



Universidad Nacional Autónoma de México

Facultad de Ingeniería

Bases de Datos

Proyecto

Grupo: 01

Profesor: Ing. Fernando Arreola Franco

Integrantes del equipo:

- Calderón González José Gabriel
- Ceballos Ricardo Fernando
- Hernández Bacilio Jesús Emilio
- Serralde Flores Andrea

Fecha de entrega: 15 de Agosto de 2021

Índice

Objetivo	3
Introducción	3
Plan de Trabajo	3
Diseño.....	4
Implementación	6
Presentación	10
Conclusiones.....	13
Bibliografía	14

Objetivo

El alumno analizará una serie de requerimientos y propondrá una solución que atienda a los mismos, aplicando los conceptos vistos en el curso.

Introducción

Dentro del desarrollo de este proyecto se llevó a cabo el diseño, implementación y análisis de una base de datos en la cual utilizamos lo aprendido a lo largo del curso cómo son consultas, joins, lenguaje DDL, etc., se puede observar durante el mismo la implementación de la base de datos una herramienta de escritorio para poder utilizarla sin uso de la línea de comandos, sin más, estaremos mostrando esto a lo largo del documento.

Plan de Trabajo

Nuestra planeación del trabajo consistió en que todos realizáramos una parte individual del trabajo, pero en caso de llegar a tener dificultades consultar con el resto del equipo para considerar nuevas formas de resolver el problema.

Integrante	Tarea
Fernando	Análisis y generación de tablas
Andrea	Consultas y documentación
Emilio	División de tareas, consultas, documentación
José	Unión de la base de datos a parte gráfica.

Diseño

Realizamos el análisis de los requerimientos de nuestra base de datos de manera que lo resaltamos con diversos colores para mejor identificación.

Se desea tener almacenados datos como la razón social, domicilio, nombre y teléfonos de los proveedores, rfc, nombre, domicilio y al menos un email de los clientes. Es necesario 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 desea guardar la marca, descripción y precio de los regalos, artículos de papelería, impresiones y recargas, siempre y cuando se tenga su correspondiente registro en el inventario. Debe 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 requiere que:

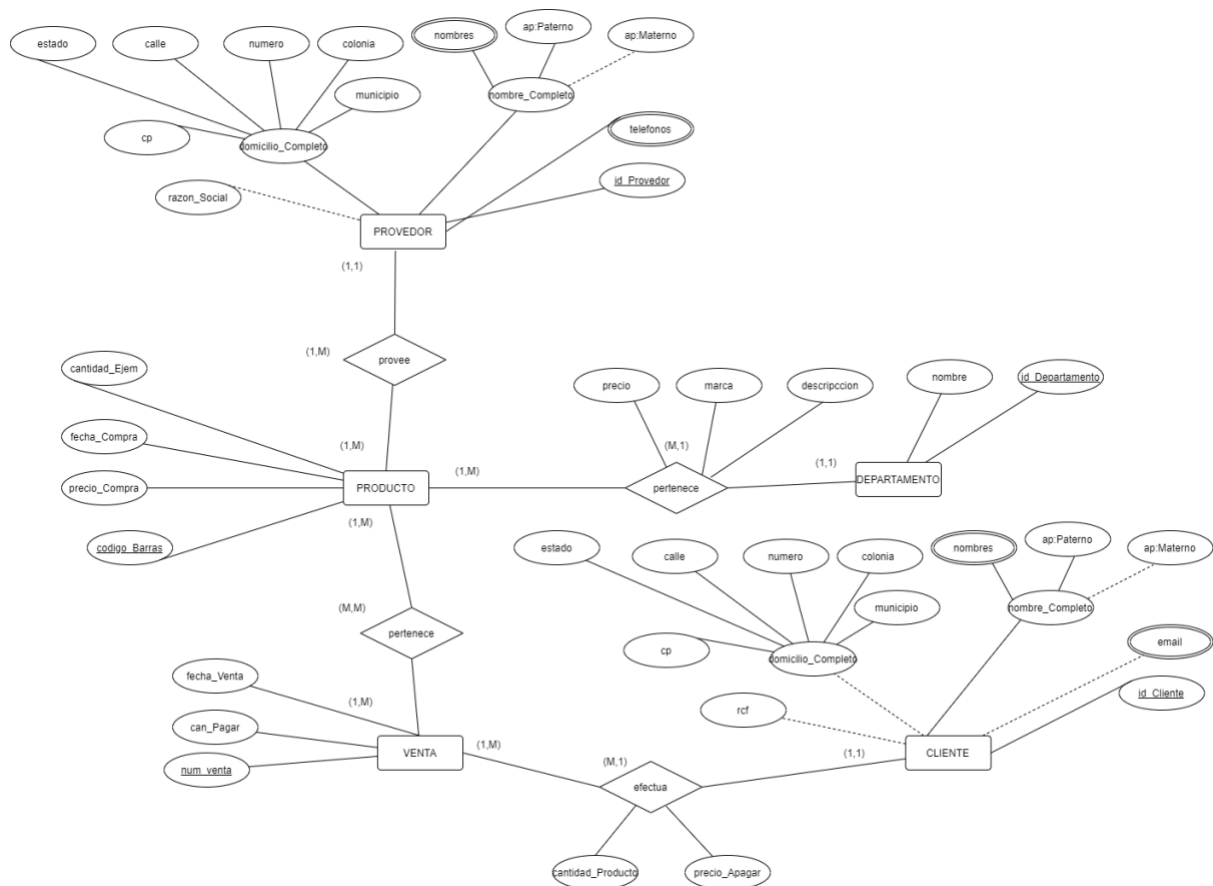
- Al recibir el código de barras de un producto, regrese la utilidad.
- Cada que haya la venta de un artículo, deberá decrementarse el stock por la cantidad vendida de ese artículo. Si el valor llega a cero, abortar la transacción. Si hay menos de 3, emitir un mensaje.
- Dada una fecha, o una fecha de inicio y fecha de fin, regresar la cantidad total que se vendió en esa fecha/periodo.

