Cristian Camilo Alvarado Vargas
SofkaU 2023

Almacén Don pepe (Ejercicio C)

En el contexto de la formación inicial de SofkaU se propone realiza el presente reto. A continuación se presenta el paso a paso de la solución obtenida:

● Indicar que ejercicio fue asignado

El ejercicio asignado se titula Tienda Virtual Don pepe (Ejercicio C)

● Realizar el modelo E-R

Para realizar el modelo entidad relación se realizó una abstracción del enunciado y se identificaron las siguientes entidades, atributos, comportamientos y requerimientos:

Requerimientos: Se requiere una base de datos para manejar el almacén de Don Pepe, en la que el cliente pueda realizar pedidos desde su casa.

Las entidades de la base de datos son:

Responsable

-Responsable: Esta entidad representa a un responsable en el almacén. Esta persona es la encargada de recibir la información de los productos solicitados por el cliente y empacarlos en una cesta. Luego de empacar los productos el responsable en el almacén entrega la cesta al domiciliario para posterior entrega. Los atributos asociados a la entidad responsables son: cedula_res, nombre_res y telefono_res.
Cardinalidad: Un responsable en almacén empaca muchos pedidos. Un pedido es empacado por solo un responsable.
Participación: El pedido puede existir sin el responsable y el responsable existe sin el pedido.

Domiciliario

-Domiciliario: Esta entidad representa un domiciliario que recibe una cesta de parte del responsable en el almacén con los productos solicitados y posteriormente la lleva a la casa del cliente. La entidad domiciliario tiene los siguientes atributos: cedula_dom, nombre_dom y telefono_dom.
Cardinalidad: Un domiciliario puede entregar muchos pedidos. Un pedido puede ser entregado por solo un domiciliario.
Un domiciliario puede usar muchas furgonetas. Una furgoneta puede ser usada solo por un domiciliario.
Un domiciliario puede trabajar en una sola zona, en una zona pueden trabajar muchos domiciliarios.
Participación: El domiciliario no puede existir sin la furgoneta y la furgoneta no puede existir sin el domiciliario.
El domiciliario no puede existir sin una zona. Una zona si puede existir sin un domiciliario.

Cristian Camilo Alvarado Vargas
SofkaU 2023

Cliente

-Cliente: Esta entidad representa al cliente que realiza un pedido, el cual tiene asociado una canasta con unos productos específicos. La entidad cliente tiene los siguientes atributos: ID_cli, cedula_cli, nombre_cli, telefono_cli, direccion_cli, email_cli, contraseña_cli y zona_cli.
Cardinalidad: Un cliente puede realizar muchos pedidos, un pedido puede ser realizado solo por un cliente.
En una zona pueden vivir muchos clientes, un cliente solo puede vivir en una zona.
Participación: Un pedido no existe sin un cliente. Un cliente existe sin un pedido.
Una zona existe sin un cliente. Un cliente existe sin una zona.

Pedido

-Pedido: Esta entidad representa un pedido realizado por un cliente. La entidad pedido tiene los siguientes atributos: codigo_pedido, fecha_pedido, total_importe, datos_pago (número de tarjeta y fecha de caducidad), estado.
Cardinalidad: Un pedido puede generar solo una factura y una factura puede ser generada por solo un pedido.
Un pedido solo puede tener una cesta asociada. Una cesta solo puede tener un pedido asociado.
Participación: Una factura no existe sin un pedido. Un pedido si existe sin una factura.
Un pedido puede existir sin una cesta. Una cesta no puede existir sin un pedido.

Cesta

-Cesta: Esta entidad representa la cesta de productos que solicita el cliente. En esta entidad se relacionan todos los productos que solicita un cliente en un pedido específico. La entidad cesta tiene los siguientes atributos: ID_cesta, ID_producto y cantidad.
Cardinalidad: Una cesta puede tener varios productos. Un producto puede ser tenido en muchas cestas.
Participación: Una cesta no existe sin un producto. Un producto si existe sin una cesta.

Producto

-Producto: Esta entidad representa los productos que se tienen en la tienda. La entidad producto tiene los siguientes atributos: ID_producto, nombre_prod, marca, origen, fotografia, unidades_disponibles, dimensiones (volumen y peso), valor_compra y valor_venta.
Cardinalidad: Un producto pertenece a una sola categoría, a una categoría pueden pertenecer muchos productos.
Un producto puede ser proveído por muchos proveedores, un proveedor puede proveer muchos productos.
Participación: Un producto no existe sin una categoría, una categoría no existe sin un producto.
Un proveedor existe sin un producto. Un producto no existe sin un proveedor.

Cristian Camilo Alvarado Vargas
SofkaU 2023

Furgoneta

-Furgoneta: Esta entidad representa una furgoneta en la cual el domiciliario entrega el pedido. La entidad furgoneta tiene los siguientes atributos: matricula_fur y estado_fur.

Zona

-Zona: Esta entidad representa una zona de la ciudad. La entidad zona tiene los siguientes atributos: ID_zona, codigo_postal y localidad_zona.
-Categoría: Esta entidad representa una categoría a la cual pertenece un producto. Pueden haber diferentes categorías como: carnes, lácteos, granos, cereales, verduras, frutas, tubérculos, entre otros. Por otra parte según la categoría se pueden identificar condiciones de almacenamiento que pueden ser frío, congelado o seco. La entidad categoría tiene los siguientes atributos: ID_categoria, nombre_cat, almacenamiento y observaciones.

Proveedor

-Proveedor: Esta entidad representa un proveedor que provee a la tienda con los productos. La entidad proveedor tiene los siguientes atributos: ID_proveedor, nombre_prov y telefono_prov.

Factura

-Factura: Esta entidad representa la factura que se genera cuando un pedido ha finalizado. La entidad factura tiene los siguientes atributos: ID_factura y total.

Categoría

-Categoría: Esta entidad representa una categoría a la que pertenece cada alimento. La tabla Categoria tiene los siguientes atributos: ID_categoria, nombre_cat, almacenamiento y observaciones. Pueden haber distintas categorías:

A. Carnes.
B. Granos.
C. Lácteos.
D. Verduras.
E. Frutas.
F. No perecederos.
G. Bebidas.

El atributo estado es un atributo derivado que se asignaría por la lógica de la aplicación. Puede tener los siguientes valores:

A. Creado por el cliente.
B. En verificación de existencias.
C. En verificación de responsable de almacen.
D. En verificación de domiciliario en zona de cliente.
E. Cancelado por existencias.
F. En espera a verificación de cliente aceptando nuevas condiciones.
G. Cancelado por no existencia de domiciliario en la zona.

H. Empacado por responsable.
I. Entregado por responsable a domiciliario.
J. En ruta.
K. Entregado a cliente.
L. Pagado por el cliente.
M. Finalizado.

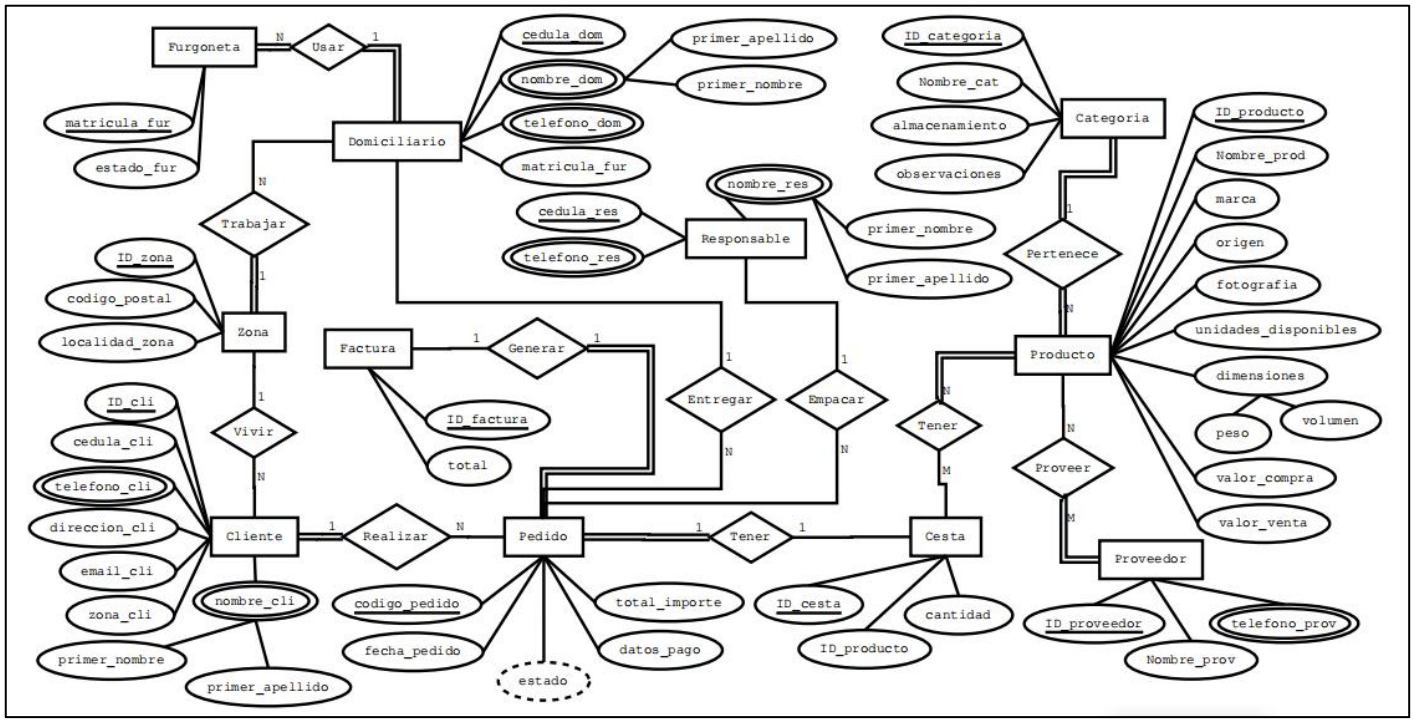
Otras consideraciones:

-Los domiciliarios son conocidos de Don Pepe ellos se encargan de llevar los domicilios a los clientes.

-Las furgonetas son propiedad de cada domiciliario, cada domiciliario tiene el deber de reportarle a Don Pepe cualquier novedad con el estado de su furgoneta.

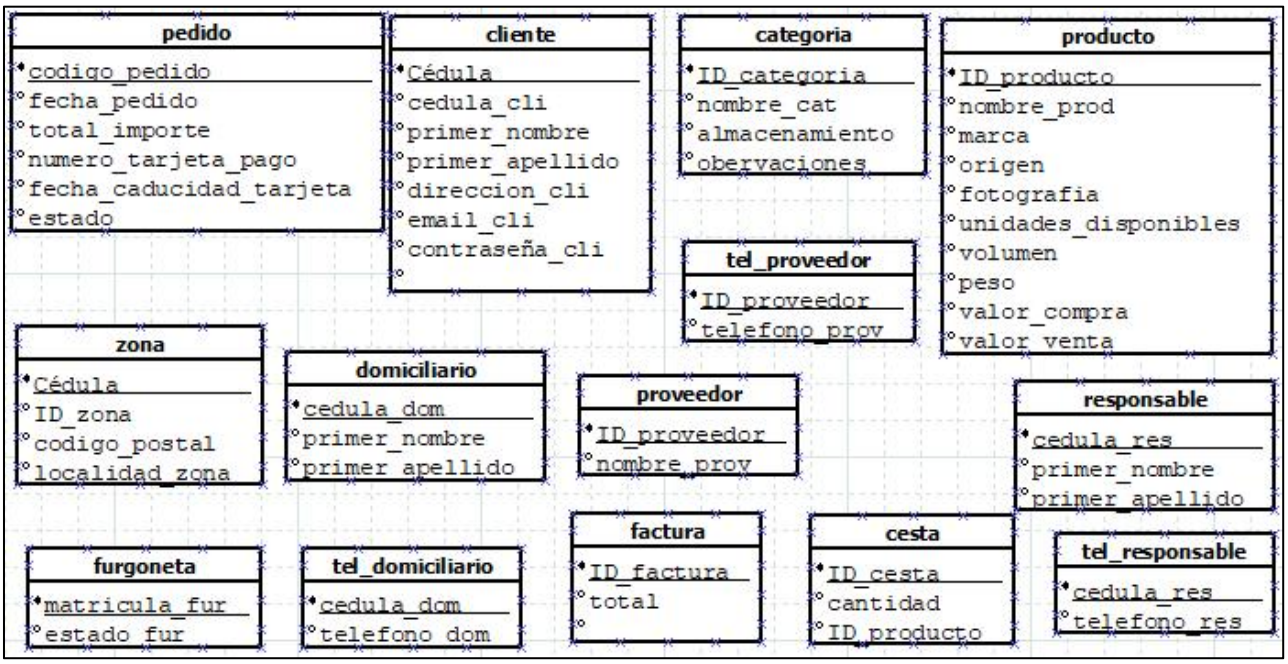
-Un domiciliario no puede realizar domicilios si no tiene furgoneta.

En función de lo anterior se realizó el siguiente diagrama E/R:



● Realizar el modelo relacional

En primer lugar se crean las tablas de cada entidad:



Una vez se tienen las tablas se identifican las relaciones según la cardinalidad. Para realizar el modelo relacional se tuvo en cuenta lo siguiente:

-Los atributos asociados a la tabla Responsable son: cedula_res, primer_nombre y primer_apellido. cedula_res es clave primaria.

-Los atributos asociados a la tabla Tel_responsable son: cedula_res y telefono_res.

-La tabla domiciliario tiene los siguientes atributos: cedula_dom, primer_nombre, primer_apellido y ID_zona. cedula_dom es clave primaria, ID_zona es una llave foranea asociada a la tabla Zona.

-La tabla Telefono_dom tiene los siguientes atributos: cedula_dom y telefono_dom.

