Estrategia

TP Gestión de Datos 1C 2024

Grupo: ALBONDIGA

Número: 11

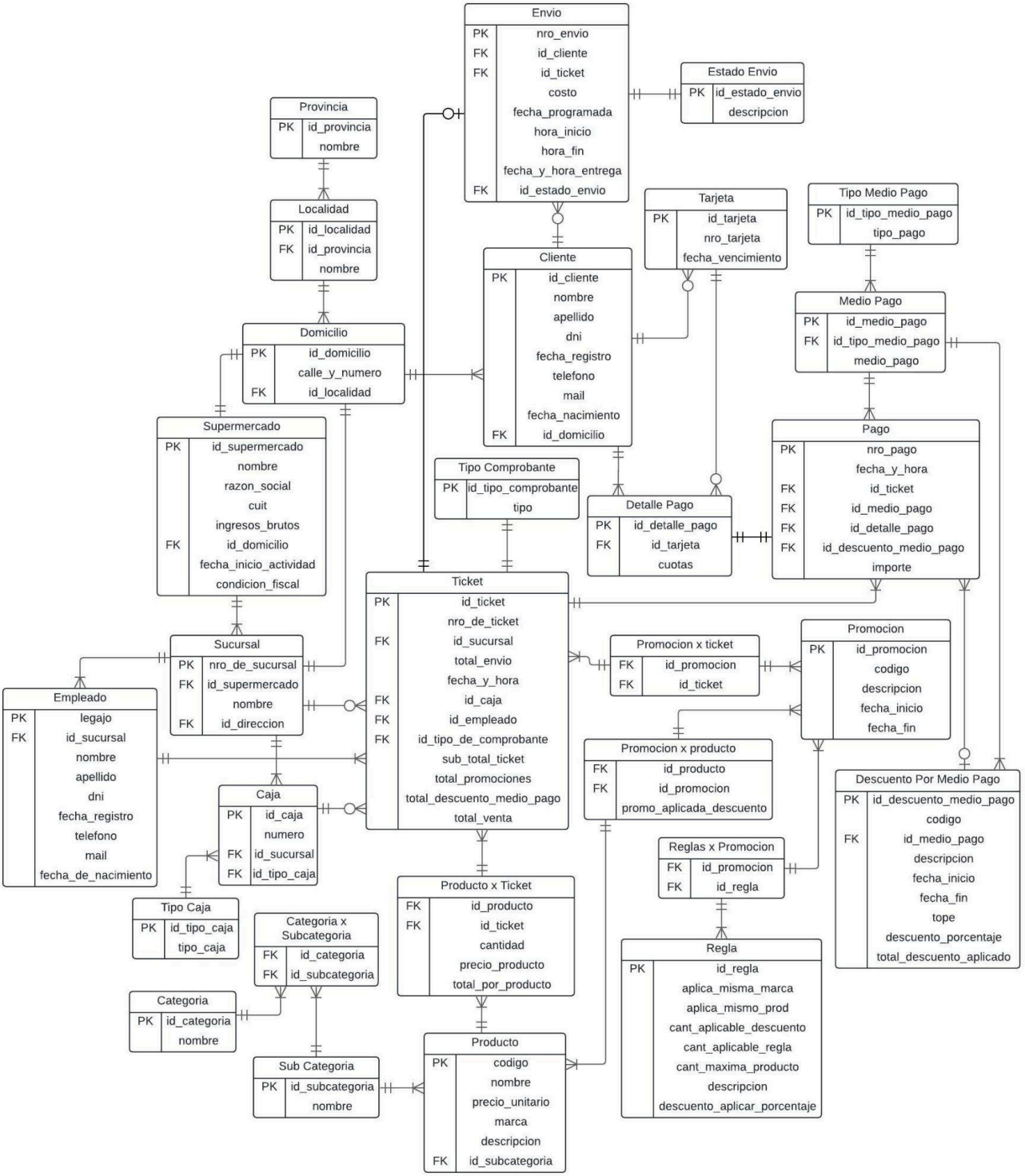
Integrantes

- Facundo Azula (167.975-2)
  Curso: K3151
- Juan Pablo Aveni (167.895-4)
  Curso: K3052
- Camila Garcia Federico (168.258-1)
  Curso: K3151