De forma que las resaltadas en rojo son las entidades, en azul los atributos, rosa las consultas y amarillo y verde notas importantes para el uso de las entidades.

Posteriormente procedemos a crear nuestra base en Postgres para validar que se encuentre en los parámetros requeridos y agregamos algunos registros para probar su integridad.

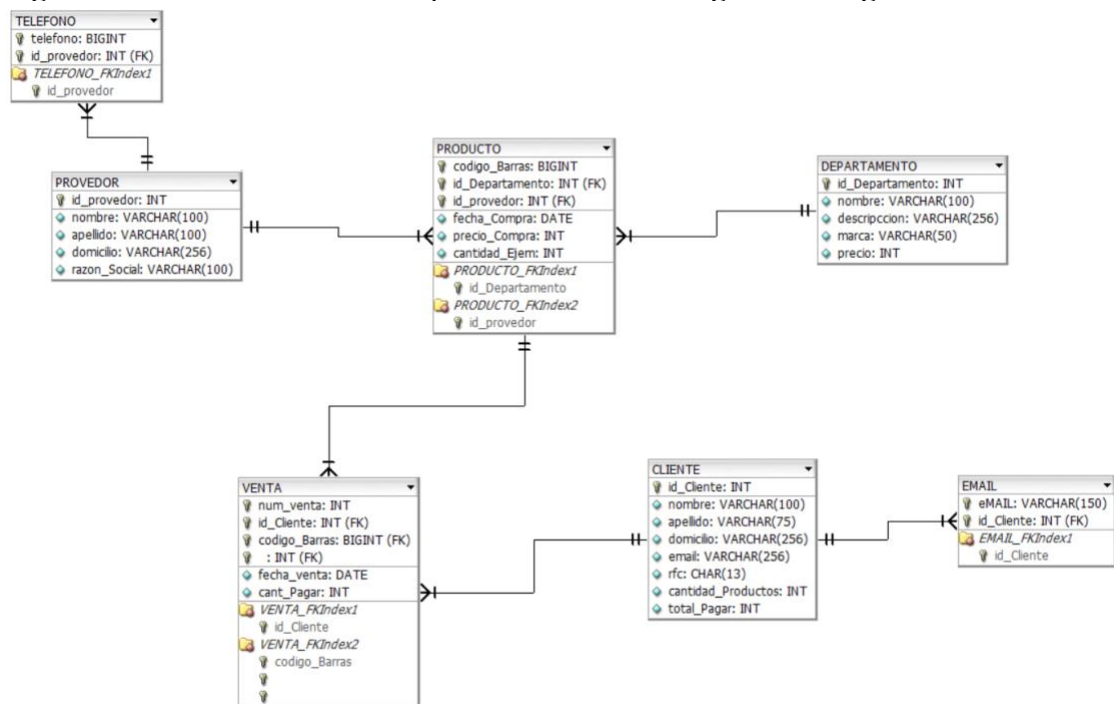
Para su versión de escritorio utilizamos (ingresar el nombre de la herramienta que haya usado el compañero) para que fuera de forma gráfica ya que al momento de realizar nuestro primer intento de alojarla en un servidor en línea se nos estaba complicando de más la conexión de la base, aun con esto, se puede visualizar de forma correcta la acción de nuestras consultas.