-La tabla furgoneta tiene los siguientes atributos: matricula_fur, estado_fur y cedula_dom. matricula_fur es llave primaria y cedula_dom es llave foranea relacionada con la tabla Domiciliario.

-La tabla Cliente tiene los siguientes atributos: ID_cli, cedula_cli, primer_nombre, primer_apellido, direccion_cli, email_cli, contraseña_cli y ID_zona. ID_cli es una llave primaria, ID_zona es llave foranea asociada a la tabla Zona.

-La tabla Cliente_Pedido tiene los siguientes atributos: ID_cli_codigo_pedido, ID_cli y codigo_pedido. ID_cli_codigo_pedido es llave primaria, ID_cli es llave foranea

relacionada con la tabla Cliente, codigo_pedido es llave foranea relacionada con la tabla Pedido.

-La tabla Zona tiene los siguientes atributos: ID_zona, codigo_postal y localidad_zona.

-La tabla Pedido tiene los siguientes atributos: codigo_pedido, fecha_pedido, total_importe, número_tarjeta_pago, fecha_caducidad_tarjeta, estado, cedula_res, cedula_dom, ID_cesta. El atributo codigo_pedido es llave primaria, el atributo cedula_dom es llave foranea relacionada con la tabla Domiciliario, el atributo cedula_res es llave foranea realacionada con la tabla Responsable, el atributo ID_cesta es llave foranea relacionada con la tabla Cesta.

-La tabla Cesta tiene los siguientes atributos: ID_cesta, ID_producto, cantidad, codigo_pedido. El atributo ID_cesta es llave primaria, el atributo ID_producto es llave foranea relacionada con la tabla Producto, el atributo codigo_pedido es llave foranea a la tabla Pedido.

-La tabla Producto tiene los siguientes atributos: ID_producto, nombre_prod, marca, origen, fotografia, unidades_disponibles, volumen, peso, valor_compra, valor_venta, ID_categoria. ID_producto es la llave primaria, ID_categoria es llave foranea asociada a la tabla Categoria.

-La tabla Producto_Proveedor tiene los siguientes atributos: ID_producto_ID_proveedor, ID_producto, ID_proveedor. ID_producto_ID_proveedor es clave primaria, ID_producto es llave foranea relacionada con la tabla Producto, ID_proveedor es llave foranea relacionada con la tabla Proveedor.

-La tabla Categoria tiene los siguientes atributos: ID_categoria, nombre_cat, almacenamiento y observaciones.

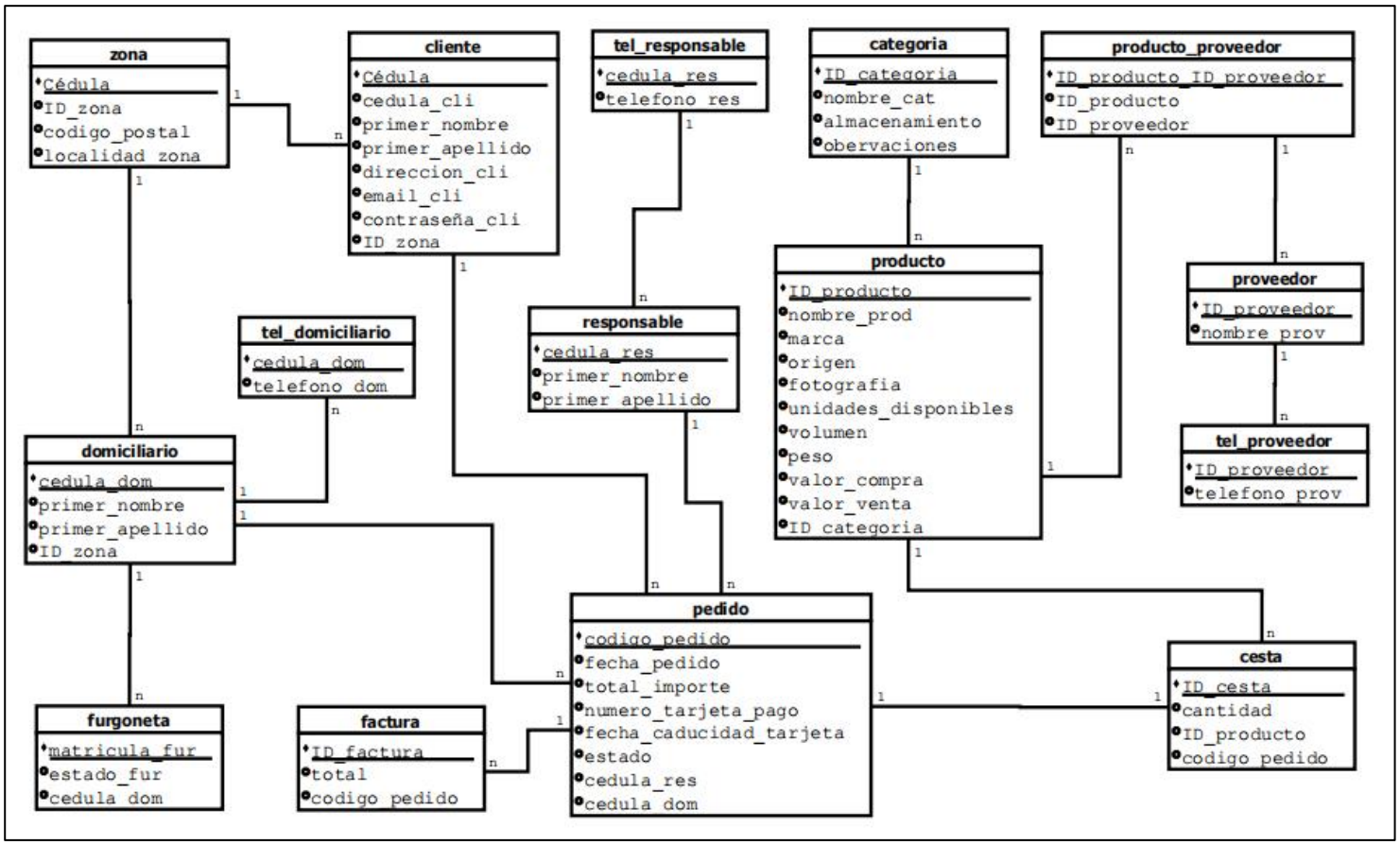
-La tabla Proveedor tiene los siguientes atributos: ID_proveedor y nombre_prov.

-La tabla Telefono_prov tiene los siguientes atributos: ID_proveedor y telefono_prov. ID_proveedor es llave primaria.

-La tabla factura tiene los siguientes atributos: ID_factura, total y codigo_pedido. ID_factura es la llave primaria, codigo_pedido es llave foranea relacionada con la tabla Pedido.

Para las relaciones N:M se crean las nuevas tablas respectivas, se tienen en cuenta las relaciones según las llaves foraneas, en consecuencia se obtiene el siguiente modelo relacional:

Cristian Camilo Alvarado Vargas
SofkaU 2023



- **Normalizar correctamente**

Se identifica que el modelo relacional no tiene atributos repetidos, cada tabla tiene una llave primaria, no hay atributos multivaluados, todos los atributos tienen valores atómicos. En consecuencia se encuentra en 1ra forma normal.

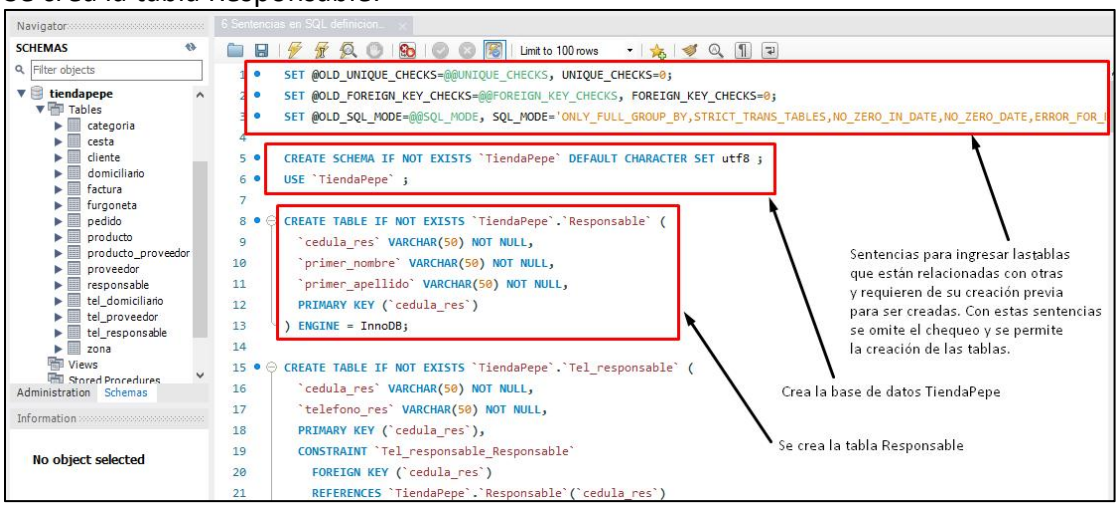
Por otra parte todos los valores de las columnas dependen únicamente de la llave primaria de cada tabla y las tablas tienen una única llave primaria que identifica a la tabla y sus atributos dependen de ella. En consecuencia se encuentra en 2da forma normal.

El modelo relacional esta en 2da forma normal, en 1ra forma normal y cada atributo que no está incluido en la clave primaria no depende transitivamente de la clave primaria. En consecuencia esta en 3ra forma normal. Por lo anterior se ha normalizado el modelo.

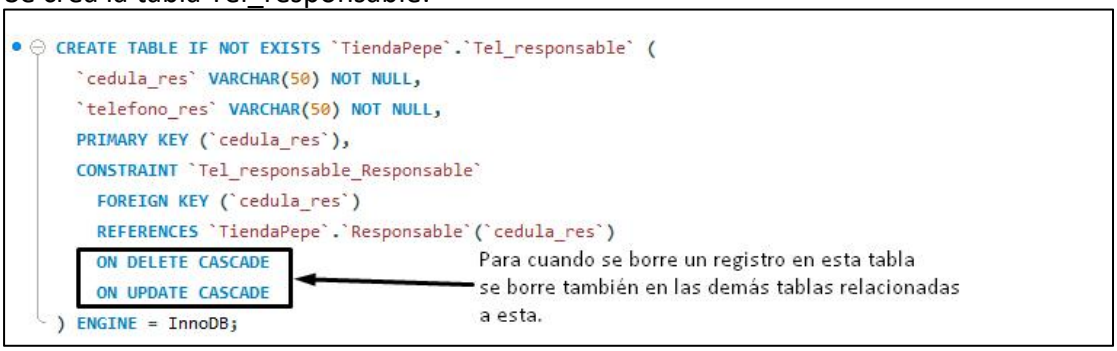
Cristian Camilo Alvarado Vargas
SofkaU 2023

- **Escribir con sentencias SQL toda la definición de la base de datos.**

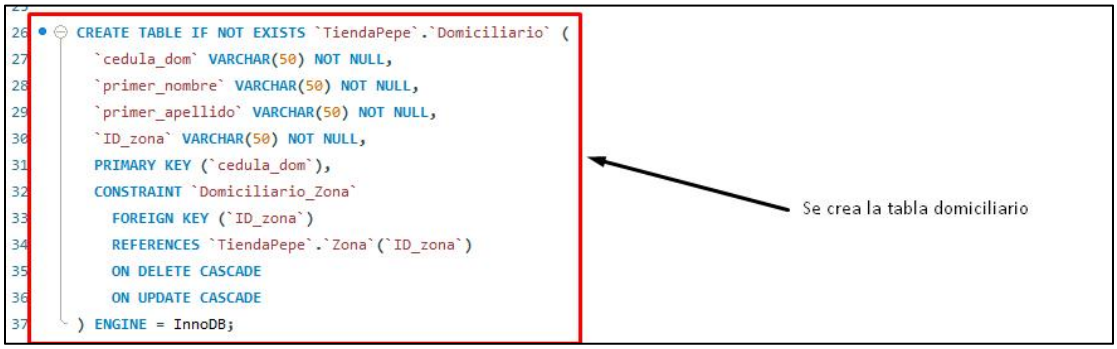
Se crea la tabla Responsable:



Se crea la tabla Tel_responsable:



Se crea la tabla domiciliario:



Se crea la tabla Tel_domiciliario:

```
38
39 • CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Tel_domiciliario` (
40     `cedula_dom` VARCHAR(50) NOT NULL,
41     `telefono_dom` VARCHAR(50) NOT NULL,
42     PRIMARY KEY (`cedula_dom`),
43     CONSTRAINT `Telefono_dom_Domiciliario`
44       FOREIGN KEY (`cedula_dom`)
45       REFERENCES `TiendaPepe`.`Domiciliario`(`cedula_dom`)
46       ON DELETE CASCADE
47       ON UPDATE CASCADE
48   ) ENGINE = InnoDB;
```

Se crea la tabla Furgoneta:

```
50 • CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Furgoneta` (
51     `matricula_fur` VARCHAR(50) NOT NULL,
52     `estado_fur` VARCHAR(50) NOT NULL,
53     `cedula_dom` VARCHAR(50) NOT NULL,
54     PRIMARY KEY (`matricula_fur`),
55     CONSTRAINT `Furgoneta_Domiciliario`
56       FOREIGN KEY (`cedula_dom`)
57       REFERENCES `TiendaPepe`.`Domiciliario`(`cedula_dom`)
58       ON DELETE CASCADE
59       ON UPDATE CASCADE
60   ) ENGINE = InnoDB;
61
```

