UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
Facultad de Ingeniería
Ingeniería en Computación

Bases de Datos
grupo: 01

Proyecto final

Alumnos:

- Fonseca Huitrón Julise
- López Aniceto Saúl Isaac
- López González Kevin
- Martínez Vázquez Diego
- Ponce Soriano Armando

Profesor:
ING. Fernando Arreola Franco

8 de diciembre de 2021

Índice

1. Introducción

Este proyecto consiste en diseñar y crear una base de datos que utilizará una cadena de papelerías en la que se puede manipular y almacenar la información de todos los productos que esta cadena ofrece. La base de datos tendrá como finalidad llevar de manera ordenada y controlada la logística que la papelería sigue para tener control de todos los servicios que ofrece, así como llevar un control de las transacciones y accesos. Este proyecto también contempla la creación de una página web para que los usuarios de esta, tengan acceso a funciones que se desarrollan en la base de datos y se pueda ver de una forma gráfica. El desarrollo de esta base de datos será detallado en el presente documento con la finalidad de explicar los procedimientos, planes y metodologías que se fueron utilizadas para llevar a cabo este proyecto. Este proyecto fue elaborado en POSTGRESQL, utilizando los recursos de PHP y Amazon Web Services.

2. Plan de trabajo

2.1. Descripción

El plan de trabajo utilizado para el desarrollo de este proyecto consiste, principalmente, en asignar tareas específicas a todos los participantes, con base en el establecimiento de metas y fechas.

2.2. Plan de actividades

Se desarrolla un cronograma estableciendo cada una de las fases que involucra la creación y desarrollo del sistema, asignando fechas y un límite de tiempo para cada tarea. En la parte de asignaciones, se asigna a cada uno de los integrantes una tarea referida en el cronograma y que forma parte de la elaboración de este proyecto.

2.3. Cronograma

2.4. Aportaciones

	Diseño	Implementación	Presentación	Acoplamiento	Documentación
Kevin López	✓		✓	✓	✓

3. Diseño

3.1. Análisis de requerimientos

Para la elaboración de la base de datos se contempla que una papelería tendrá una serie de registros que a su vez tendrán información, se requiere almacenar información de los proveedores, se requiere almacenar información sobre un Inventario que almacena también información. Se deberá guardar información detallada de cada venta realizada. Requerimos también toda la información relacionada con el cliente y también información específica del o los tipos de producto que tenemos a la venta. Los requerimientos antes mencionados nos llevan a conceptualizar más las ideas, encontrando que hay dependencias en algunos de ellos, nos lleva a analizar la cardinalidad que hay entre las tablas. El análisis de estos requerimientos es la base para la construcción de nuestros modelo EntidadRelación y en general de todo el modelo Conceptual.

3.2. Modelo conceptual

Gracias al análisis de requerimientos hecho anteriormente, pudimos llevara a cabo el modelo conceptual, donde obtuvimos de manera más clara las entidades y las relaciones que hay para realizar el Modelo Entidad-Relación.

Entidades

- PROVEEDOR: { id_Proveedor, razón social, domicilio (estado, código postal, colonia, calle y número), nombre, teléfonos }
- CLIENTE: {RFC, nomre (nombre, ap_Paterno, ap_Materno), domicilio (estado, código postal, colonia, calle y número), emails }
- INVENTARIO: {id_Inventario, precio_compra, fecha_compra, cantidad_ejemplares }
- PRODUCTO: {código_Barras, marca, descripción, precio, categoria}
- VENTA : {num_venta, fecha_venta, pago_Total, cantidad_articulo, pago_total_Articulo }

Relaciones

- Un proveedor surte a muchos inventarios.
- Un inventario es surtido por muchos proveedores.
- Un inventario almacena muchos productos.
- Un producto es almacenado por un inventario.

- Una venta contiene muchos productos.
- Un producto es contenido es muchas ventas.

- Un cliente concreta muchas ventas.
- Una venta es concretada por un cliente.