Índice

Entrega 1: DER

Modelo de datos



Estrategias

En nuestro DER tomamos las siguientes decisiones:

1. Ticket

Se crea esta tabla para almacenar información sobre los tickets generados por una venta, este mismo guarda el campo **id_ticket** el cual consideramos utilizar como **PK** que es un valor autoincremental, también guardamos **nro_de_ticket** que está en la tabla maestra. Otro campo que guardamos es **id_sucursal** el cual es una **FK** que hace referencia al identificador único de la sucursal donde se realizó la venta, permite asociar el ticket con la sucursal correspondiente en la base de datos. También guardamos **total_envio** para poder calcular el total del ticket.

Sucursal: Esta tabla almacena información sobre las sucursales registradas en el sistema. Utiliza el campo **nro_de_sucursal** como **PK**, obtenido de la tabla maestra ya que es único. Además, guarda el nombre de la sucursal y utiliza una **FK** a **id_direccion**, relacionándose así con la tabla Domicilio para registrar su ubicación. Cada sucursal estará asociada a un Supermercado, por ello creamos la **FK id_supermercado.**

Supermercado: Creamos esta tabla para almacenar el **nombre**, **razon_social**, **cuit**, **ingresos_brutos**, **fecha_inicio_actividad**, **condicion_fiscal** que son todos campos básicos que obtenemos de la tabla maestra de los supermercados. Luego también tenemos una **FK** a la tabla Domicilio el cual guardará la dirección del mismo. Un supermercado tendrá varias sucursales, por eso hicimos esta abstracción. La **PK** de está tabla, denominada **id_supermercado**, es autoincremental ya que no disponíamos de una clave candidata.

El campo **id_caja** es una **FK** la cual hace referencia a la tabla Caja para dejar guardado en qué caja fue generado el ticket.

Caja: Tabla que almacena información sobre las cajas de cada sucursal, por eso mismo tenemos la **FK id_sucursal**, para saber a cual pertenece. La **PK** de está tabla, denominada **id_caja**, es autoincremental ya que no disponíamos de una clave candidata. A su vez guardamos su **id_tipo_caja** como **FK**, que es a su vez la **PK** de la tabla Tipo Caja que además de la **PK** antes mencionada, también almacena el **tipo_caja** de caja, que puede ser Prioridad, Rápida o Envío.
El campo **id_empleado** es una **FK** que guarda qué empleado generó el ticket o se asocia la venta.

Empleado: Tabla que almacena todos los datos de los empleados de las diferentes sucursales, por ello guardamos en esta tabla la **FK id_sucursal**. Cada empleado tendrá un **legajo** que será único, obtenido de la tabla maestra y que utilizaremos como **PK** de la misma. También guardamos los datos básicos de los empleados en los campos **nombre, apellido, dni, fecha_registro, telefono, mail y fecha_de_nacimiento.**
El campo **id_tipo_de_comprobante** es una **FK** que relacionamos con la tabla Tipo Comprobante.

Tipo Comprobante: Esta tabla fue creada para guardar los tipos de comprobantes, los cuales pueden variar entre A, B o C. La **PK** de está tabla, denominada **id_tipo_comprobante**, es autoincremental ya que no disponíamos de una clave candidata.

Producto x Ticket: Sirve para registrar todos los productos vendidos incluidos en el ticket. Esta relación permite detallar la cantidad vendida de cada producto. Dado que un ticket puede contener múltiples productos y un producto puede estar presente en varios tickets, se optó por crear esta tabla intermedia para gestionar esta relación muchos a muchos.
Por último, tenemos los campos numéricos **sub_total_ticket**, **total_promociones**, **total_descuento_medio_pago**, **total_venta** y

precio_producto los cuales son valores ya calculados obtenidos de la tabla maestra.
Promocion x ticket: Esta tabla fue creada con la intención de romper la relación muchos a muchos que existe entre ticket y promoción tiene una primary key compuesta por las dos claves foráneas **id_promocion** y **id_ticket**

2. Pago

Se crea esta tabla para almacenar información sobre los pagos generados por cada ticket, ya que puede haber dos pagos con sus respectivos descuentos por cada compra. Ya que un ticket puede relacionarse con más de un pago es que guardamos el campo **nro_pago** como **PK** que es un valor incremental único para cada pago, y **id_ticket** como clave foránea para poder relacionar el ticket con sus correspondientes pagos. A su vez también guardamos como claves foráneas **id_medio_pago**, **id_detalle_pago** y **id_descuento_medio_pago** siendo estas últimas guardadas ya que debemos conocer las condiciones de las posibles promociones vigentes y a su vez la forma de pago para poder analizar la elegibilidad de las mismas.