(aún falta la información de lo que ha realizado nuestro compañero de la herramienta gráfica para poder continuar con esta parte del desarrollo)



MER (Modelo Entidad Relación).

El diagrama Entidad-Relación correspondiente al diseño lógico es el siguiente



MR (Modelo Relacional).

CLIENTE							
ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
id_Cliente	INT	PK	NN				
nombre	VARCHAR(100)						
apellido	VARCHAR(75)						
domicilio	VARCHAR(256)						
email	VARCHAR(256)						
rfc	CHAR(13)						
cantidad_Productos	INT						
total_Pagar	INT						
IndexName	IndexType		Columns				
PRIMARY	PRIMARY		id_Cliente				

DEPARTAMENTO							
ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
id_Departamento	INT	PK	NN				AI
nombre	VARCHAR(100)						
descripcion	VARCHAR(256)						
marca	VARCHAR(50)						
precio	INT						
IndexName	IndexType		Columns				
PRIMARY	PRIMARY		id_Departamento				

EMAIL							
ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
eMAIL	VARCHAR(150)	PK	NN				AI
id_Cliente	INT	PK	NN				
IndexName	IndexType		Columns				
PRIMARY	PRIMARY		eMAIL, id_Cliente				
EMAIL_FKIndex1	Index		id_Cliente				

Implementación

Creación de tablas:

Para la creación de la tabla Producto

```
CREATE TABLE PRODUCTO (
CODIGO_BARRAS NUMERIC(12,0) NOT NULL,
PRECIO_COMPRA NUMERIC(10,0) NOT NULL,
PRECIO_VENTA NUMERIC(10,0) NOT NULL,
FECHA_COMPRA DATE,
CANTIDAD_EJEM NUMERIC(10,0) NOT NULL,
NOMBRE VARCHAR(100),
MARCA VARCHAR(50),
DESCRIPCION VARCHAR(100),
ID_PROVEDOR NUMERIC(4,0),
CONSTRAINT PK_PRODUCTO PRIMARY KEY (CODIGO_BARRAS)
);
```

Para la creación de la tabla Proveedor

```
CREATE TABLE PROVEDOR (
ID_PROVEDOR NUMERIC(4,0) NOT NULL,
NOMBRE VARCHAR(100) NOT NULL,
APELLIDO VARCHAR(100) NOT NULL,
ESTADO VARCHAR(32),
COD_POSTAL CHAR(5),
COLONIA VARCHAR(32),
CALLE VARCHAR(32),
NUM NUMERIC(4,0),
RAZON_SOCIAL VARCHAR(100) NOT NULL,
```

```
CONSTRAINT PK_PROVEDOR PRIMARY KEY (ID_PROVEDOR)
);
```

Para la creación de la tabla Telefono

```
CREATE TABLE TELEFONO (
ID_PROVEDOR NUMERIC(4,0) NOT NULL,
TELEFONO NUMERIC(10,0) NOT NULL,
CONSTRAINT PK_TELEFONO PRIMARY KEY (TELEFONO)
);
```

