GESTIÓN DE DATOS

GRUPO N° 38	**CURSO: K3114**

PROFESOR: Marcelo mozcuzza

NOMBRE GRUPO: PEDIDOS_APP

ENTREGA N°: 1

TÍTULO: Entrega de modelo de datos y migración

INTEGRANTES PRESENTES EL DÍA QUE SE REALIZÓ	
Martín Spagnol	176546
Luciano Rodriguez	1771930
Gabriel Soares de Oliveira	2027770
Mateo Michini	1758160

RESPONSABLE: Martín Spagnol

Diagrama Entidad Relación

Usuario

Cada usuario es único, es por eso que lleva como atributo el DNI como PK (primary key) para poder ser identificado.

Usuario_Dirección

Cada puede tener registrado en el sistema varias direcciones que serán de utilidad para referenciarlas en el pedido. La misma cuenta con un código de ID propio como atributo PK para identificar unívocamente a las direcciones.

Medio de pago

Los usuarios registrados en el sistema deben contar con un medio de pago, estos deben realizarlo a través de un tipo de tarjeta donde el número de la misma será utilizado como PK en un atributo.

Cupones y sus tipos

Se modelaron dos ramas de cupones: cupón y cupón reclamo. Ambas entidades cuentan con un ID único q los identifica y encuentran relacionadas con USUARIO_X_CUPON que hace de intermediario entre CUPÓN y USUARIO, la misma cuenta con una PK con un id especifico.

En el caso de cupón, que no es cupón reclamo, lo encontramos relacionado al pedido mediante PEDIDO_X_CUPON la cual contiene a cupón y pedido como FK.

Luego para cupón reclamo contiene los datos propios del cupón y como FK al reclamo que se aplicó este cupón.

Reclamo y operador

Para identificar a los operadores utilizamos como PK OPERADOR_DNI. Esta entidad de operador la asociamos a la de Reclamo a través de otra entidad RECLAMO_X_OPERADOR que tiene como FK al reclamo asociado con el operador, y esta entidad cuenta con su propio identificador RECLAMO_x_USUARIO_NRO, de esta forma desnormalizamos el reclamo con el operador. Por otro lado, el reclamo tiene asociado (según el enunciado) el pedido al cual se le realizó, por este motivo también se agregó la clave del pedido y usuario a sus atributos.

Pedidos

Cada Pedido tiene los atributos que se pidieron en el TP, este cuenta con una PK llamada PEDIDO_NRO y con medio de pago, local y usuario como FK.

Los pedidos pueden tener uno o más productos en sí mismos y es por eso que utilizamos la tabla PRODUCTO_X_PEDIDO para referenciarlos con las FK de producto_local, pedido y local de manera única con una PK denominada PRODUCTO_X_PEDIDO_NRO.

Producto

Los productos se ven reflejados en la entidad PRODUCTO_LOCAL, el cual tiene como atributos un nombre, precio y descripción, así como una FK al Local al que dispone este producto. Los mismos cuentan con una PK denominada PRODUCTO_LOCAL_CODIGO

Local y Horario

Un local tiene su id que denominamos LOCAL_CODIGO y los atributos mencionados en el enunciado. Desnormalizamos el horario y creamos otra entidad con su propio PK y como atributo FK tiene al local asociado, junto con atributos como el día, apertura del local ese día y la hora de cierre del mismo.

Envío

Un pedido está relacionado a un envió donde cuenta con la FK de pedido y del repartidor asociado a este. Este cuenta con una PK para identificar el envío denominado ENVIO_NRO

Repartidor y tipo movilidad

Un repartido cuenta con su DNI como PK. Se normalizo la movilidad respecto del repartido creando una entidad TIPO_MOVILIDAD con su propio PK denominado TIPO_MOVILIDAD_COD y su movilidad como atributos.

Envío mensajería y paquete

Envío mensajería cuenta con todos los atributos mencionados en el enunciado, utilizando como FK a usuario, repartidor, medio de pago y paquete.

Paquete cuenta con su PK denominada PAQUETE_NRO y una FK para envío mensajería

Migración y Modelo de datos.

Creación de tablas

A partir de las correcciones del DER y revisación de la tabla maestra, decidimos rediseñar nuestro DER para luego facilitar la migración. Hacer coincidir los tipos de datos y nombres similares dentro de nuestras tablas fue algo casi necesario y excluyente para que podamos obtener una buena migración. El tipo de datos, era meramente necesario ya que si no la migración sería imposible. Los nombres, procuramos tenerlos lo más parecido posible para así no generar confusiones en las migraciones.

Vimos necesario que antes de crear cada tabla, constraint y procedures sea necesario eliminarlos o dropearlos de la base de datos para así poder probar varias veces y que se vuelvan a generar constantemente para no generar errores. Realizamos los drops en orden, ya que fue necesario primero eliminar los constraint que estaban relacionados con las tablas, luego las tablas creadas y por último los procedures.

Constraints

Decidimos escribir las **Constraints** aparte de las tablas, realizando alter tables, donde definíamos las **Foreign Keys** de cada tabla. Esto nos ayudó para poder generar una relación pertinente entre las tablas donde nos asegurábamos que el lugar donde íbamos a buscar la clave foránea era el adecuado.

Stored Procedures

Al momento de migrar los datos, decidimos utilizar **Stored Procedures** llamados "MIGRAR_NOMBRE_DE_TABLA". En ellos fuimos obteniendo los datos necesarios de gd_esquema.Maestra y pasarlos a la tabla indicada por el procedure en cuestión.

Fuimos limitando los posibles datos que podrían migrar en nuestras tablas realizando IS NOT NULL en los campos necesarios para no obtener campos nulos y así no generar inconsistencias en las tablas. A su vez, realizamos JOIN con otras tablas para poder obtener los datos adecuados.

Los JOIN nos obligaron a tener que realizar las ejecuciones de los procedures en orden, ya que para ciertas tablas, necesitábamos cargar otras tablas previamente.