Medio Pago: Esta tabla almacena información sobre los medios de pago registrados en el sistema, utiliza **id_medio_pago** como su **PK**, a su vez contiene a **id_tipo_medio_pago** como **FK** para así conocer la forma general que tiene el medio de pago, o bien el Tipo Medio Pago (si es una billetera virtual, una tarjeta de débito, etc)

Detalle Pago: Esta tabla almacena información detallada en caso de que se haga un pago con tarjeta en cuotas, la tabla contiene **id_detalle_pago** como **PK** que es un valor autoincremental ya que no disponíamos de una clave candidata, a su vez contiene como **FK** a **id_tarjeta** para relacionar el detalle con la tarjeta que se realizó el pago.

3. Envío

Se crea esta tabla para almacenar información de los envíos, el **nro_envio** es la clave principal de esta tabla la cual la obtenemos de la tabla maestra debido es un campo unívoco, contiene la llave foránea **id_cliente** que relaciona el envío con un Cliente, decidimos poner al cliente ya que el envío se hará al domicilio que tenga asociado y no a otro. Un Cliente tiene asociado una dirección el cual lo relaciona con la tabla Domicilio.

Domicilio: Esta tabla guarda la calle_y_numero de la dirección y también una FK a id_localidad, el cual relaciona esta tabla con la tabla Localidad. La **PK** de está tabla, denominada **id_domicilio**, es autoincremental ya que no disponíamos de una clave candidata.

Localidad: Esta tabla guarda el nombre de la localidad y una FK a id_provincia, el cual relaciona esta tabla con una Provincia. Decidimos abstraerlo de Domicilio ya que al crear una tabla aparte para Localidad y relacionarla con Provincia, podemos mantener la integridad de los datos, evitar la redundancia y simplificar la gestión de la información relacionada con las localidades en nuestro sistema. La **PK** de está tabla, denominada **id_localidad**, es autoincremental ya que no disponíamos de una clave candidata.

Provincia: Esta tabla guarda el nombre de la provincia. La **PK** de está tabla, denominada **id_provincia**, es autoincremental ya que no disponíamos de una clave candidata.

Otras llaves foráneas son **id_ticket** que relaciona al envío con la tabla Ticket y **id_estado_envio** esta tabla guarda información sobre los distintos estados en que puede estar un envío desde que se realiza hasta que se entrega.

Estado Envío: es una tabla que contiene los estados posibles de un envió. La **PK** de está tabla, denominada

id_estado_envio, es autoincremental ya que no disponíamos de una clave candidata.
Por último, tenemos los campos que tienen que ver con la fecha y horario en el cual se entregará o se programará el envío, estos campos los obtenemos de la tabla maestra y son **fecha_programada**, **hora_inicio, hora_fin**, **fecha_y_hora_entrega**. También poseemos el **costo** del envío el cual será un valor numérico.

4. Cliente

Se crea esta tabla para persistir la información referente a los clientes, su clave primaria es el **id_cliente** que sirve para identificar un cliente de manera única, tiene la llave foránea **id_domicilio**.
Se relaciona con las tablas tarjeta, domicilio, envio y detalle pago.

Tarjeta: es una tabla que contiene la información de las tarjetas de los clientes, tiene como clave primaria el **id_tarjeta.**

5. Producto

En esta tabla se almacena la información de cada producto en específico, su primary key es **codigo**, tiene la foreign key **id_subcategoria**.
Se relaciona con las tablas Producto x Ticket, Promocion x producto y Sub Categoria (que esta a su vez se relaciona con Categoria x Subcategoria [que a su vez se relaciona con Categoria]).