Para la creación de la tabla Venta

```
CREATE TABLE VENTA (
NUM_VENTA CHAR(8) NOT NULL,
ID_CLIENTE NUMERIC(4,0) NOT NULL,
CODIGO_BARRAS NUMERIC(12,0) NOT NULL,
FECHA_VENTA DATE NOT NULL,
CANTIDAD_PAGAR NUMERIC(10,0) NOT NULL,
CONSTRAINT PK_VENTA PRIMARY KEY (NUM_VENTA)
);
```



```
serralde_a=# CREATE TABLE PRODUCTO(
serralde_a=# CODIGO_BARRAS NUMERIC(12,0) NOT NULL,
serralde_a=# PRECIO_COMPRA NUMERIC(10,0) NOT NULL,
serralde_a=# PRECIO_VENTA NUMERIC(10,0) NOT NULL,
serralde_a=# FECHA_COMPRA DATE,
serralde_a=# CANTIDAD Ejem NUMERIC(10,0) NOT NULL,
serralde_a=# NOMBRE VARCHAR(100),
serralde_a=# MARCA VARCHAR(50),
serralde_a=# DESCRIPCION VARCHAR(100),
serralde_a=# ID_PROVEDOR NUMERIC(4,0),
serralde_a=# CONSTRAINT PK_PRODUCTO PRIMARY KEY (CODIGO_BARRAS)
serralde_a=# );
CREATE TABLE
serralde_a=#
serralde_a=#
serralde_a=# CREATE TABLE PROVEDOR(
serralde_a=# ID_PROVEDOR NUMERIC(4,0) NOT NULL,
serralde_a=# NOMBRE VARCHAR(100) NOT NULL,
serralde_a=# APELLIDO VARCHAR(100) NOT NULL,
serralde_a=# ESTADO VARCHAR(32),
serralde_a=# COD_POSTAL CHAR(5),
serralde_a=# COLONIA VARCHAR(32),
serralde_a=# CALLE VARCHAR(32),
serralde_a=# NUM NUMERIC(4,0),
serralde_a=# RAZON_SOCIAL VARCHAR(100) NOT NULL,
serralde_a=# CONSTRAINT PK_PROVEDOR PRIMARY KEY (ID_PROVEDOR)
serralde_a=# );
CREATE TABLE
serralde_a=#
serralde_a=#
serralde_a=# CREATE TABLE TELEFONO(
serralde_a=# ID_PROVEDOR NUMERIC(4,0) NOT NULL,
serralde_a=# TELEFONO NUMERIC(10,0) NOT NULL,
serralde_a=# CONSTRAINT PK_TELEFONO PRIMARY KEY (TELEFONO)
serralde_a=# );
CREATE TABLE
serralde_a=#
serralde_a=#
serralde_a=# CREATE TABLE VENTA(
serralde_a=# NUM_VENTA CHAR(8) NOT NULL,
serralde_a=# ID_CLIENTE NUMERIC(4,0) NOT NULL,
serralde_a=# CODIGO_BARRAS NUMERIC(12,0) NOT NULL,
serralde_a=# FECHA_VENTA DATE NOT NULL,
serralde_a=# CANTIDAD_PAGAR NUMERIC(10,0) NOT NULL,
serralde_a=# CONSTRAINT PK_VENTA PRIMARY KEY (NUM_VENTA)
serralde_a=# );
CREATE TABLE
```

```
serralde_a — psql -p5432 serralde_a — 67x22
~ — psql -p5432 serralde_a

serralde_a=# CREATE TABLE CLIENTE(
serralde_a(# ID_CLIENTE NUMERIC(4,0) NOT NULL,
serralde_a(# NOMBRE VARCHAR(100) NOT NULL,
serralde_a(# APELLIDO VARCHAR(100) NOT NULL,
serralde_a(# ESTADO VARCHAR(32),
serralde_a(# COD_POSTAL CHAR(5),
serralde_a(# COLONIA VARCHAR(32),
serralde_a(# CALLE VARCHAR(32),
serralde_a(# NUM NUMERIC(4,0),
serralde_a(# RFC VARCHAR(13) NOT NULL,
serralde_a(# CONSTRAINT PK_CLIENTE PRIMARY KEY (ID_CLIENTE)
serralde_a(# );
CREATE TABLE
serralde_a=#
serralde_a=#
serralde_a=# CREATE TABLE EMAIL(
serralde_a(# ID_CLIENTE NUMERIC(4,0) NOT NULL,
serralde_a(# EMAIL VARCHAR(150) NOT NULL,
serralde_a(# CONSTRAINT PK_EMAIL PRIMARY KEY (EMAIL)
serralde_a(# );
CREATE TABLE
serralde_a=#
```