Se crea la tabla Cliente:

```
62 • CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Cliente` (
63     `ID_cli` VARCHAR(50) NOT NULL,
64     `cedula_cli` VARCHAR(50) NOT NULL,
65     `primer_nombre` VARCHAR(50) NOT NULL,
66     `primer_apellido` VARCHAR(50) NOT NULL,
67     `direccion_cli` VARCHAR(50) NOT NULL,
68     `email_cli` VARCHAR(50) NOT NULL,
69     `contraseña_cli` VARCHAR(50) NOT NULL,
70     `ID_zona` VARCHAR(50) NOT NULL,
71     PRIMARY KEY (`ID_cli`),
72     CONSTRAINT `Cliente_Zona`
73       FOREIGN KEY (`ID_zona`)
74       REFERENCES `TiendaPepe`.`Zona`(`ID_zona`)
75       ON DELETE CASCADE
76       ON UPDATE CASCADE
77   ) ENGINE = InnoDB;
```

Se crea la tabla Zona:

```
79 • CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Zona` (
80     `ID_zona` VARCHAR(50) NOT NULL,
81     `codigo_postal` VARCHAR(50) NOT NULL,
82     `localidad_zona` VARCHAR(50) NOT NULL,
83     PRIMARY KEY (`ID_zona`)
84   ) ENGINE = InnoDB;
```

Se crea la tabla Factura:

```
CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Factura` (
  `ID_factura` VARCHAR(45) NOT NULL,
  `total` VARCHAR(45) NOT NULL,
  `codigo_pedido` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`ID_factura`),
  FOREIGN KEY (`codigo_pedido`)
  REFERENCES `TiendaPepe`.`Pedido`(`codigo_pedido`)
) ENGINE = InnoDB;
```

Se crea la tabla Pedido:

```
CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Pedido` (
  `codigo_pedido` VARCHAR(45) NOT NULL,
  `fecha_pedido` VARCHAR(45) NOT NULL,
  `total_importe` VARCHAR(45) NOT NULL,
  `numero_tarjeta_pago` VARCHAR(16) NOT NULL,
  `fecha_caducidad_tarjeta` VARCHAR(16) NOT NULL,
  `estado` VARCHAR(45) NOT NULL,
  `cedula_res` VARCHAR(45) NOT NULL,
  `cedula_dom` VARCHAR(45) NOT NULL,
  `ID_cli` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`codigo_pedido`),
  CONSTRAINT `Pedido_Domiciliario1`
    FOREIGN KEY (`cedula_dom`)
    REFERENCES `TiendaPepe`.`Domiciliario` (`cedula_dom`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `Pedido_Responsable1`
    FOREIGN KEY (`cedula_res`)
    REFERENCES `TiendaPepe`.`Responsable` (`cedula_res`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
```

Se crea la tabla Cesta:

```
CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Cesta` (
  `ID_cesta` VARCHAR(45) NOT NULL,
  `ID_producto` VARCHAR(45) NOT NULL,
  `cantidad` VARCHAR(45) NOT NULL,
  `codigo_pedido` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`ID_cesta`),
  CONSTRAINT `Cesta_Producto1`
    FOREIGN KEY (`ID_producto`)
    REFERENCES `TiendaPepe`.`Producto` (`ID_producto`),
  CONSTRAINT `Cesta_Pedido1`
    FOREIGN KEY (`codigo_pedido`)
    REFERENCES `TiendaPepe`.`Pedido` (`codigo_pedido`)
) ENGINE = InnoDB;
```

Se crea la tabla Categoria:

```
CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Categoria` (
  `ID_categoria` VARCHAR(45) NOT NULL,
  `nombre_cat` VARCHAR(45) NOT NULL,
  `almacenamiento` VARCHAR(45),
  `observaciones` VARCHAR(50),
  PRIMARY KEY (`ID_categoria`)
) ENGINE = InnoDB;
```

Se crea la tabla Producto:

```
144
145 • CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Producto` (
146     `ID_producto` VARCHAR(45) NOT NULL,
147     `nombre_prod` VARCHAR(45) NOT NULL,
148     `marca` VARCHAR(45) NOT NULL,
149     `origen` VARCHAR(45) NOT NULL,
150     `fotografia` VARCHAR(45),
151     `unidades_disponibles` VARCHAR(45) NOT NULL,
152     `volumen` VARCHAR(45),
153     `peso` VARCHAR(45),
154     `valor_compra` VARCHAR(45) NOT NULL,
155     `valor_venta` VARCHAR(45) NOT NULL,
156     `ID_categoria` VARCHAR(45) NOT NULL,
157     PRIMARY KEY (`ID_producto`),
158     FOREIGN KEY (`ID_categoria`)
159     REFERENCES `TiendaPepe`.`Categoria`(`ID_categoria`)
160   ) ENGINE = InnoDB;
```

Se crea la tabla Producto_Proveedor:

```
162 • CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Producto_Proveedor` (
163     `ID_producto_ID_proveedor` VARCHAR(45) NOT NULL,
164     `ID_producto` VARCHAR(45) NOT NULL,
165     `ID_proveedor` VARCHAR(45) NOT NULL,
166     PRIMARY KEY (`ID_producto_ID_proveedor`,`ID_producto`,`ID_proveedor`),
167     FOREIGN KEY (`ID_producto`)
168     REFERENCES `TiendaPepe`.`Producto`(`ID_producto`),
169     FOREIGN KEY (`ID_proveedor`)
170     REFERENCES `TiendaPepe`.`Proveedor`(`ID_proveedor`)
171   ) ENGINE = InnoDB;
172
```

Se crea la tabla Proveedor:

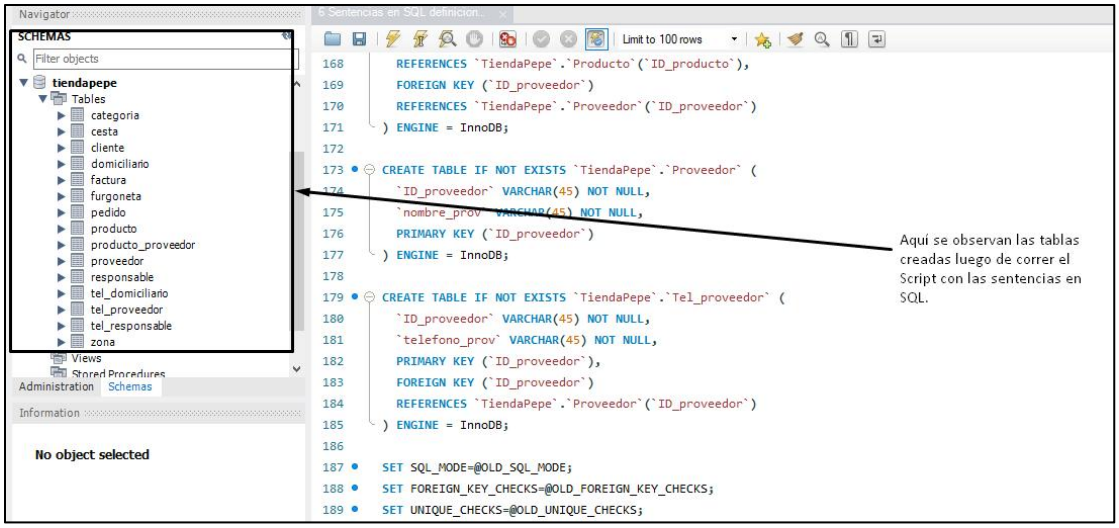
```
172
173 • CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Proveedor` (
174     `ID_proveedor` VARCHAR(45) NOT NULL,
175     `nombre_prov` VARCHAR(45) NOT NULL,
176     PRIMARY KEY (`ID_proveedor`)
177   ) ENGINE = InnoDB;
178
```

Se crea la tabla Tel_proveedor:

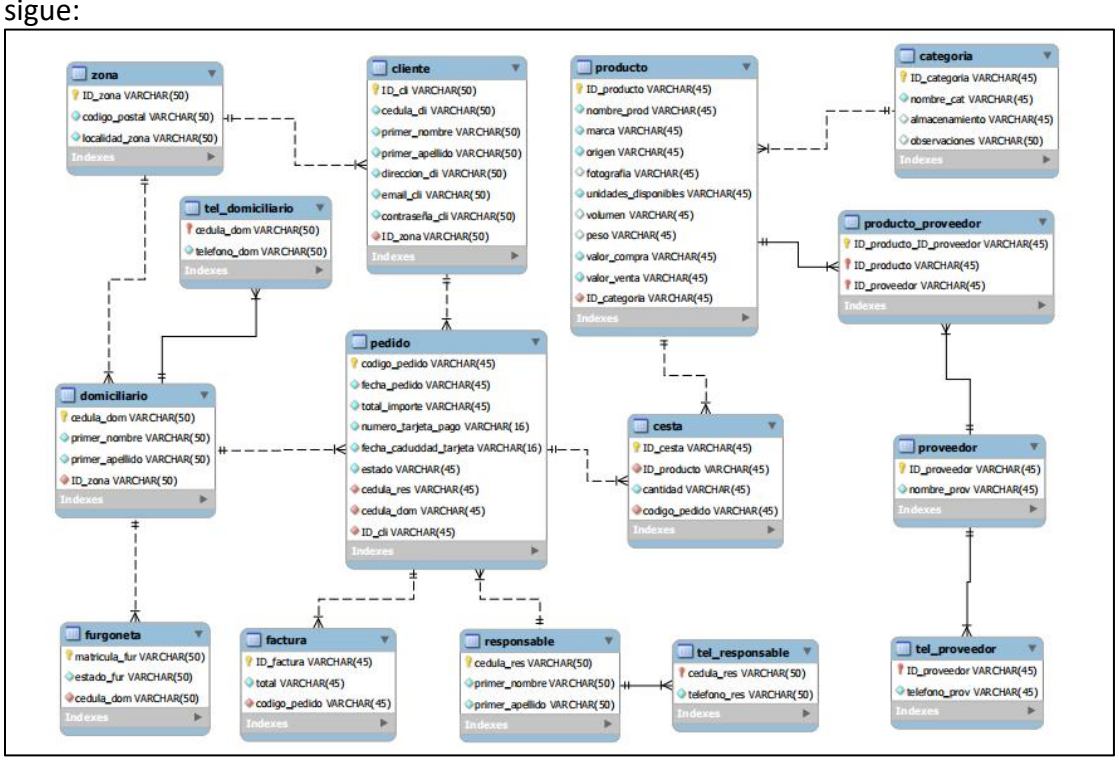
```
179 • CREATE TABLE IF NOT EXISTS `TiendaPepe`.`Tel_proveedor` (
180     `ID_proveedor` VARCHAR(45) NOT NULL,
181     `telefono_prov` VARCHAR(45) NOT NULL,
182     PRIMARY KEY (`ID_proveedor`),
183     FOREIGN KEY (`ID_proveedor`)
184     REFERENCES `TiendaPepe`.`Proveedor`(`ID_proveedor`)
185   ) ENGINE = InnoDB;
```

A continuación se observa la creación de las tablas en MySQL:

Cristian Camilo Alvarado Vargas
SofkaU 2023



Realizando ingeniería inversa se obtiene el modelo relacional en Workbench como sigue:



- **Escribir consultas que me permitan ver la información de cada tabla o de varias tablas (10).**

Para mostrar ejemplos las consultas previamente se insertan algunos registros en cada una de las tablas como sigue:

Ingreso de registros en la tabla proveedor:

```
SELECT * FROM tiendapepe.proveedor;
INSERT INTO `TiendaPepe`.`Proveedor`(`ID_proveedor`,`nombre_prov`)
VALUES
('PROV01','Soluciones logisticas de transporte'),
('PROV02','Cargolargo'),
('PROV03','Transportes la garza');
```

Ingreso de registros en la tabla tel_proveedor:

```
SELECT * FROM tiendapepe.tel_proveedor;
INSERT INTO `TiendaPepe`.`Tel_proveedor`(`ID_proveedor`,`telefono_prov`)
VALUES
('PROV01','3193202102'),
('PROV02','3196966956'),
('PROV03','3195858499');

```

Ingreso de registros en la tabla categoria:

```
SELECT * FROM tiendapepe.categoria;
INSERT INTO `TiendaPepe`.`Categoria`(`ID_categoria`,`nombre_cat`,`almacenamiento`,`observaciones`)
VALUES
('CA01', 'carnes','Almacenamiento en congelador','Llevar tiempos de viaje y almacenamiento'),
('CA02', 'granos','Almacenaiento en lugar fresco','Acomodar periodicamente'),
('CA03', 'vegetales','Almacenaiento en lugar fresco','Revisar el estado y hacer cambio'),
('CA04', 'frutas','Almacenaiento en lugar fresco','Revisar el estado y hacer cambio'),
('CA05', 'lacteos','Almacenamiento en congelador','Llevar tiempos de viaje y almacenamiento'),
('CA06', 'Aseo y hogar','Almacenamiento al aire libre','Limpiar con trapo húmedo cada 3 días');

```

Ingreso de registros en la tabla producto:

```
SELECT * FROM tiendapepe.producto;
INSERT INTO `TiendaPepe`.`producto` (`ID_producto`,`nombre_prod`,`marca`,`origen`,`fotografia`,`unidades_
VALUES
('PRO01','Jabón líquido','Axion','Colombiano','fotografía1','59','42','50','13000','14500','CA06');
INSERT INTO `TiendaPepe`.`producto` (`ID_producto`,`nombre_prod`,`marca`,`origen`,`fotografia`,`unidades_
VALUES
('PRO02','Carne de res','Frigor','Argentina','fotografía2','23','10','500','12000','15000','CA01');
INSERT INTO `TiendaPepe`.`producto` (`ID_producto`,`nombre_prod`,`marca`,`origen`,`fotografia`,`unidades_
VALUES
('PRO03','Frijol rojo','Luker','Colombia','fotografía3','87','5','100','4000','6000','CA02');
INSERT INTO `TiendaPepe`.`producto` (`ID_producto`,`nombre_prod`,`marca`,`origen`,`fotografia`,`unidades_
VALUES
('PRO04','Leche deslactosada','Alpina','Colombia','fotografía4','48','5','1000','4000','6000','CA05');
INSERT INTO `TiendaPepe`.`producto` (`ID_producto`,`nombre_prod`,`marca`,`origen`,`fotografia`,`unidades_
VALUES
('PRO05','Tomate','Del huerto','Colombia','fotografía5','72','2','50','1500','2000','CA03');
```