Producto x Ticket: Esta tabla se usa para evitar la relación de muchos a muchos entre producto y ticket, su clave primaria es compuesta y son los campos **id_producto** y **id_ticket**. Se relaciona con producto y ticket

Promocion x producto: Esta tabla se usa para evitar la relación mucho a muchos entre Promocion y Producto, su clave primaria es compuesta y se trata de **id_producto** y

id_promocion. Se relaciona con Promoción y Producto

Sub Categoria: Es una tabla que se utiliza para registrar las subcategorías de un producto, su clave primaria es **id_subcategoria**, se relaciona con Producto y Categoria x Subcategoria

Categoria x Subcategoria: Esta tabla se utiliza para evitar la relación muchos a muchos su llave primaria es compuesta por los campos **id_categoria** y **id_subcategoria**, se relaciona con Sub Categoria y Categoria

Categoria: Esta tabla almacena las Categorías de los productos del supermercado su llave primaria es **id_categoria**, Se relaciona solo con Categoria x Subcategoria

6. Promoción

Se crea esta tabla para almacenar información sobre las promociones, el descuento que las mismas poseen, su tiempo de validez y las condiciones, o bien reglas, de las mismas. Esta tabla contiene el campo **id_promocion** como su **PK** es un autoincremental ya que no disponíamos de una clave candidata. Si bien esta tabla no contiene claves foráneas, su clave principal es utilizada por varias tablas como clave foránea:

Regla: Esta tabla contiene las bases y condiciones de las promociones que se encuentran en el sistema, la misma cuenta con **id_regla** como su **PK**, la cual es autoincremental ya que no disponíamos de una clave candidata. Estas 2 tablas se relacionan por medio de la tabla Regla x Promocion para evitar la relación de muchos a muchos, dicha tabla solo contiene las **PKs** de las 2 tablas que relaciona como **FKs**

Descuento Por Medio Pago: Esta tabla contiene la información sobre las promociones que cuentan como origen la forma de medio de pago y el total del descuento

aplicado. Cuenta con **id_descuento_medio_pago** como su **PK** la cual es autoincremental ya que no disponíamos de una clave candidata, y con **id_medio_pago** como su **FK** para poder establecer relaciones entre su respectivo descuento dependiendo el medio de pago elegido por el cliente en el pago correspondiente.

Entrega 2: Migración de datos

Migración

Creamos el script de migración, donde primero utilizamos el comando DROP para eliminar datos si es que existen. Utilizamos un DROP para todas las tablas, stored procedures y el esquema. Posteriormente realizamos la creación del esquema y de las tablas.