Se alteran para definir las llaves foraneas

```
serralde_a — psql -p5432 serralde_a — 67x31
~ — psql -p5432 serralde_a

serralde_a=#
serralde_a=# ALTER TABLE PRODUCTO
serralde_a=# ADD CONSTRAINT FK_PROPROVEDOR
serralde_a=# FOREIGN KEY (ID_PROVEDOR)
serralde_a=# REFERENCES PROVEDOR (ID_PROVEDOR);
ALTER TABLE
serralde_a=#
serralde_a=# ALTER TABLE VENTA
serralde_a=# ADD CONSTRAINT FK_VENCLIENTE
serralde_a=# FOREIGN KEY (ID_CLIENTE)
serralde_a=# REFERENCES CLIENTE (ID_CLIENTE);
ALTER TABLE
serralde_a=#
serralde_a=# ALTER TABLE VENTA
serralde_a=# ADD CONSTRAINT FK_VENPRODCUTO
serralde_a=# FOREIGN KEY (CODIGO_BARRAS)
serralde_a=# REFERENCES PRODUCTO (CODIGO_BARRAS);
ALTER TABLE
serralde_a=#
serralde_a=# ALTER TABLE TELEFONO
serralde_a=# ADD CONSTRAINT FK_TELPROVEDOR
serralde_a=# FOREIGN KEY (ID_PROVEDOR)
serralde_a=# REFERENCES PROVEDOR (ID_PROVEDOR);
ALTER TABLE
serralde_a=#
serralde_a=# ALTER TABLE EMAIL
serralde_a=# ADD CONSTRAINT FK_EMAILCLIEN
serralde_a=# FOREIGN KEY (ID_CLIENTE)
serralde_a=# REFERENCES CLIENTE (ID_CLIENTE);
ALTER TABLE
serralde_a=#
```



```
papeleria-# \d producto;
Table "public.producto"
  Column      |      Type      | Collation | Nullable | Default
-----+-----+-----+-----+-----
codigo_barras | numeric(12,0)   |           | not null |
precio_compra | numeric(10,0)   |           | not null |
precio_venta  | numeric(10,0)   |           | not null |
fecha_compra  | date            |           |          |
cantidad Ejem | numeric(10,0)   |           | not null |
nombre        | character varying(100) |           |          |
marca         | character varying(50)  |           |          |
descripcion   | character varying(100) |           |          |
id_proveedor  | numeric(4,0)     |           |          |
Indexes:
    "pk_producto" PRIMARY KEY, btree (codigo_barras)
Foreign-key constraints:
    "fk_proveedor" FOREIGN KEY (id_proveedor) REFERENCES proveedor(id_proveedor)
Referenced by:
    TABLE "venta" CONSTRAINT "fk_venprodcuto" FOREIGN KEY (codigo_barras) REFERENCES producto(codigo_barras)

papeleria-# \d proveedor;
Table "public.proveedor"
  Column      |      Type      | Collation | Nullable | Default
-----+-----+-----+-----+-----
id_proveedor  | numeric(4,0)     |           | not null |
nombre        | character varying(100) |           | not null |
apellido      | character varying(100) |           | not null |
estado        | character varying(32) |           |          |
cod_postal    | character(5)      |           |          |
colonia       | character varying(32) |           |          |
calle         | character varying(32) |           |          |
num           | numeric(4,0)     |           |          |
razon_social  | character varying(100) |           | not null |
Indexes:
    "pk_proveedor" PRIMARY KEY, btree (id_proveedor)
Referenced by:
    TABLE "producto" CONSTRAINT "fk_proveedor" FOREIGN KEY (id_proveedor) REFERENCES proveedor(id_proveedor)
    TABLE "telefono" CONSTRAINT "fk_telproveedor" FOREIGN KEY (id_proveedor) REFERENCES proveedor(id_proveedor)

papeleria-# \d telefono;
Table "public.telefono"
  Column      |      Type      | Collation | Nullable | Default
-----+-----+-----+-----+-----
id_proveedor  | numeric(4,0)     |           | not null |
telefono      | numeric(10,0)    |           | not null |
Indexes:
    "pk_telefono" PRIMARY KEY, btree (telefono)
Foreign-key constraints:
    "fk_telproveedor" FOREIGN KEY (id_proveedor) REFERENCES proveedor(id_proveedor)

papeleria-# \d cliente;
Table "public.cliente"
  Column      |      Type      | Collation | Nullable | Default
-----+-----+-----+-----+-----
id_cliente    | numeric(4,0)     |           | not null |
nombre        | character varying(100) |           | not null |
apellido      | character varying(100) |           | not null |
estado        | character varying(32) |           |          |
cod_postal    | character(5)      |           |          |
colonia       | character varying(32) |           |          |
calle         | character varying(32) |           |          |
num           | numeric(4,0)     |           |          |
rfc           | character varying(13) |           | not null |
Indexes:
    "pk_cliente" PRIMARY KEY, btree (id_cliente)
Referenced by:
    TABLE "email" CONSTRAINT "fk_emailclien" FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente)
    TABLE "venta" CONSTRAINT "fk_venciente" FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente)

papeleria-# \d email;
Table "public.email"
  Column      |      Type      | Collation | Nullable | Default
-----+-----+-----+-----+-----
id_cliente    | numeric(4,0)     |           | not null |
email         | character varying(150) |           | not null |
Indexes:
    "pk_email" PRIMARY KEY, btree (email)
Foreign-key constraints:
    "fk_emailclien" FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente)

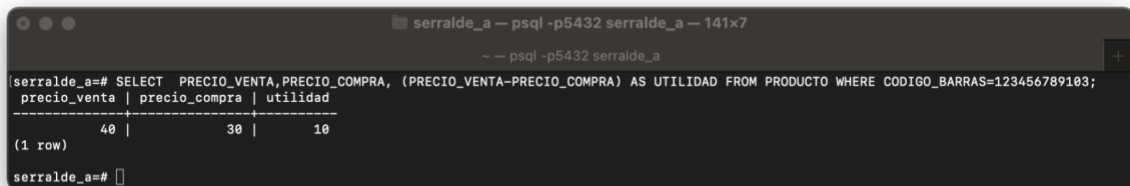
papeleria-# \d venta;
Table "public.venta"
  Column      |      Type      | Collation | Nullable | Default
-----+-----+-----+-----+-----
num_venta     | character(8)     |           | not null |
id_cliente    | numeric(4,0)     |           | not null |
codigo_barras | numeric(12,0)    |           | not null |
fecha_venta   | date             |           | not null |
cantidad_pagar | numeric(10,0)    |           | not null |
Indexes:
    "pk_venta" PRIMARY KEY, btree (num_venta)
Foreign-key constraints:
    "fk_venciente" FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente)
    "fk_venprodcuto" FOREIGN KEY (codigo_barras) REFERENCES producto(codigo_barras)
```