3.2.1. Modelo Entidad-Relación

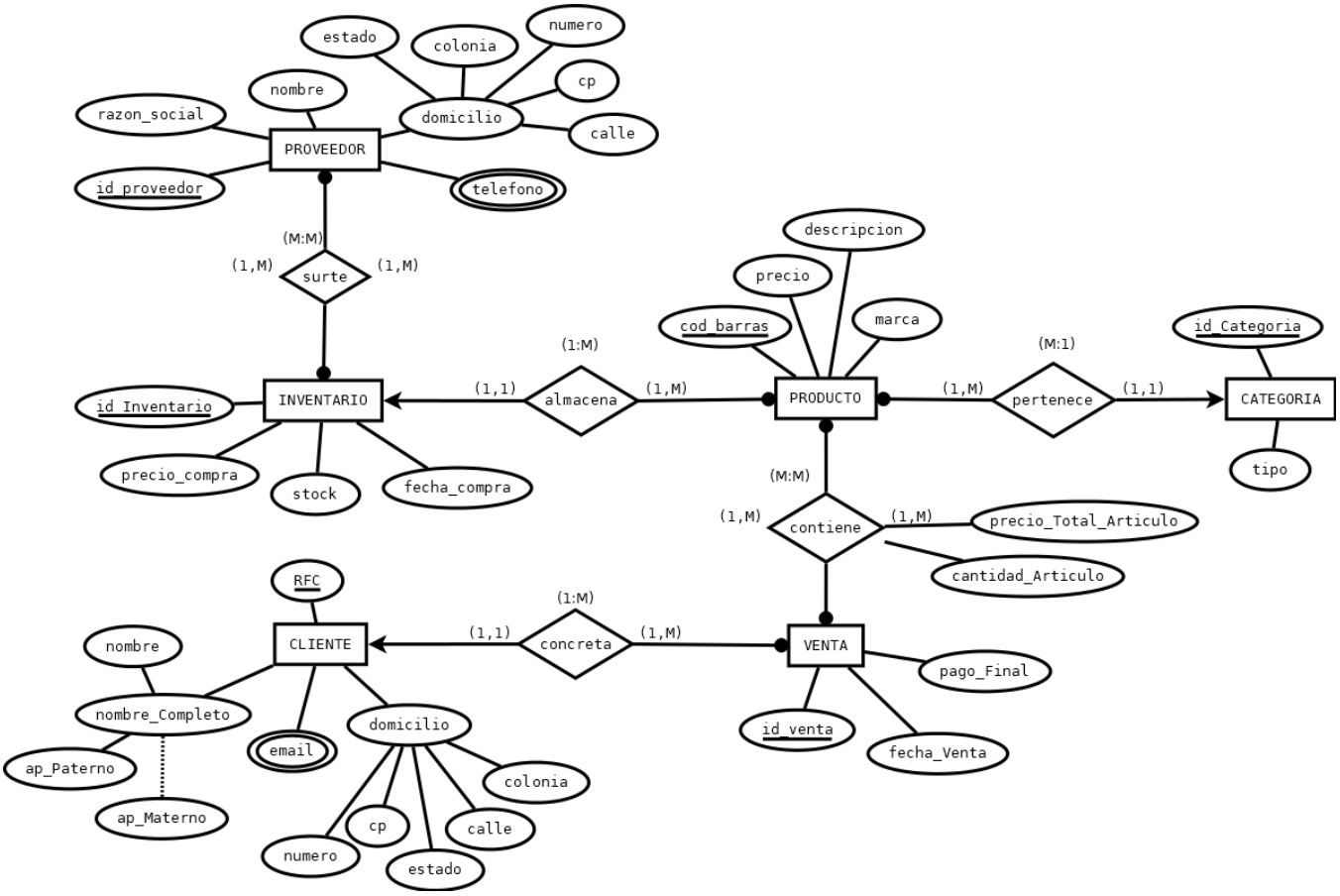

Figura 1: Modelo Entidad-Relación.

3.3. Modelo lógico

3.3.1. Representación Intermedia

Realizando el Modelo Entidad-Relación, pudimos seguir las reglas para de la transformación de MER a MR, nos derivó en las siguientes tablas:

- PROVEEDOR: { id_proveedor smallint (PK), nombre varchar 50, razón social varchar 50, estado varchar 50, colonia varchar 50, numero smallint, cp smallint, calle varchar 50}
- TELEFONO: {teléfono bigint(PK), id_proveedor smallint (FK)}

- INVENTARIO: {id_Inventario smallint (PK), precio_compra decimal (10,2), stock smallint, fecha_compra date }
- SURTE: {[id_Provedor smaillint (FK), id_Inventario smallint (FK)] (PK)}
- PRODUCTO: {cod_barras integer PK, id_categoria smallint FK, precio smallint NOT NULL, marca varchar(20) NOT NULL, descripcion varchar(50), id_inventario smallint (FK)}
- CATEGORÍA: { id_categoria smallint PK, tipo varchar(20) NOT NULL}
- CLIENTE: {RFC varchar(13) (PK), nombre varchar(20), ap_paterno varchar (20), ap_materno varchar (20) (N), cp smallint, numero smallint, estado varchar (32), calle varchar (32), colonia varchar (32)}
- EMAIL: {RFC varchar(13) (FK), email varchar (64) (PK)}
- VENTA: {id_venta int(PK), fecha_venta date, pago_final decimal(7,2), RFC varchar(13)(FK)}
- CONTIENE: { [cod_barras int , id_venta int](PK)(FK), precioTotalArt decimal(7,2), cantidad articulo int}

3.3.2. Transformación de MER a MR

Para la transformación de Modelo Entidad Relación a Modelo Relacional, primero debemos utilizar la transformación intermedia que obtuvimos y desarrollamos. Una vez finalizado ese procedimiento utilizamos las reglas para la trasformación de MER a MR y obtendremos nuestro modelo relacional.