Ingreso de registros en la tabla cesta:

```sql
SELECT * FROM tiendapepe.cesta;
INSERT INTO `TiendaPepe`.`Cesta`(`ID_cesta`,`ID_producto`,`cantidad`,`codigo_pedido`)
VALUES
('C01', 'PRO01','12','P01'),
('C02', 'PRO04','5','P01'),
('C03', 'PRO09','16','P01'),
('C04', 'PRO03','12','P01'),
('C05', 'PRO08','4','P02'),
('C06', 'PRO05','11','P03'),
('C07', 'PRO01','12','P03'),
('C08', 'PRO04','9','P03'),
('C09', 'PRO06','1','P03');
```

Ingreso de registros en la tabla domiciliario:

```sql
SELECT * FROM tiendapepe.domiciliario;
INSERT INTO `TiendaPepe`.`Domiciliario`(`cedula_dom`,`primer_nombre`,`primer_apellido`,`ID_zona`)
VALUES
('896698', 'Duran','Deles','Zona1'),
('852258', 'Dario','Durango','Zona2'),
('874478', 'Demian','Dolores','Zona3');
```

Ingreso de registros en la tabla tel_domiciliario:

```sql
SELECT * FROM tiendapepe.tel_domiciliario;
INSERT INTO `TiendaPepe`.`Tel_domiciliario`(`cedula_dom`,`telefono_dom`)
VALUES
('896698', '320896698'),
('852258', '320852258'),
('874478', '320874478');
```

Ingreso de registros en la tabla cliente:

```sql
SELECT * FROM tiendapepe.cliente;
INSERT INTO `TiendaPepe`.`Cliente`(`ID_cli`,`cedula_cli`,`primer_nombre`,`primer_apellido`,`direccion_cli`
VALUES
('Cli1','456654','Carlos','Cubides','Dirección1','carlos@gmail.com','carlos1234','Zona1'),
('Cli2','478874','Carla','Claron','Dirección2','carla@gmail.com','carla1234','Zona2'),
('Cli3','423324','Cintia','Camero','Dirección3','cintia@gmail.com','cintia1234','Zona3');
```

Ingreso de registros en la tabla furgoneta:

```sql
SELECT * FROM tiendapepe.furgoneta;
INSERT INTO `TiendaPepe`.`Furgoneta`(`matricula_fur`,`estado_fur`,`cedula_dom`)
VALUES
('MCK-896','operativa','896698'),
('BAS-852','en mantenimiento','852258'),
('LOD-874','operativa','874478');
```

Ingreso de registros en la tabla pedido:

```
SELECT * FROM tiendapepe.pedido;
INSERT INTO `TiendaPepe`.`Pedido`(`codigo_pedido`,`fecha_pedido`,`total_importe`,`numero_tarjeta_pago`,`f
VALUES
('P01','17/02/2023','658000','123321','17/05/2024','En ruta','987789','896698','Cli1'),
('P02','17/02/2023','120000','145541','17/06/2025','En ruta','963369','852258','Cli2'),
('P03','17/02/2023','340000','159951','28/04/2025','Entregado a cliente','951159','874478','Cli3');
```

Ingreso de registros en la tabla factura:

```
SELECT * FROM tiendapepe.factura;
INSERT INTO `TiendaPepe`.`Factura`(`ID_factura`,`total`,`codigo_pedido`)
VALUES
('F01','658000','P01'),
('F02','120000','P02'),
('F03','340000','P03');

```

Ingreso de registros en la tabla responsable:

```
SELECT * FROM tiendapepe.responsable;
INSERT INTO `TiendaPepe`.`Responsable`(`cedula_res`, `primer_nombre`,`primer_apellido` )
VALUES
('987789', 'Rita','Ruales'),
('963369', 'Román','Rincón'),
('951159', 'Rina','Romelo');

```

Ingreso de registros en la tabla tel_responsable:

```
SELECT * FROM tiendapepe.tel_responsable;
INSERT INTO `TiendaPepe`.`Tel_responsable`(`cedula_res`, `telefono_res`)
VALUES
('987789', '311987789'),
('963369', '311963369'),
('951159', '311951159');
```

Ingreso de registros en la tabla producto_proveedores:

```
SELECT * FROM tiendapepe.producto_proveedor;
INSERT INTO `TiendaPepe`.`Producto_Proveedor`(`ID_producto_ID_proveedor`,`ID_producto`,`ID_proveedor`)
VALUES
('PP01','PRO01','PROV02'),
('PP02','PRO02','PROV02'),
('PP03','PRO03','PROV02');
```

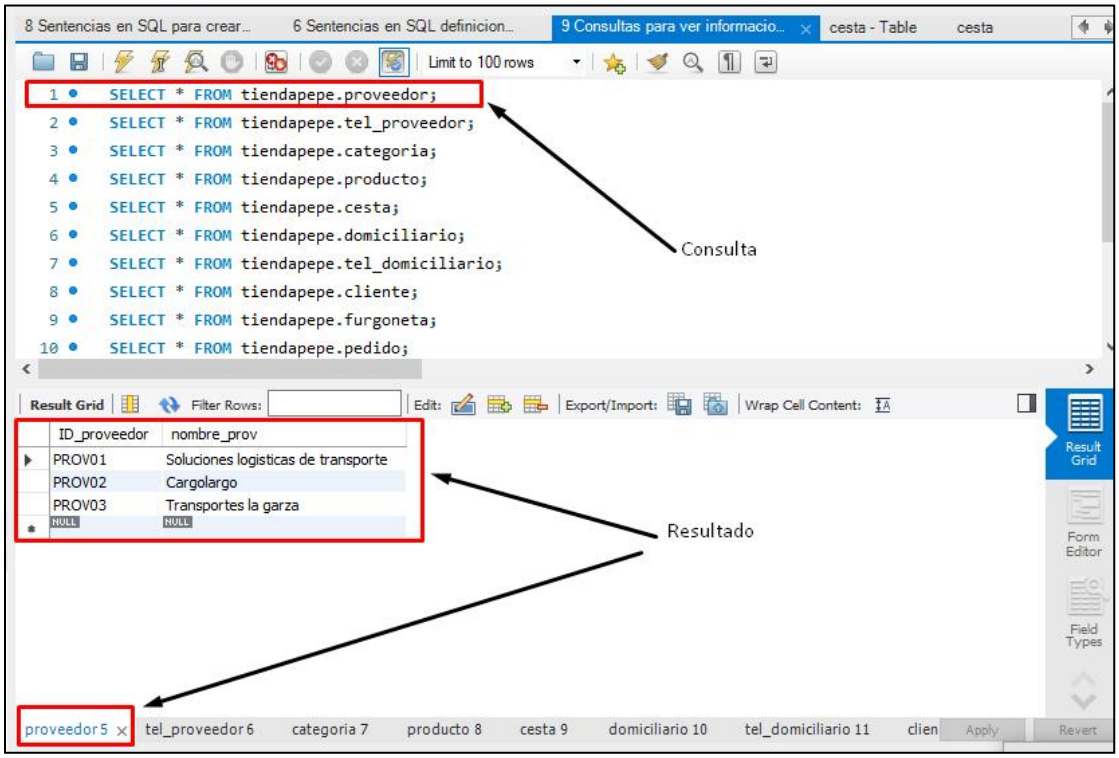
Ingreso de registros en la tabla zona:

```
SELECT * FROM tiendapepe.zona;
INSERT INTO `TiendaPepe`.`Zona`(`ID_zona`,`codigo_postal`,`localidad_zona`)
VALUES
('Zona1','111111','Localidad1'),
('Zona2','222222','Localidad2'),
('Zona3','333333','Localidad3');
```

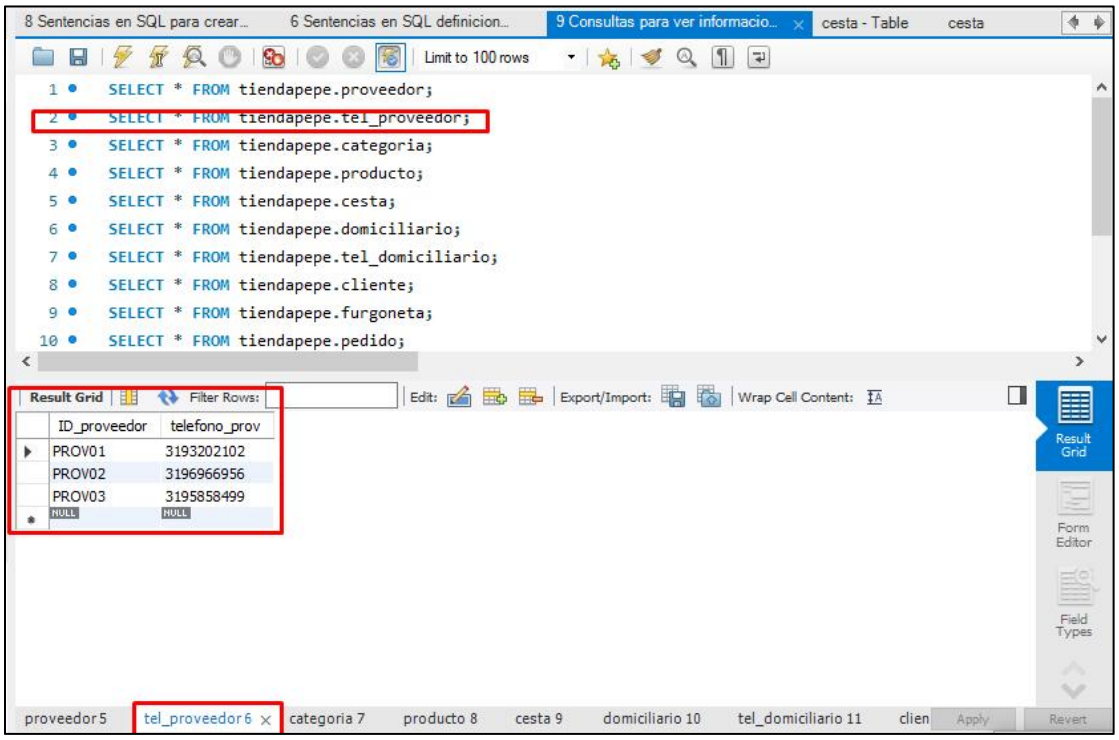
Para ver la información de cada tabla se escriben las siguientes consultas:

Cristian Camilo Alvarado Vargas
SofkaU 2023

Consulta tabla proveedor:

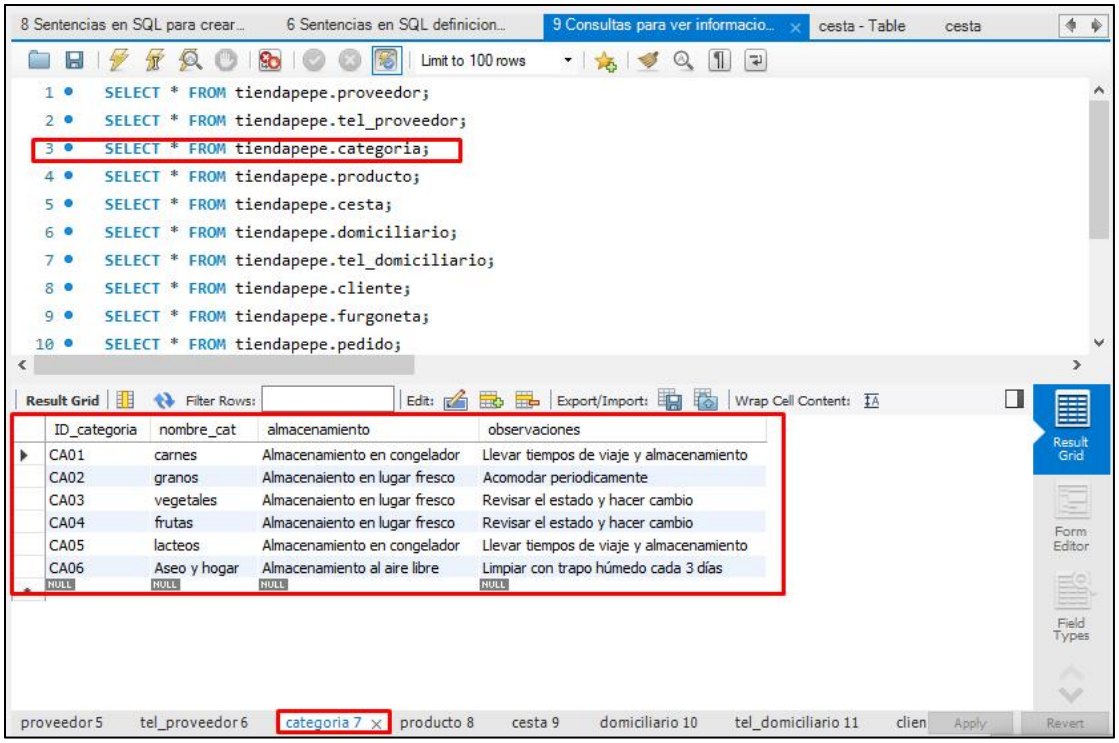


Consulta tabla tel_proveedor:

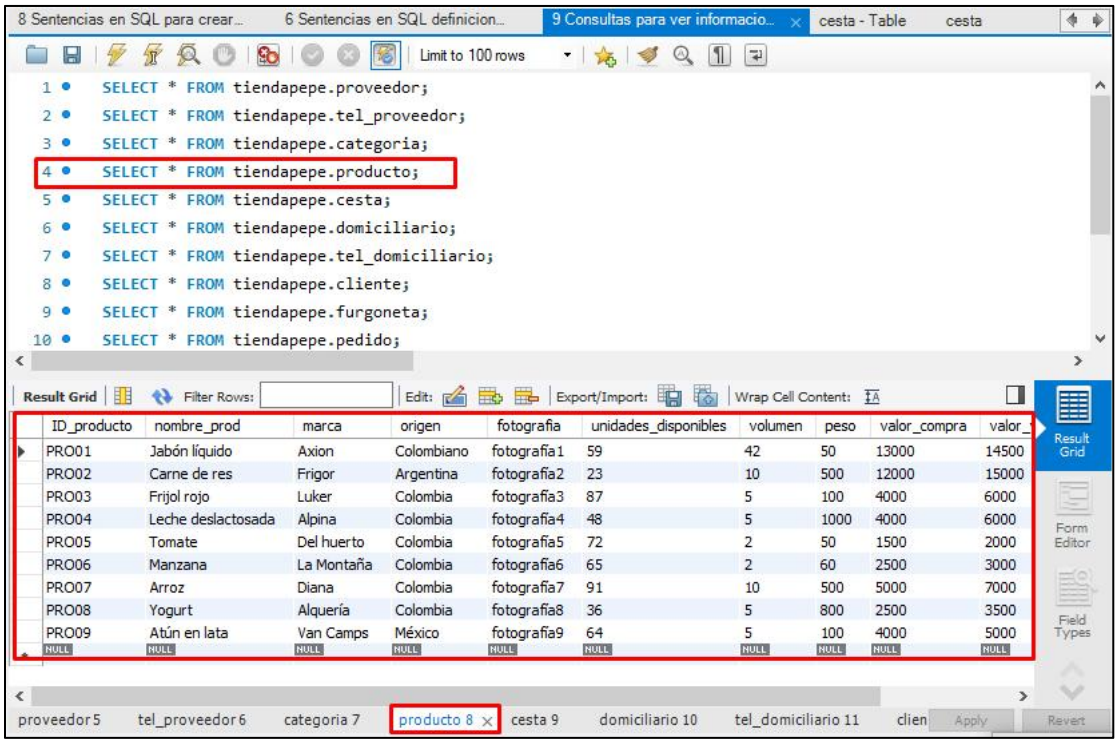


Consulta tabla categoria:

Cristian Camilo Alvarado Vargas
SofkaU 2023

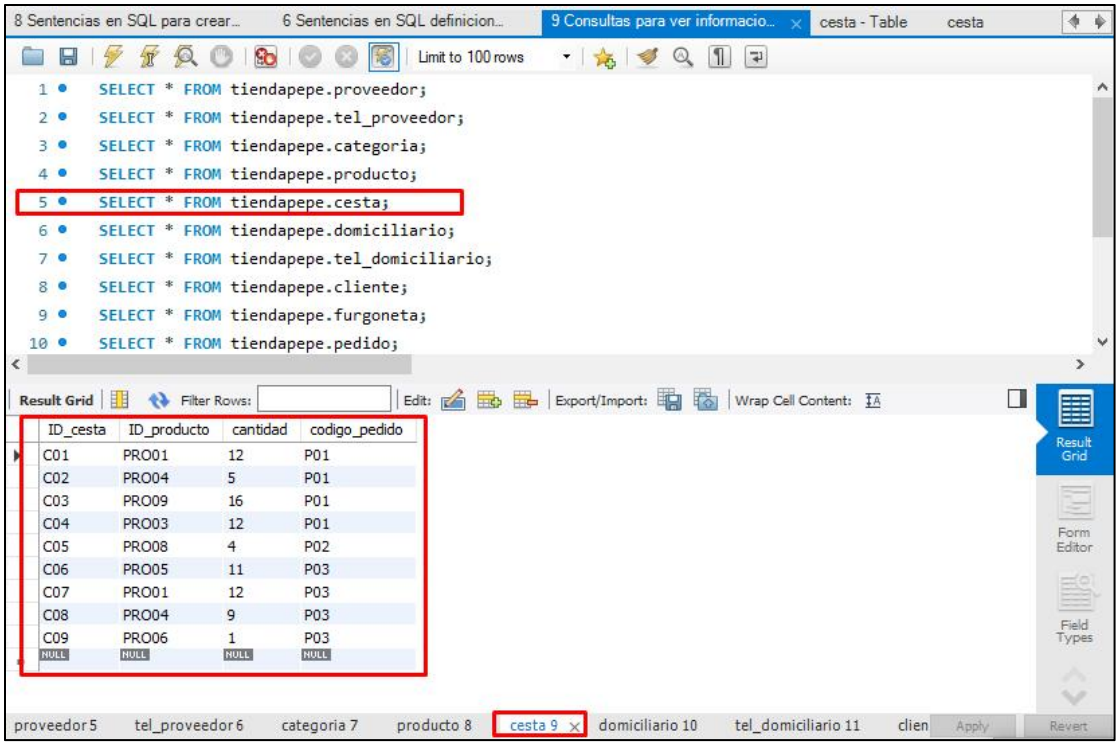


Consulta tabla producto:

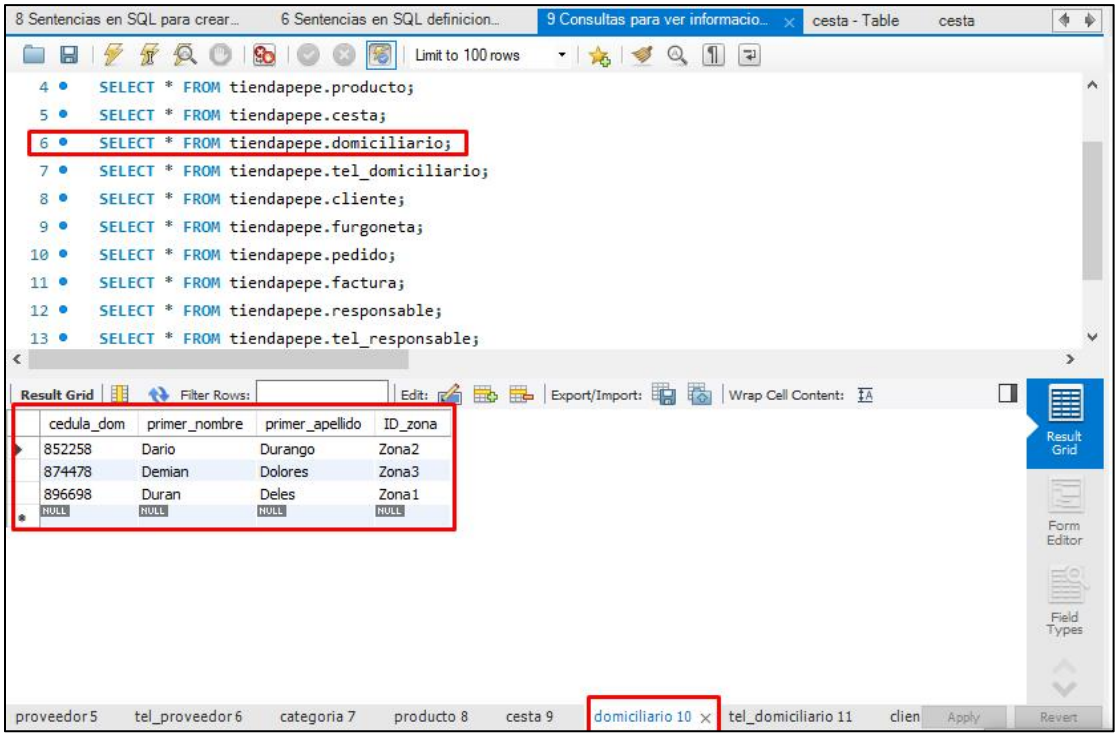


Consulta tabla cesta:

Cristian Camilo Alvarado Vargas
SofkaU 2023

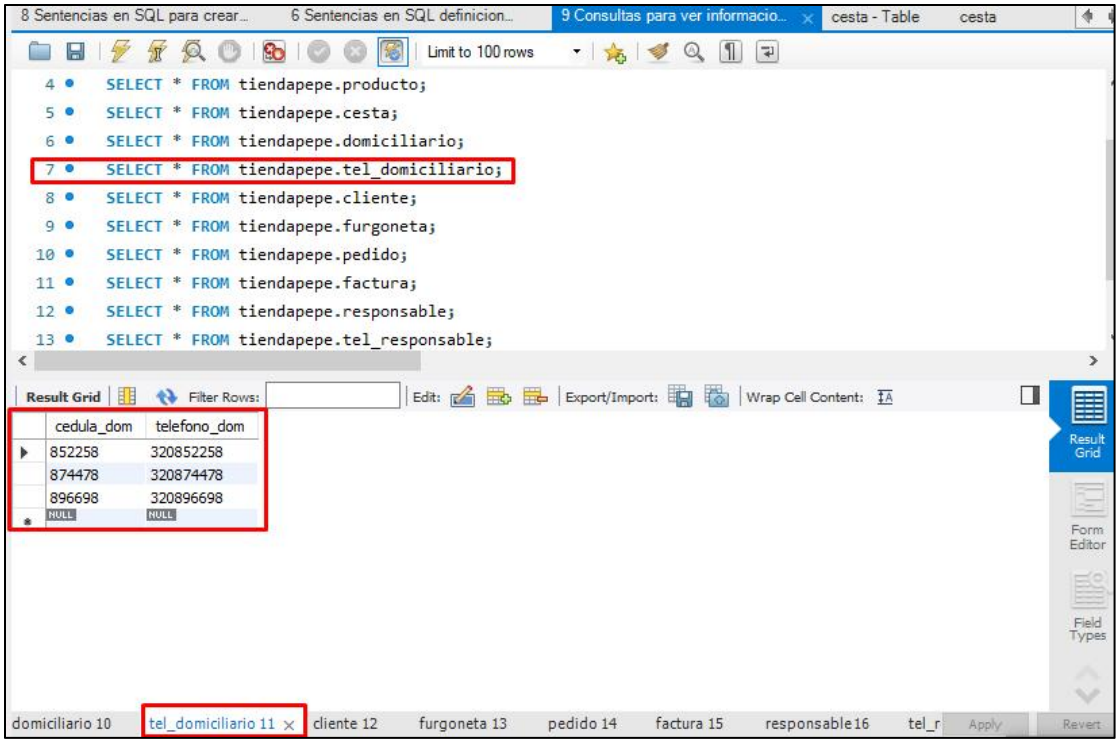


Consulta tabla domiciliario:

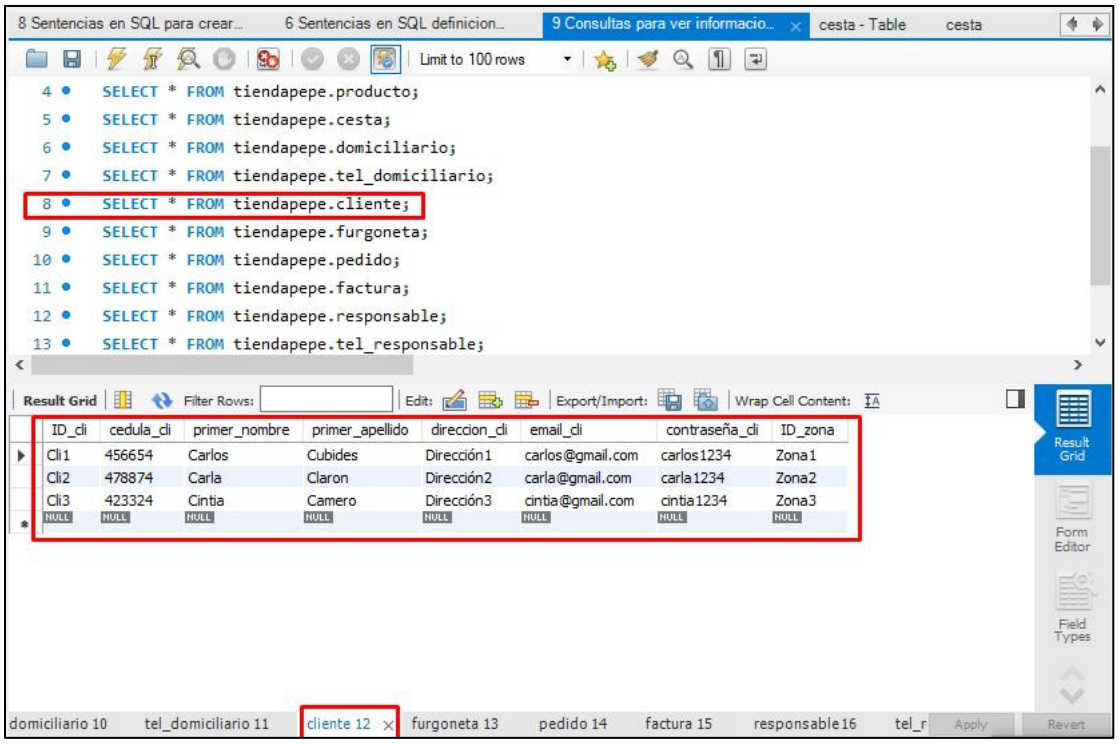


Consulta tabla tel_domiciliario:

Cristian Camilo Alvarado Vargas
SofkaU 2023

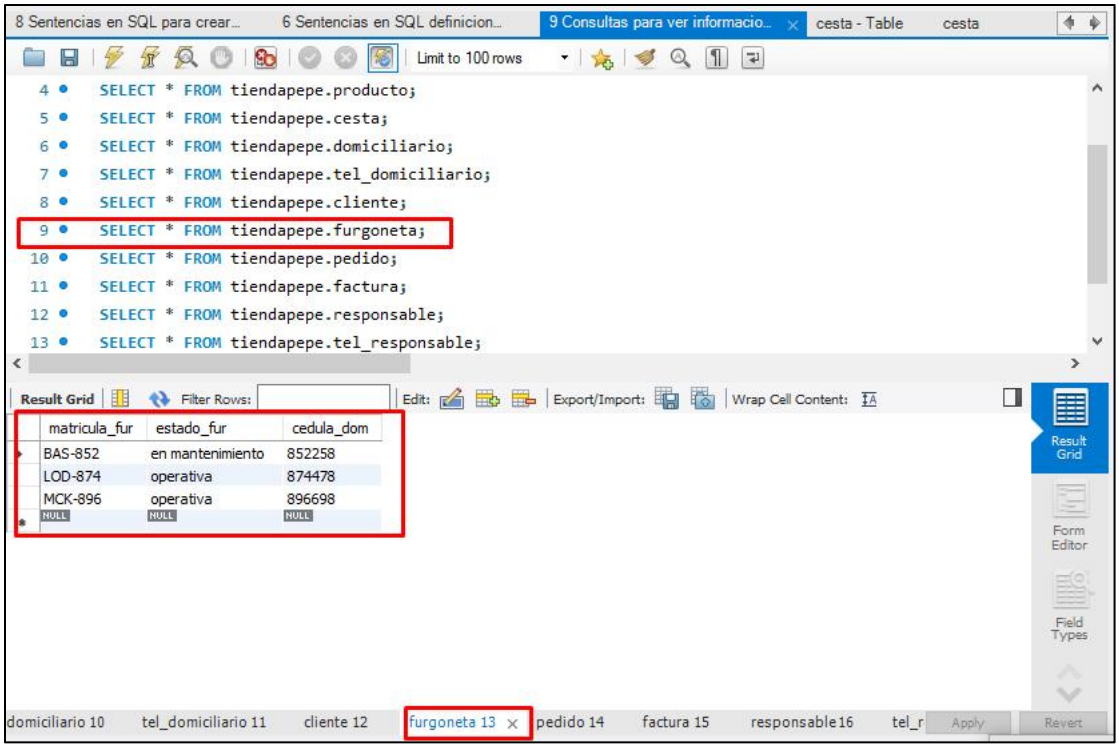


Consulta tabla cliente:

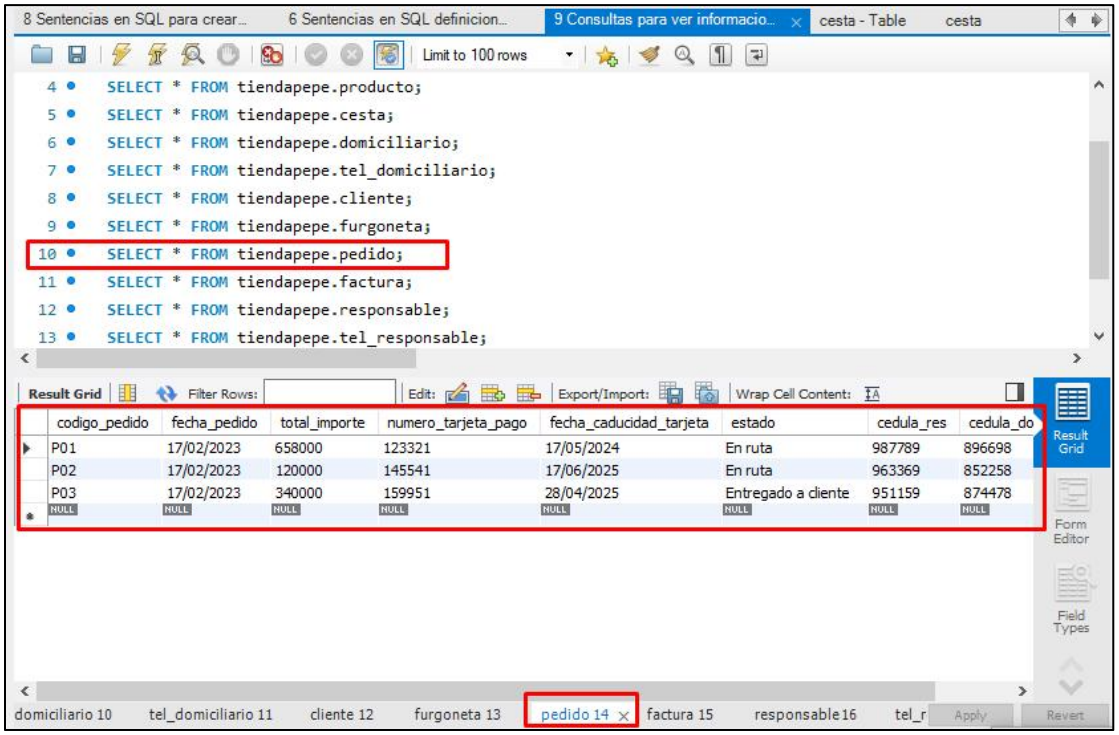


Consulta tabla furgoneta:

Cristian Camilo Alvarado Vargas
SofkaU 2023

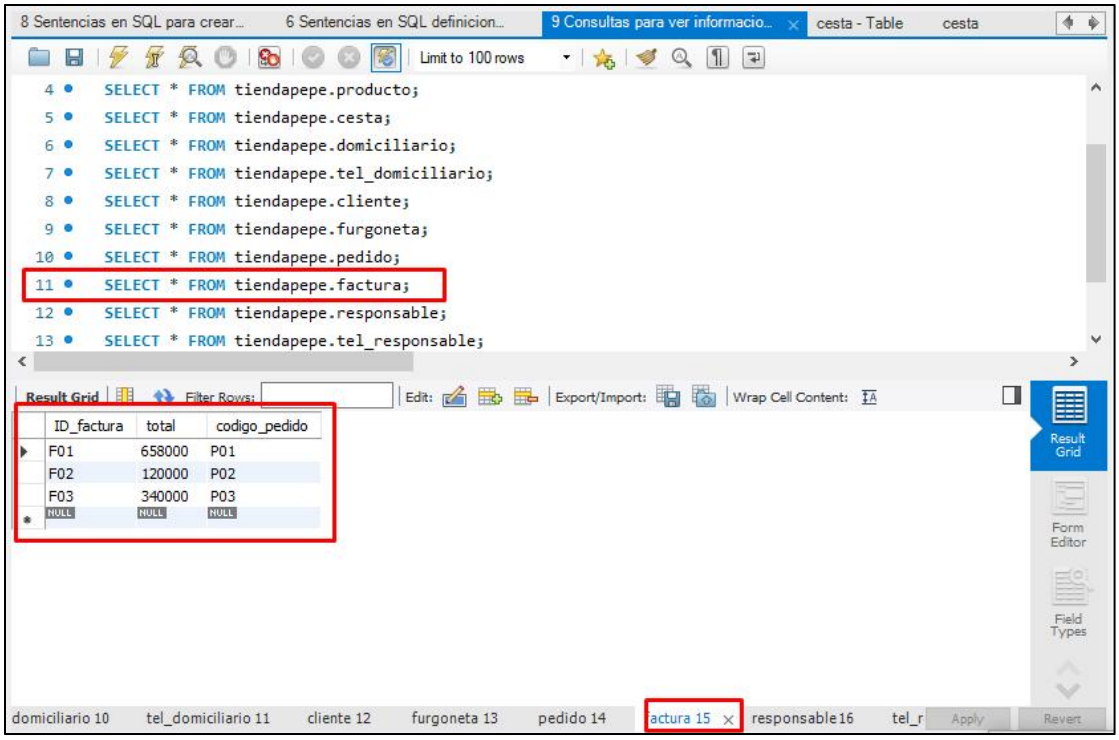


Consulta tabla pedido:

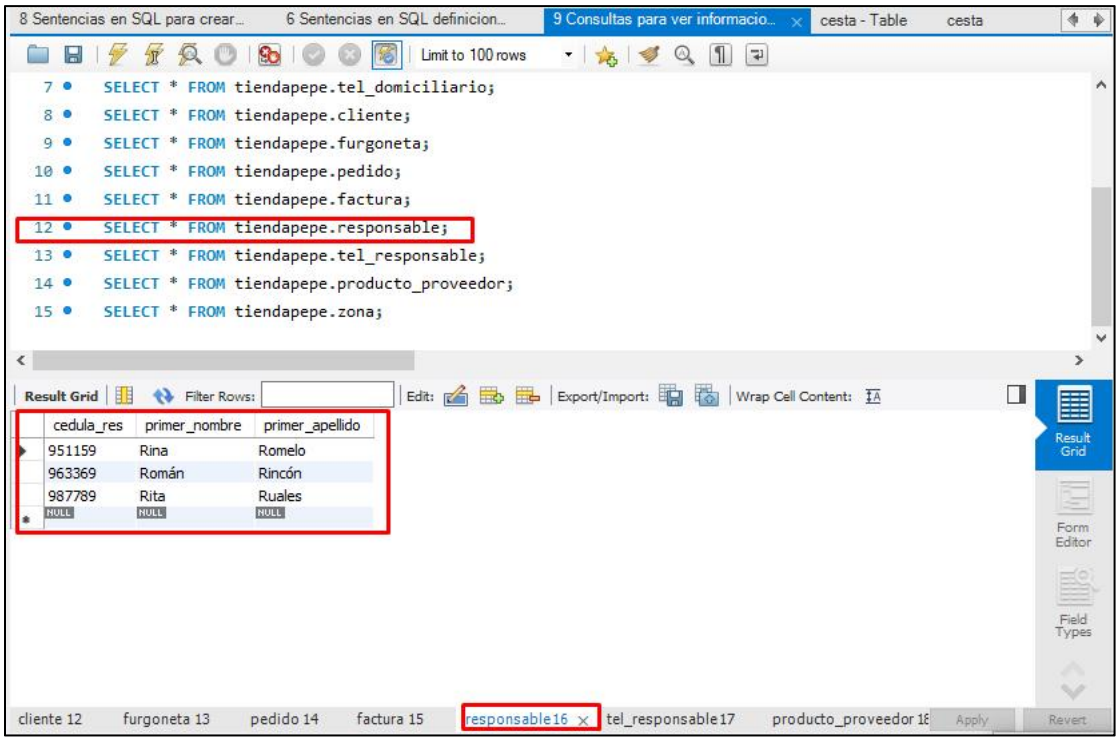


Consulta tabla factura:

Cristian Camilo Alvarado Vargas
SofkaU 2023

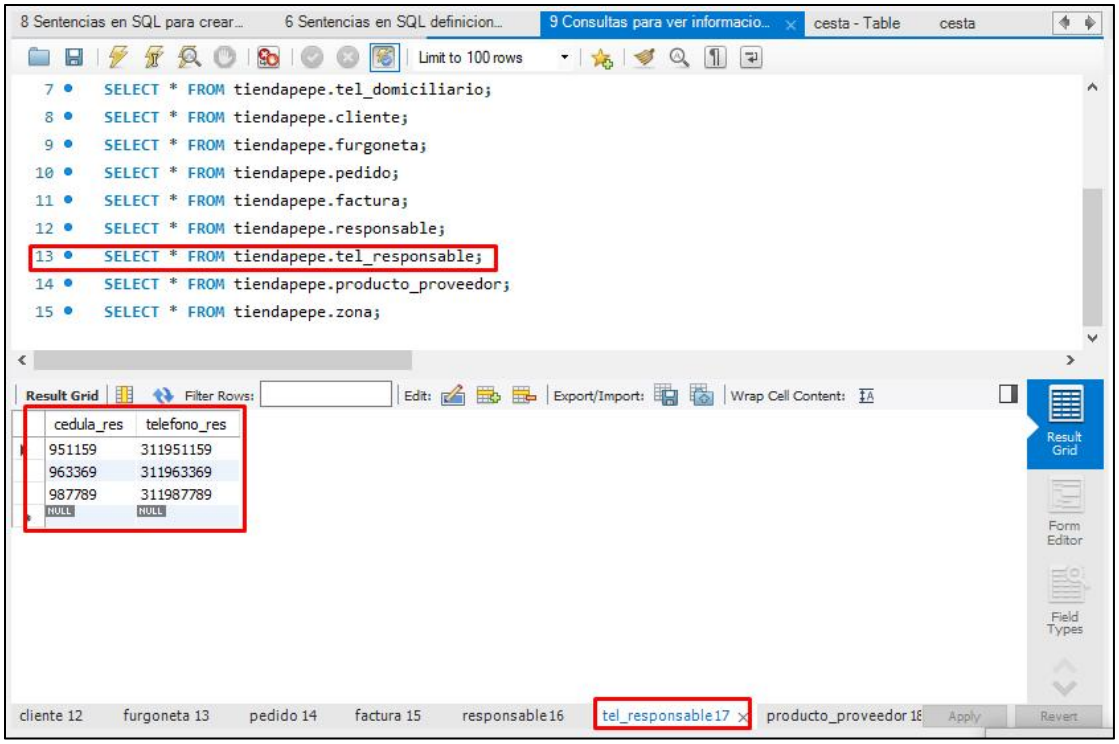


Consulta tabla responsable:

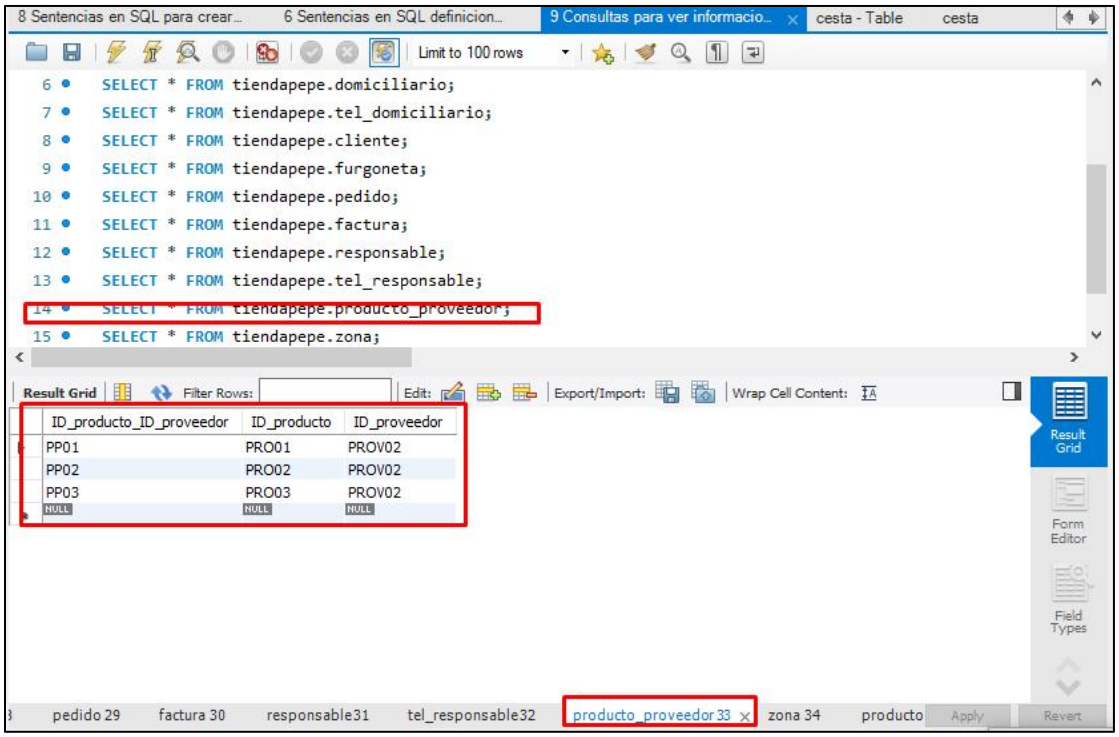


Consulta tabla tel_responsable:

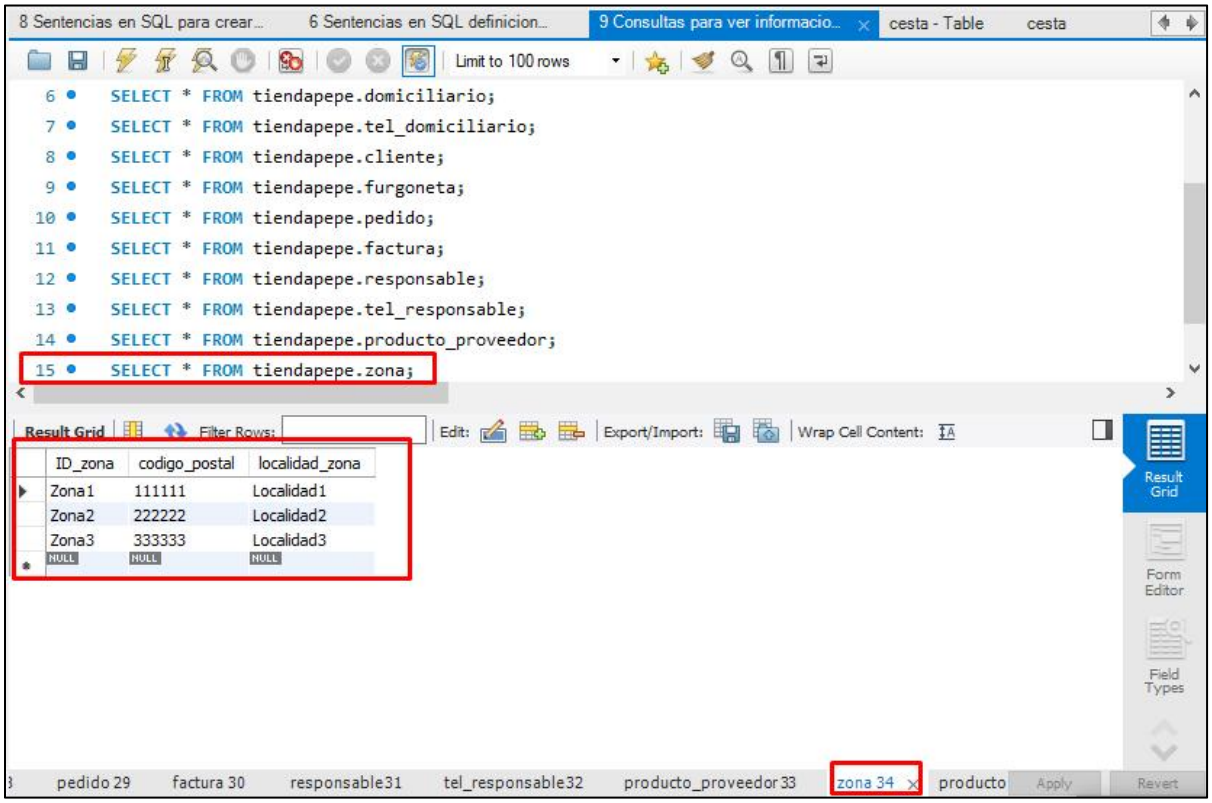
Cristian Camilo Alvarado Vargas
SofkaU 2023



Consulta tabla producto_proveedor:



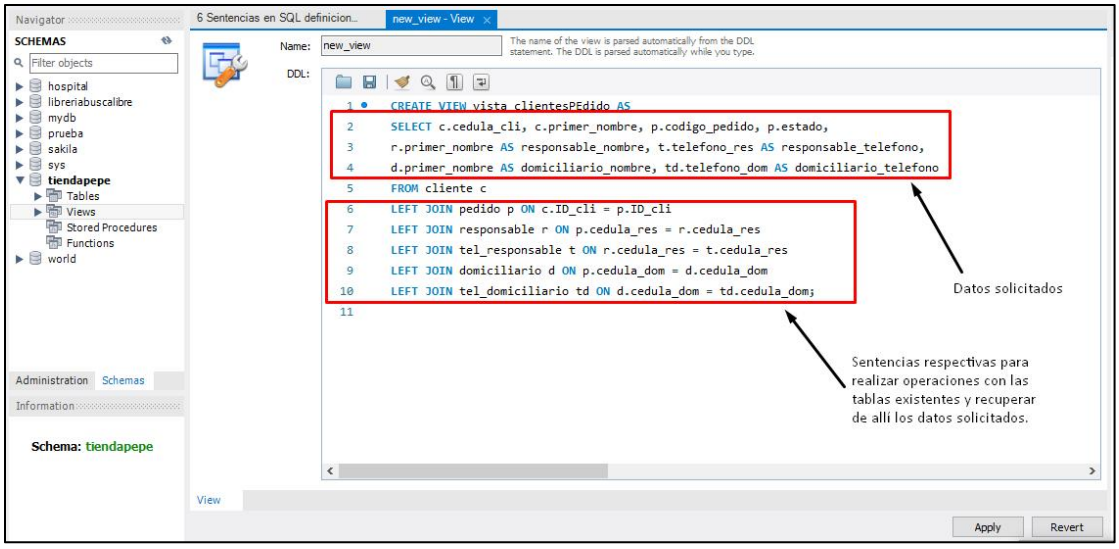
Consulta tabla zona:



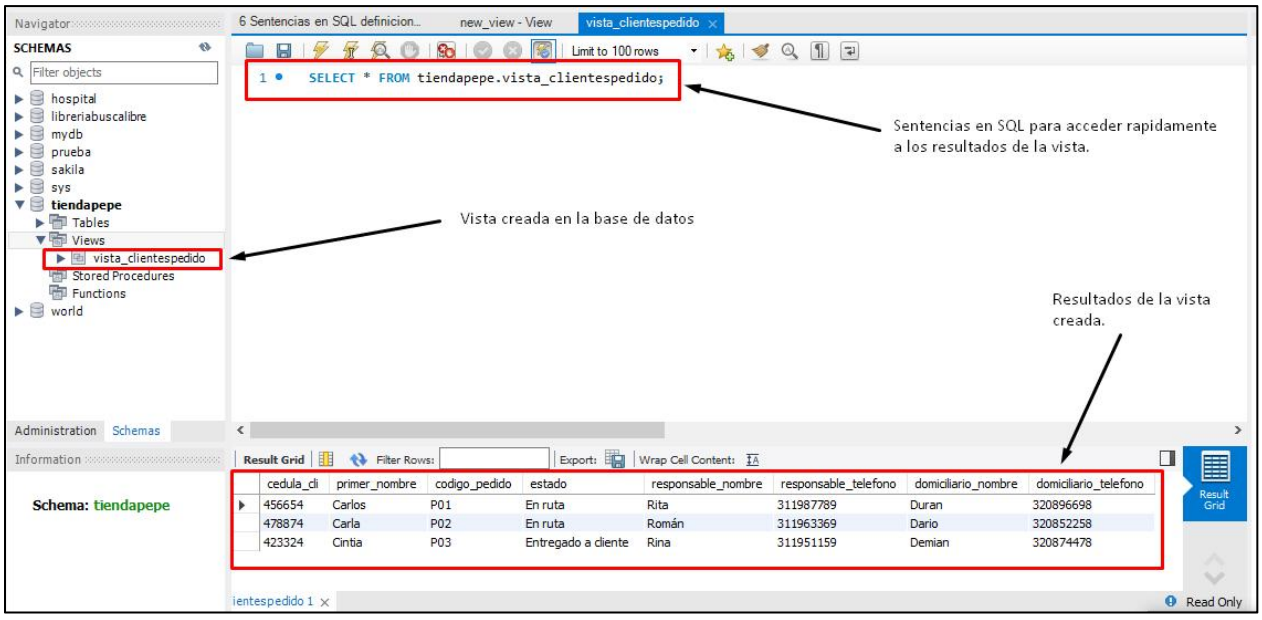
- **Generar de 4 a 6 vistas donde se evidencie lo más importante de cada ejercicio (haga una selección muy responsable de la información realmente importante según el contexto).**

Vista Número 1: vista_clientesPEdido

La vista número 1 esta asociada a la vista en la que don Pepe puede ver en tiempo real el estado de sus pedidos, es decir, en que estado esta cada pedido, quien es el encargado de empacar y quien es el encargado de entregar el domicilio. De esta forma si hay alguna novedad con el pedido don Pepe puede llamar inmediatamente a la persona respectiva para solicitar información. Por ejemplo si un cliente llama enojado diciendo que el servicio esta demorado y el estado del pedido es “En ruta” don Pepe puede saber inmediatamente a cual domiciliario llamar para solicitar más información y prestar soporte en caso de que sea necesario. En la siguiente imagen se muestran las sentencias usadas para realizar la vista deseada:



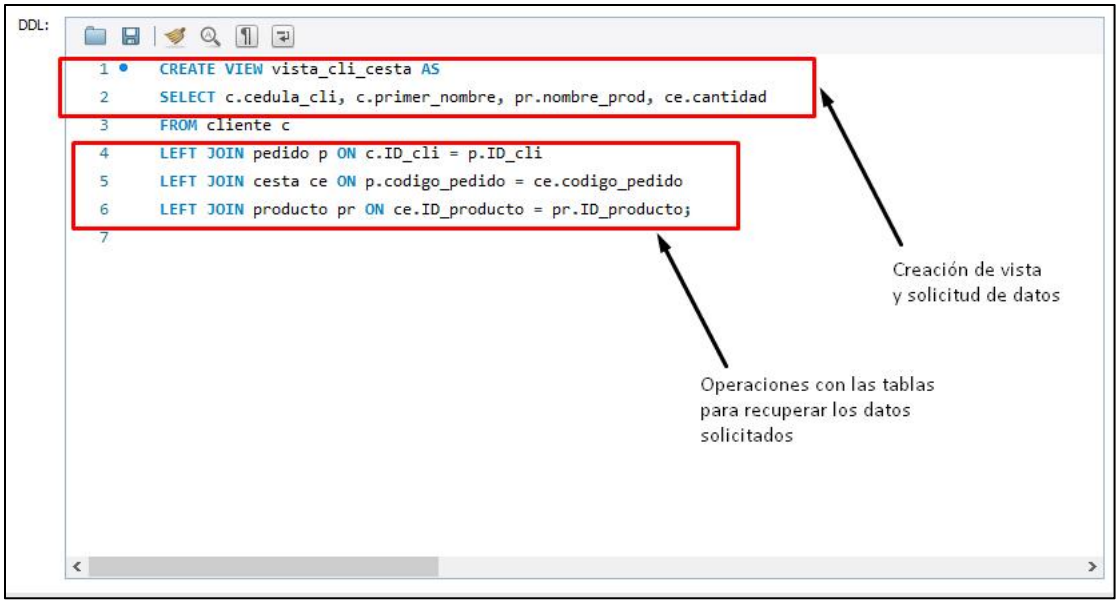
A continuación se encuentra la respuesta a la vista generada según los datos almacenados:



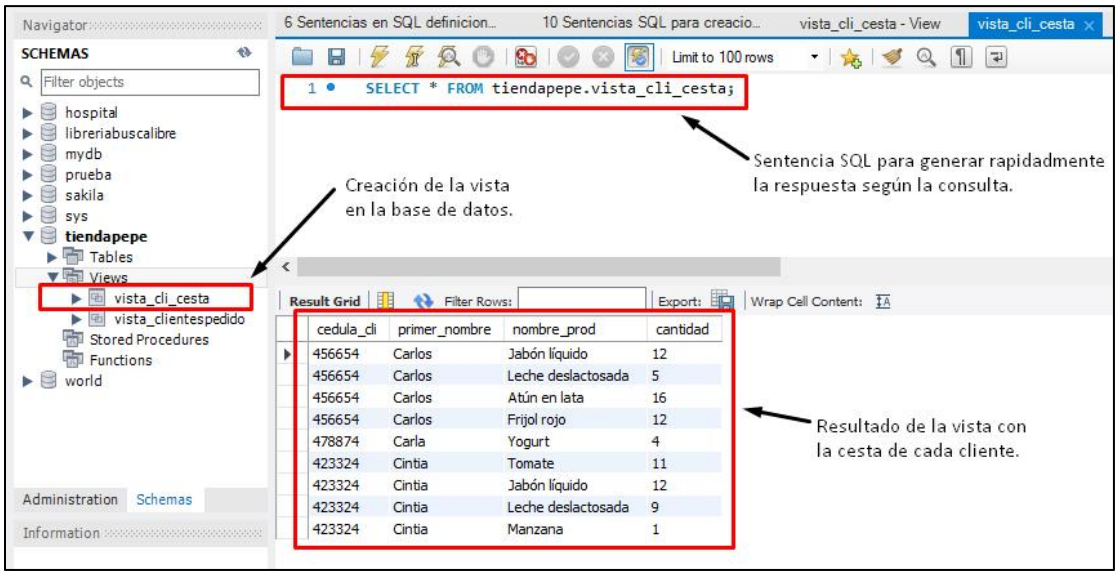
Para este caso se puede identificar que para el pedido de Carla el encargado del domicilio en Dario. Si Carla reportase alguna novedad en ese momento el número al que habría que llamar sería 320852258. De esa forma se podría prestar un pronto apoyo a Dario en caso de alguna novedad en la entrega.