Presentación

Descripción de lo que hace la modalidad seleccionada como forma de conexión hacia la base de datos.

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

```
CREATE OR REPLACE FUNCTION UTILIPRODUC(V_CODIGO_BARRAS NUMERIC (13)) RETURNS TEXT AS $$  
  
BEGIN  
  
SELECT *, (PRECIO_VENTA-PRECIO_COMPRA) AS UTILIDAD FROM PRODUCTO WHERE  
CODIGO_BARRAS=V_CODIGO_BARRAS;  
  
END;  
  
$$ LANGUAGE PLPGSQL;  
  
SELECT PRECIO_VENTA,PRECIO_COMPRA, (PRECIO_VENTA-PRECIO_COMPRA) AS UTILIDAD FROM PRODUCTO WHERE  
CODIGO_BARRAS=123456789103;  
  
SELECT PRECIO_VENTA,PRECIO_COMPRA, (PRECIO_VENTA-PRECIO_COMPRA) AS UTILIDAD FROM PRODUCTO WHERE  
CODIGO_BARRAS=123456789103;
```



The screenshot shows a terminal window titled "serralde_a — psql -p5432 serralde_a — 141x7". The prompt is "serralde_a=#". The user enters the query: `SELECT PRECIO_VENTA,PRECIO_COMPRA, (PRECIO_VENTA-PRECIO_COMPRA) AS UTILIDAD FROM PRODUCTO WHERE CODIGO_BARRAS=123456789103;`. The output is displayed as a table with three columns: `precio_venta`, `precio_compra`, and `utilidad`. The values are 40, 30, and 10 respectively. The prompt returns to "serralde_a=#".