3.3.3. Modelo Relacional

3.3.4. Normalización

4. Implementación

Para llevar a cabo la implementación de esta base de datos, primero tendremos que crear la base de datos en el manejador POSTGRESQL.8 Una vez finalizada la creación de la base de datos, tendremos que crear las tablas que obtuvimos en nuestro modelo relacional. Estas tablas tendrán que tener sus respectivos constraints para las llevas primarias y foráneas que se utilizan en toda la base de datos. Cuando las tablas estén creadas en las base de datos, podremos verificar su utilidad haciendo inserciones de prueba. Para que el sistema resuelva cuando se reciba un código de barras regrese la utilidad, para que cuando haya la venta de un artículo y se decremente el stock por la cantidad vendida, dada una fecha o una fecha de inicio y de fin, regrese la cantidad total que se vendió y su ganancia correspondiente. Para obtener el nombre de aquellos productos que están es stock, que de forma automática genere vista que contenga información para desplegar una factura de compra y la creación del índice. Para todas esas funciones mencionadas, tendremos que hacer uso de la creación de Functions, Procedures y Triggers.

4.1. Modelo físico

Para el desarrollo de la base de datos y su implementación, tendremos que trabajar con la creación de tablas y sus respectivas inserciones en el manejador, tendremos que utilizar la programación en base de datos para que se logren hacer las funciones que se nos solicitan. Lo primero que realizamos fue la creación de tablas, que a su vez son las mismas que obtuvimos del Modelo Relacional, las tablas que creamos son las que se ilustran a continuación:

También para comprobar que la creación de estas tablas fue correcta, tuvimos que realizar inserciones que tuvieran relación con estas tablas. Las inserciones que realizamos son las siguientes:

Para lograr obtener las funciones solicitadas en los requerimientos, tendremos que utilizar la programación en base de datos. Lo que realizamos fue para cada uno de los puntos que tenemos en los requerimientos. Cabe aclarar que estas funciones, luego tendrán que ir acompañadas de triggers, para que al momento de usar estas funciones, sea de manera inmediata. Función que retorna la utilidad. - El primer requerimiento de las funcione nos pide que al recibir un código de barras de un producto, este regrese la utilidad, en este caso la utilidad represente el precio al que el producto es ofertado menos el precio al que el producto fue adquirido al proveedor.

Las siguientes Functions, representan al requerimiento que nos solicita dada una fecha, o fecha de inicio y fecha de fin se nos solicita regresar la cantidad del total que se vendió y la ganancia correspondiente.

Función para retornar pagos finales

Función para retornar pagos finales entre fechas

Función para retornar las ganancias

Función para retornar las ganancias por fechas

4.1.1. IaaS

4.2. Códigos

4.3. DDL

5. Presentación

Una interfaz gráfica es un programa que nos permite manipular información a través de objetos gráficos que proporcionen un entorno visual, con el fin facilitar la interacción del usuario con la computadora.

En este caso, se optó por desarrollar una página web como interfaz gráfica que permita la manipulación de información de nuestra base de datos.

5.1. Django

Django es un framework de Python de alto nivel que permite diseñar aplicaciones web de una forma rápida, limpia y pragmática. Además, ayuda a los desarrolladores a evitar muchos errores de seguridad comunes, como la inyección de SQL, las secuencias de comandos entre sitios, la falsificación de solicitudes entre sitios y el secuestro de clics.

5.1.1. Mapeo Relacional de Objetos

El Mapeo Relacional de Objetos o ORM (*Object Relational Mapping*), es una tecnología que soluciona el desajuste entre las bases de datos relacionales y orientadas a objetos.

Por lo general, asigna una clase a una tabla uno a uno. Cada instancia de la clase corresponde a un registro en la tabla y cada atributo de la clase corresponde a cada campo en la tabla. ORM proporciona una asignación a la base de datos, en lugar de escribir código SQL directamente, solo es necesario manipular los datos de la base de datos como un objeto operativo.

Si bien, el ORM es una herrmienta muy util, no se utilizará en este proyecto, ya que preferimos escribir directamente la sentencia SQL para comunicarnos con la base de datos.

5.1.2. Ejecutar SQL personalizado directamente

El objeto **django.db.connection** representa la conexión por defecto entre django y la base de datos. Las funciones que utilizamos para la comunicación con la base de datos son las siguientes:

- **connection.cursor()**

  Para obtener un objeto cursor.

- **cursor.execute(sql, [params])**

  Para ejecutar sentencias SQL.

- **cursor.fetchall()**

  Para devolver las filas resultantes de la consulta.

5.2. Diseño

6. Conclusiones

-

- López González Kevin

  Bla bla bla

-