
INSERT INTO FACTURA(id_venta,razon_social,fecha,cantidad,
nombre_producto,descripcion,importe)
VALUES(li_id_venta,ls_razon_social,ld_fecha_date,
```

```
li_cantidad,ls_nombre_producto,ls_descripcion,li_importe);
return new;
END $$;
CREATE TRIGGER tr_crea_factura AFTER INSERT on VENTA_
DETALLES FOR EACH ROW
EXECUTE PROCEDURE fn_genera_factura()
```

Presentación

Conclusiones

1. **Luis Alfonso Bravo Flores:** Los objetivos del curso se cumplieron con este proyecto, ya que pusimos en práctica todo lo visto en clase. El proyecto fue un reto importante ya que enfrentamos dificultades de operación puesto que nos tuvimos que comunicar en linea por el problema de la pandemia. Cada uno de los integrantes aportó deacuerdo a sus habilidades para que este proyecto se construyera. Nos llevamos una buena experiencia para futuros proyectos.

2. **Cruz Galván Alberto Israel:** sin lugar a dudas la realización de este proyecto, conjugó un reto en múltiples sentidos. Desde la capacidad de trabajo en equipo, organización, delegar responsabilidades, también, poner a prueba nuestros conocimientos adquiridos a lo largo del semestre de la asignatura y desde luego, nuestras capacidades como ingenieros para resolver problemas.Durante el desarrollo del proyecto, se tuvo que llevar a la práctica la teoría del diseño de la base de datos, desde la parte del diseño, entendiendo los requerimientos del problema para que conjuntamente, se plantearan y se resolvieran en equipo de la mejor manera posible. El trabajo en equipo y la organización son inherentes a cualquier ramo ingenieril.

3. **Gonzalez Lopez Roberto Alejandro:** Este proyecto me demostro la importancia del manejo de tiempo y de personas. Ademas de la importancia del trabajo de investigacion, ya que se dieron problemas con la interfaz grafica que no se tenian contempladas y otros problemas con la base de datos con las especificaciones. Aun asi solo queda aprender de este proyecto y aplicarlo a futuros.

4. **Navarro Osorio Armando:**Con el desarrollo de este proyecto se adquirieron nuevos conocimientos así como nuevas habilidades, las cuales son muy usadas en la vida profesional. La que más cabe mencionar fue el trabajo en equipo ya que se tiene que tener un buena bitácora para desarrollar cada uno de los puntos en tiempo y forma. Fue un proyecto en donde se puso en juego nuestra propia disciplina así como el uso de ser autodidactas en la parte de la base de datos.

Bibliography

[1] campusMVP. Diseñando una base de datos en el modelo relacional. 2014. URL: https://www.campusmvp.es/recursos/post/Disenando-una-base-de-datos-en-el-modelo-relacional.aspx (visited on 05/22/2020).