precio_venta	precio_compra	utilidad
40	30	10

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

```
def venta(idc, cod):
    conn=psycopg2.connect(
        dbname="papeleria",
        user="postgres",
        password="postgres",
        host="localhost",
        port="5432")
    cursor=conn.cursor()
    query='''INSERT INTO VENTA VALUES(%s,%s,%s,CURRENT_DATE,%s)'''
    query2=f'''SELECT PRECIO_VENTA FROM PRODUCTO WHERE codigo_barras={cod}'''
    query3='''SELECT COUNT(*) FROM VENTA'''
    query4=f'''SELECT CANTIDAD_EJEM FROM PRODUCTO WHERE codigo_barras={cod}'''
    cursor.execute(query3)
    row=cursor.fetchone()
    nextnv=int((str(row)[1:-2]))+1
    if nextnv<10:
        snextnv=f"VENT-00{nextnv}"
    elif nextnv<100:
        snextnv=f"VENT-0{nextnv}"
    elif nextnv<1000:
        snextnv=f"VENT-{nextnv}"
    else:
        print("La base de datos esta llena\n")
    cursor.execute(query4)
    row=cursor.fetchone()
    ncantidad=(int(str(row)[10:-4]))-1
    if ncantidad<0:
        messagebox.showerror("ERROR", "Ya no hay stock!")
    else:
        if ncantidad<3:
            messagebox.showwarning("ERROR", f"Con esta venta quedarían {ncantidad} artículos!")
        query5=f'''UPDATE PRODUCTO SET CANTIDAD_EJEM={ncantidad} WHERE codigo_barras={cod}'''
        cursor.execute(query5)
        cursor.execute(query2)
        row=cursor.fetchone()
        cursor.execute(query,(snextnv, idc, cod, str(row)[10:-4]))
        messagebox.showinfo("Venta", "Compra registrada!")
    conn.commit()
    conn.close()
    display_producto()
    display_venta()
```

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

```
serralde_a=# SELECT * FROM VENTA ORDER BY FECHA_VENTA;
num_venta | id_cliente | codigo_barras | fecha_venta | cantidad_pagar
-----
VENT-006 | 2 | 962496942016 | 2020-09-23 | 2765
VENT-003 | 1 | 487822308085 | 2020-10-02 | 4153
VENT-011 | 4 | 943548825401 | 2021-01-20 | 2407
VENT-010 | 4 | 219981320258 | 2021-01-31 | 200
VENT-007 | 3 | 161396493145 | 2021-03-29 | 2200
VENT-002 | 1 | 236544955187 | 2021-04-06 | 4142
VENT-012 | 1 | 236544955187 | 2021-04-07 | 4142
VENT-008 | 3 | 537724750452 | 2021-06-02 | 2100
VENT-005 | 2 | 122476609819 | 2021-08-28 | 2765
VENT-001 | 1 | 123456789103 | 2021-09-08 | 40
VENT-004 | 2 | 13151543527 | 2021-12-19 | 2765
VENT-009 | 4 | 794739366684 | 2022-02-19 | 1300
(12 rows)

serralde_a=# SELECT * from venta WHERE fecha_venta BETWEEN '2020-09-23' AND '2021-04-07' ORDER BY FECHA_VENTA;
num_venta | id_cliente | codigo_barras | fecha_venta | cantidad_pagar
-----
VENT-006 | 2 | 962496942016 | 2020-09-23 | 2765
VENT-003 | 1 | 487822308085 | 2020-10-02 | 4153
VENT-011 | 4 | 943548825401 | 2021-01-20 | 2407
VENT-010 | 4 | 219981320258 | 2021-01-31 | 200
VENT-007 | 3 | 161396493145 | 2021-03-29 | 2200
VENT-002 | 1 | 236544955187 | 2021-04-06 | 4142
VENT-012 | 1 | 236544955187 | 2021-04-07 | 4142
(7 rows)

serralde_a=# SELECT SUM (CANTIDAD_PAGAR) from venta WHERE fecha_venta BETWEEN '2020-09-23' AND '2021-04-07';
sum
-----
20009
(1 row)

serralde_a=#
```

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

```
SELECT NOMBRE FROM PRODUCTO WHERE CANTIDAD_EJEM >3;
```

```
serralde_a -- psql -p5432 serralde_a -- 118x30
-- psql -p5432 serralde_a

serralde_a=# SELECT NOMBRE, CANTIDAD_EJEM FROM PRODUCTO;
      nombre      | cantidad_ejem
-----+-----
 PLUMA_NEGRA      |           100
 TABLA_SURFF      |             5
 MESA_NEGRA       |             8
 LEGO_AVENGERS    |             6
 LEGO_HARVOTTER   |             2
 LEGO_FRIENDS     |             3
 LLANTAS_BICI     |            14
 AIRPODS          |             5
 CHARM            |            10
 CALCETAS         |            40
 CHARM            |             8
(11 rows)

serralde_a=# SELECT NOMBRE FROM PRODUCTO WHERE CANTIDAD_EJEM >3;
      nombre
-----
 PLUMA_NEGRA
 TABLA_SURFF
 MESA_NEGRA
 LEGO_AVENGERS
 LLANTAS_BICI
 AIRPODS
 CHARM
 CALCETAS
 CHARM
(9 rows)
```

Mostramos la interfaz grafica con la que enlazamos la base de datos para que el proyecto sea mas amigable con el usuario y este podrá realizar las funciones solicitadas para el proyecto.