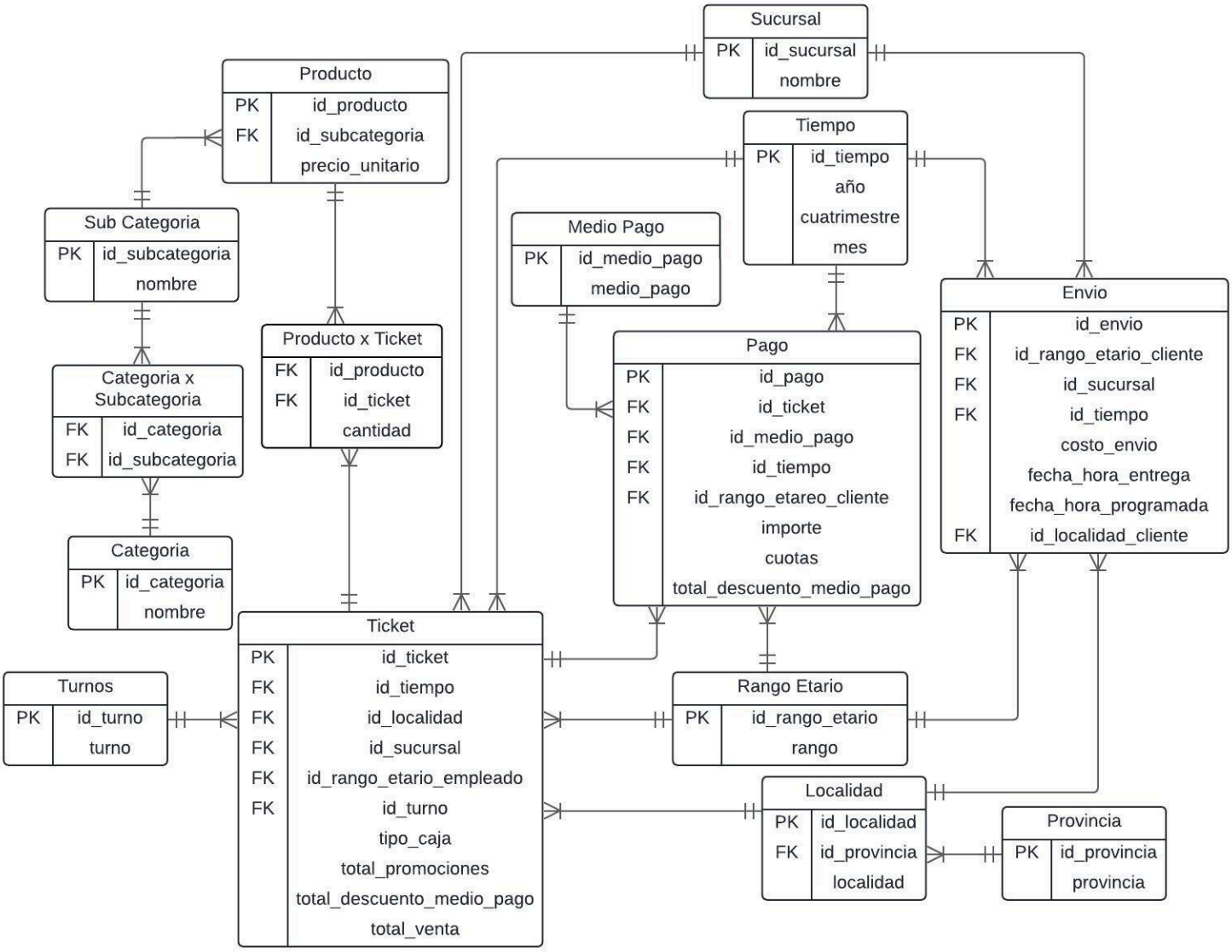
Procedimos con la creación de las tablas necesarias teniendo en cuenta los tipos de datos que ya se utilizaban en la tabla maestra para poder realizar la migración. Utilizamos como PKs valores que se encontraban en la tabla maestra donde pudimos y donde no resultamos a utilizar valores autoincrementales para identificar de forma inequívoca a todos nuestros registros.
Para las FKs utilizamos CONSTRAINTS después de la creación de las tablas para poder referenciar las tablas correspondientes. Realizamos un CONSTRAINT y un ALTER TABLE por cada FK que agregamos.

A continuación creamos los stored procedures, que necesitamos uno por cada tabla que queremos migrar. Tuvimos en cuenta el hecho de no migrar los valores que eran NULL y de que las FKs o PKs tampoco puedan ser NULL.

Por último se ejecutan todos los stored procedures imprimiendo un mensaje como “Se comienza a migrar ...” para poder tener claro cuando se está migrando cada tabla.

Entrega 3: Modelo y Migración BI

Modelo BI



Estrategias

Para esta entrega, creamos tanto las tablas dimensionales como las tablas de hechos. Las **tablas de hechos** que nos quedaron son **BI_Envio, BI_Ticket y BI_Pago** ya que son las tablas que contienen datos calculados o datos importantes para obtener las métricas.
En esta instancia se crearon tablas para guardar solo los datos que necesitamos para nuestro modelo BI, a continuación se explican las tablas creadas.

1. BI_Tiempo

Se crea esta tabla **dimensional** para almacenar información sobre el tiempo, tiene como **PK** la columna id_tiempo y se encuentra como **FK** en las tablas de BI_Ticket y BI_Envio

2. BI_Turno

Se crea esta tabla **dimensional** para almacenar información sobre la duración del turno, tiene como **PK** la columna id_turno y se encuentra como **FK** en la tabla de BI_Ticket.
Se creó un nuevo rango además de los que se especificaron en el enunciado para que caiga dentro de este si es que no coincide con ninguno de los rangos anteriores, este se llama “Fuera de rango”