SISTEMA DE LA PAPELERIA

Registrar producto

Codigo de barras:

Precio Compra:

Precio Venta:

Fecha:

Cantidad:

Nombre:

Marca:

Descripcion:

Proveedor:

Guardar producto

Registrar proveedor

ID Proveedor:

Nombre:

Apellido:

Estado:

Cod Postal:

Colonia:

Calle:

Numero:

Razon Social:

Telefono 1:

Telefono 2:

Guardar proveedor

Registrar cliente

ID Cliente:

Nombre:

Apellido:

Estado:

Cod Postal:

Colonia:

Calle:

Numero:

RFC:

Email:

Email2:

Guardar cliente

Venta

Num Venta:

ID Cliente:

Codigo de barras:

Fecha:

Cantidad:

Vender

CodigoBarras	Nombre	Precio	Stock
123123123123	Lapiz 60 0		
123123123456	Goma 25 2		

ID	Apellido	Nombre	Razon Social	Telefono
1	Perez	Juan	Lapices corp	1 5545 789865

ID	Apellido	Nombre	RFC	EMAIL
1	Serralde	Andrea	SEFA991891	1 andy@gmail.com

Numero	CodigoBarras	Cliente	Fecha	Precio
VENT-001	123123123123	1	2021-08-15	60
VENT-002	123123123123	1	2021-08-15	60
VENT-003	123123123123	1	2021-08-15	60
VENT-004	123123123123	1	2021-08-15	60

Borrar producto:

Borrar proveedor:

Borrar cliente:

Borrar venta:

Calcular utilidad:

Ganancia del periodo:

Pocos productos:

Interfaz de trabajo (PAPELERIA)

Conclusiones

Calderón González José Gabriel: En este proyecto considero que implementamos correctamente todos los temas vistos durante el curso, además de otros nuevos que se tuvieron que investigar en el caso de las consultas, considero que los roles de cada integrante los asignamos correctamente y que la comunicación fue lo más importante, personalmente desarrolle una aplicación gráfica para la base y aprendí mucho de cómo dar formato a las vistas e inserciones ya que en la vida real no todos son programadores y se tiene que entregar proyectos con interfaces amigables para que cualquier usuario pueda utilizar la base sin problemas.

Ceballos Ricardo Fernando: Realizamos este proyecto de forma que revisamos lo aprendido a lo largo del semestre, la parte de consultas fue la más complicada incluyendo la conexión a parte gráfica ya que no habíamos realizado anteriormente algo de este estilo por lo que llevó más tiempo del planeado para conseguirlo de forma correcta, sin embargo, se pudo conseguir el objetivo del proyecto.

Hernández Bacilio Jesús Emilio :Concluyo que este proyecto, tuvo sus problemas al momento de la realización, sin embargo, pudimos completar los objetivos del mismo, el uso de las consultas por medio de PL costaron más trabajo del pensado al igual que la conexión con la herramienta gráfica, aún con estas complicaciones se realizó el objetivo del proyecto.

Serralde Flores Andrea: Analizamos que para ejecutar la base de datos, la creación de consultas a la base de datos consiste en archivos que permiten el uso de datos visibles para realizar muchas tareas diferentes. También se pueden utilizar para controlar los registros mostrados Las consultas de la base de datos no contienen información de la base de datos, sino solo las instrucciones necesarias para seleccionar los registros y campos requeridos de la base de datos. Algunos aspectos importantes aprendidos son la definición, requisitos, ventajas y características de la base de datos, podemos decir que la base de datos: es una colección de datos o información que se utiliza para brindar servicios para muchas aplicaciones al mismo tiempo. Finalizando podemos decir que las bases de datos forman el núcleo de las principales aplicaciones, sitio web y servicios corporativos.

Bibliografía

- <https://www.postgresql.org/message->

- id/attachment/92321/pl_pgsql_y_otros_lenguajes_procedurales_en_postgresql.pdf
<https://www.tutorialesprogramacionya.com/postgresqlya/>