3. BI_Rango_Etario

Se crea esta tabla **dimensional** para almacenar información sobre rangos de edad de personas, tiene como **PK** la columna id_rango_etario y se encuentra como **FK** en las tablas de BI_Ticket y BI_Envio.
Se creó un nuevo rango además de los que se especificaron en el enunciado para que caiga dentro de este si es que no

coincide con ninguno de los rangos anteriores, este se llama “Desconocido”

4. BI_Producto

Se crea esta tabla **dimensional** para almacenar información sobre los precios del producto, tiene como **PK** la columna id_producto y tiene una **FK** id_subcategoria, proveniente de la tabla BI_SubCategoria

5. BI_SubCategoria

Se crea esta tabla para persistir información correspondiente a la subcategoría tiene una **PK** sobre la columna id_subcategoria

6. BI_Categoria_x_Subcategoria

Se crea esta clase para evitar la relación muchos a muchos, contiene dos **FK** a las tablas BI_SubCategoria(id_subcategoria) y BI_Categoria(id_categoria)

7. BI_Categoria

Se crea esta tabla para persistir información correspondiente a la categoría tiene como **PK** la columna id_categoria

8. BI_Ticket

Es una de las **tablas de hechos** creada para almacenar información sobre los tickets, tiene como **PK** la columna id_ticket, luego contiene varias **FK** a varias tablas: BI_Tiempo(id_tiempo), BI_Localidad(id_localidad), BI_Sucursal(id_sucursal), BI_Rango_Etario(id_rango_etario_empleado), BI_Turno(id_turno)

9. BI_Localidad

Se creó esta tabla para persistir información correspondiente a la localidad, esta tabla tiene una **PK** en la columna id_localidad y luego una **FK** id_provincia haciendo referencia a la tabla BI_Provincia

10. BI_Provincia

Se creó esta tabla para persistir información correspondiente a la provincia, tiene una **PK** en la columna id_provincia

11. BI_Sucursal

Se creó esta tabla para persistir información correspondiente a la sucursal, tiene una **PK** en la columna id_sucursal

12. BI_Envio

Es una de las **tablas de hechos** creada para almacenar información sobre los envíos, tiene como **PK** a la columna id_envio, luego contiene varias **FK** a varias tablas: BI_Rango_Etario(id_rango_etario_cliente), BI_Sucursal(id_sucursal), BI_Tiempo(id_tiempo), BI_Localidad(id_localidad_cliente)

13. BI_Pago

Es una de las **tablas de hechos** creada para almacenar información sobre los pagos, tiene como **PK** a la columna id_pago, luego tiene varias **FK** a varias tablas, BI_Ticket(id_ticket), BI_Medio_Pago(id_medio_pago),BI_Tiempo(id_tiempo),BI_Rango_Etario(id_rango_etario_cliente)

14. BI_Medio_Pago

Se crea esta tabla para persistir información correspondiente al medio de pago, esta tabla tiene una **PK** en la columna id_medio_pago

15. BI_Producto_x_Ticket

Se crea esta tabla para evitar la relación muchos a muchos entre producto y ticket, contiene dos **FK** a las tablas BI_Producto(id_producto) y BI_Ticket(id_ticket)

Funciones

En esta instancia se crearon distintas funciones que nos servirán para guardar los datos en las tablas con el formato requerido para las vistas, a continuación se explican las funciones creadas.

edadActual: Toma una fecha de nacimiento con formato DATETIME y nos devuelve la edad en formato INT, esta función sola no se utiliza, se utiliza en conjunto con la del rango etario para, luego de obtener la edad, obtenemos el rango etario al que pertenece el usuario/repartidor/operador.
rangoEtario: Toma una edad en formato INT y nos devuelve el rango etario al que pertenece, lo decidimos poner al rango como un VARCHAR que nos muestre a qué rango etario pertenece según la edad que nos pasan por parámetro.
obtenerHora: Toma una fecha en formato DATETIME y nos devuelve solo la hora en formato INT.
rangoHorario: Toma una hora en formato INT y en base a esto nos devuelve el rango horario al que pertenece en formato VARCHAR.
obtenerCuatrimestre: Toma una fecha en formato DATETIME y nos devuelve a que cuatrimestre del año pertenece según el mes del año.