Vista Número 2: vista_cli_cesta

El responsable de empacar el pedido en el almacen debe tener a la mano la información necesaria para identificar la cesta de cada pedido. Es decir debe tener un listado con los nombres y la cantidad productos que debe empacar para determinado pedido. En consecuencia se genera la siguiente vista:

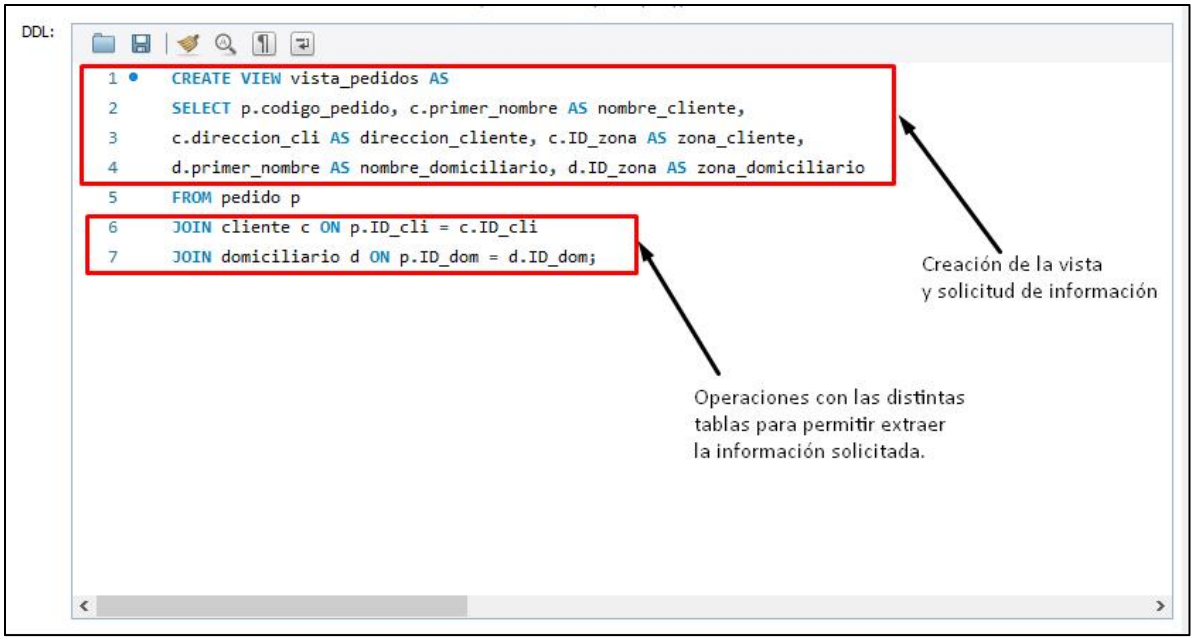


A continuación se encuentra la respuesta a la vista generada según los datos almacenados:

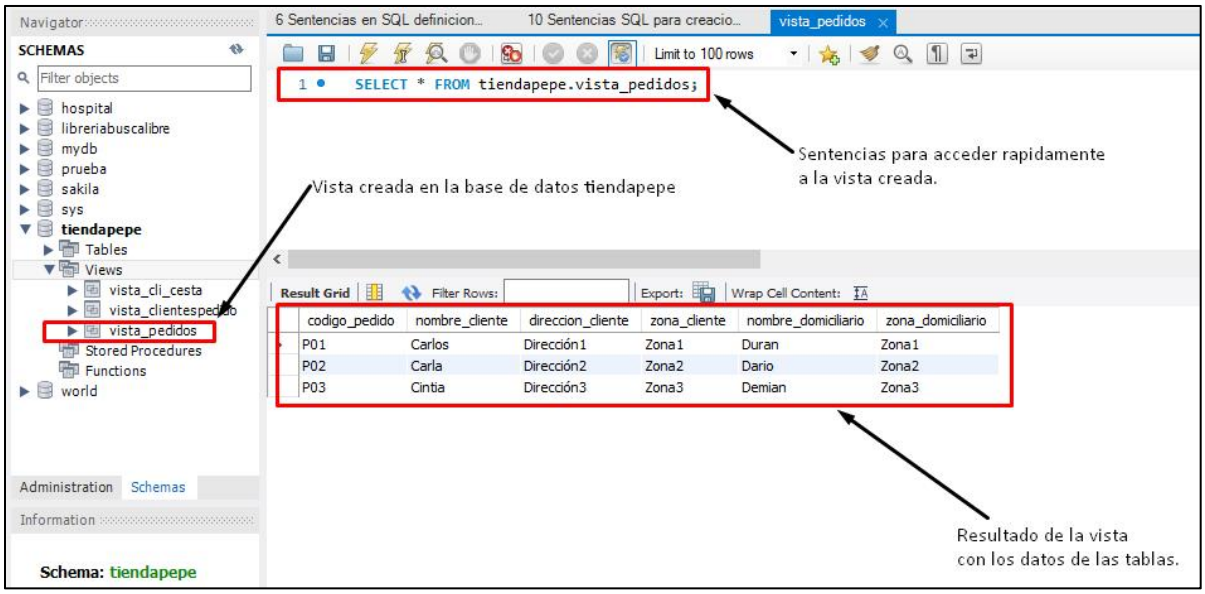


Vista Número 3: vista_pedidos

Don Pepe (o en la lógica del programa) debe revisar que el domiciliario escogido para un envío reparta en la misma zona que el cliente, ya que si el domiciliario escogido no está en la zona donde vive el cliente el pedido tendría que cancelarse. Debe haber una forma de identificar que la zona donde reparte el domiciliario coincida con la zona donde vive el cliente. En consecuencia se genera la siguiente vista:

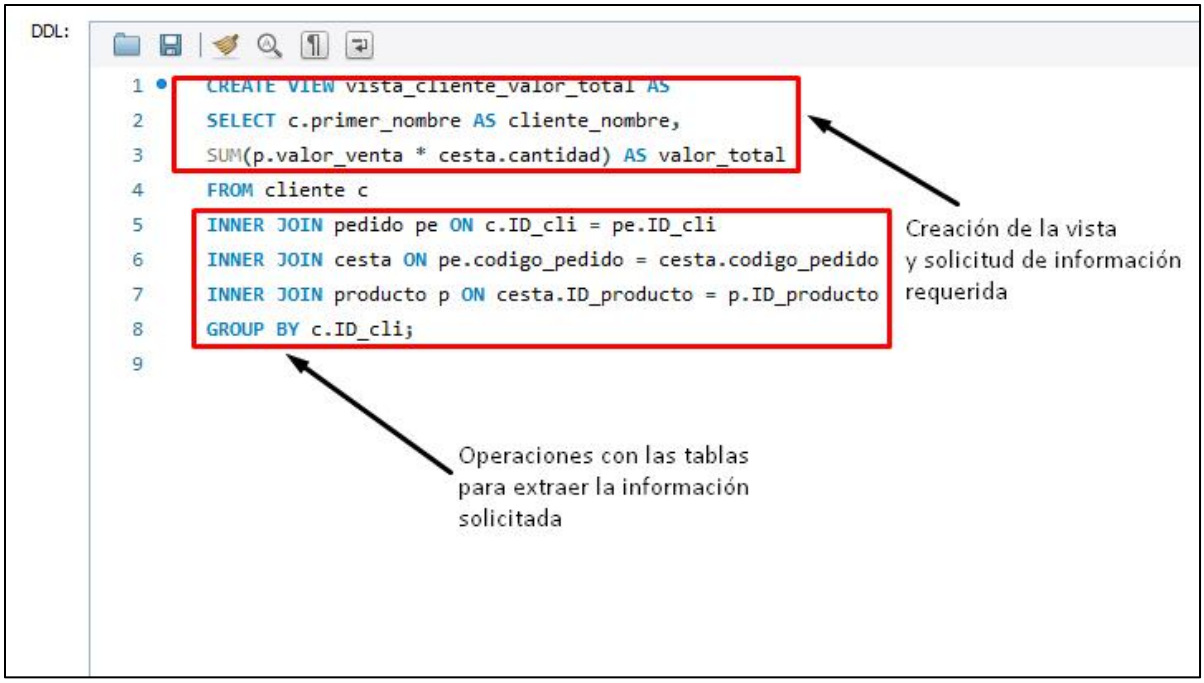


A continuación se encuentra la respuesta a la vista generada según los datos almacenados:

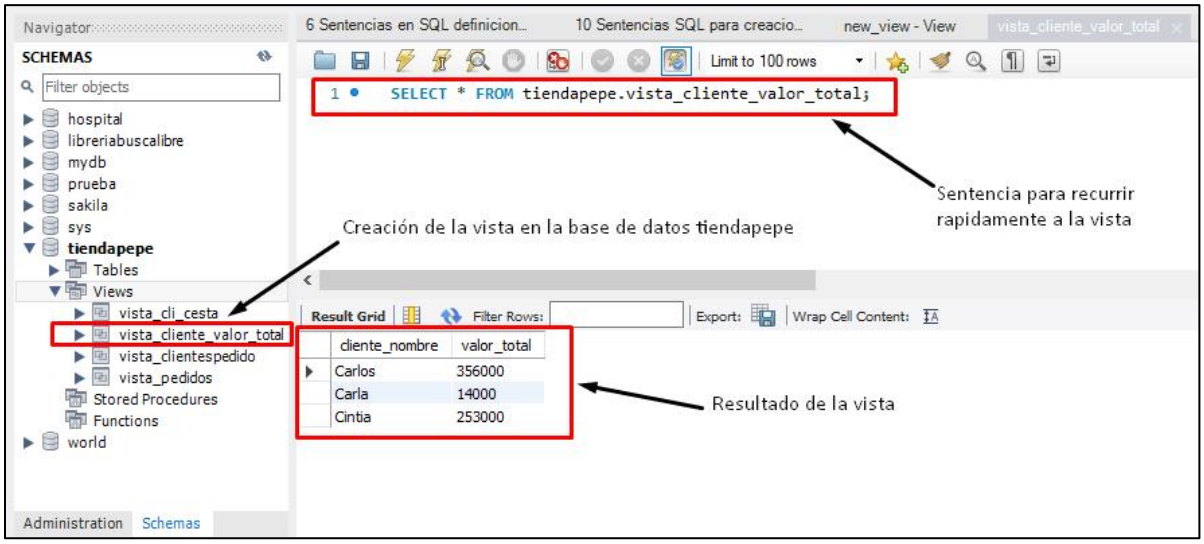


Vista Número 4: vista_cliente_valor_total

Para generar la factura al cliente se debe conocer el número total de productos vendidos y multiplicarlo por el costo de venta de cada producto para obtener el total del pedido. La información resultante es un insumo para ser informado al domiciliario, al cliente y para colocarlo en la factura que luego se le envía al cliente. En consecuencia se genera la siguiente vista:



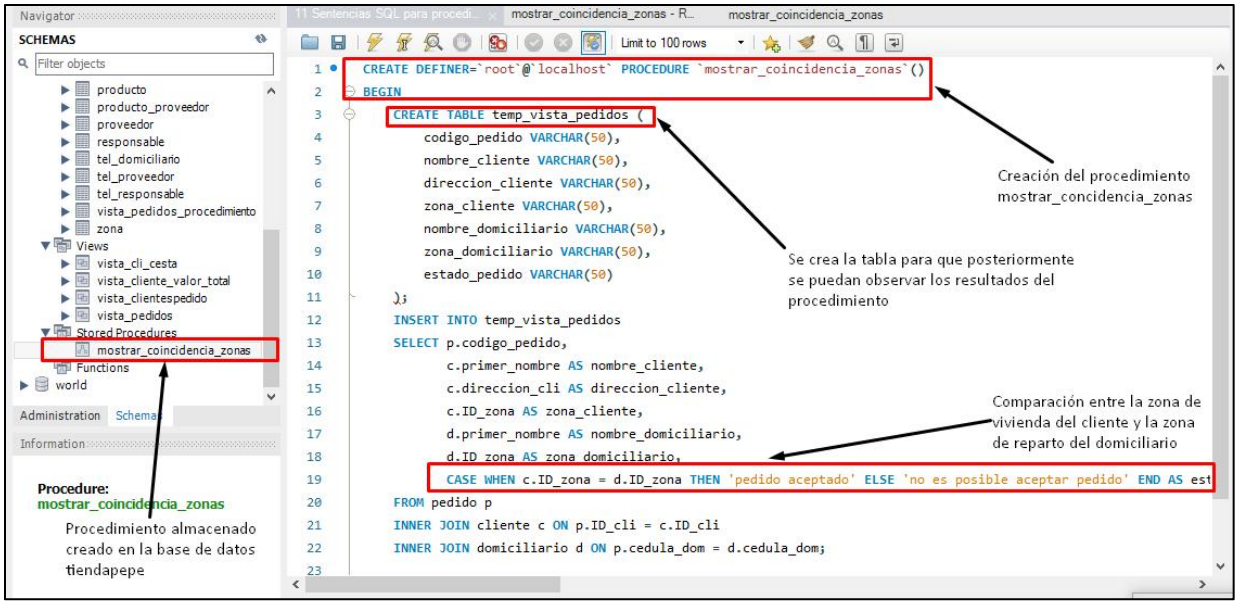
A continuación se encuentra la respuesta a la vista generada según los datos almacenados:



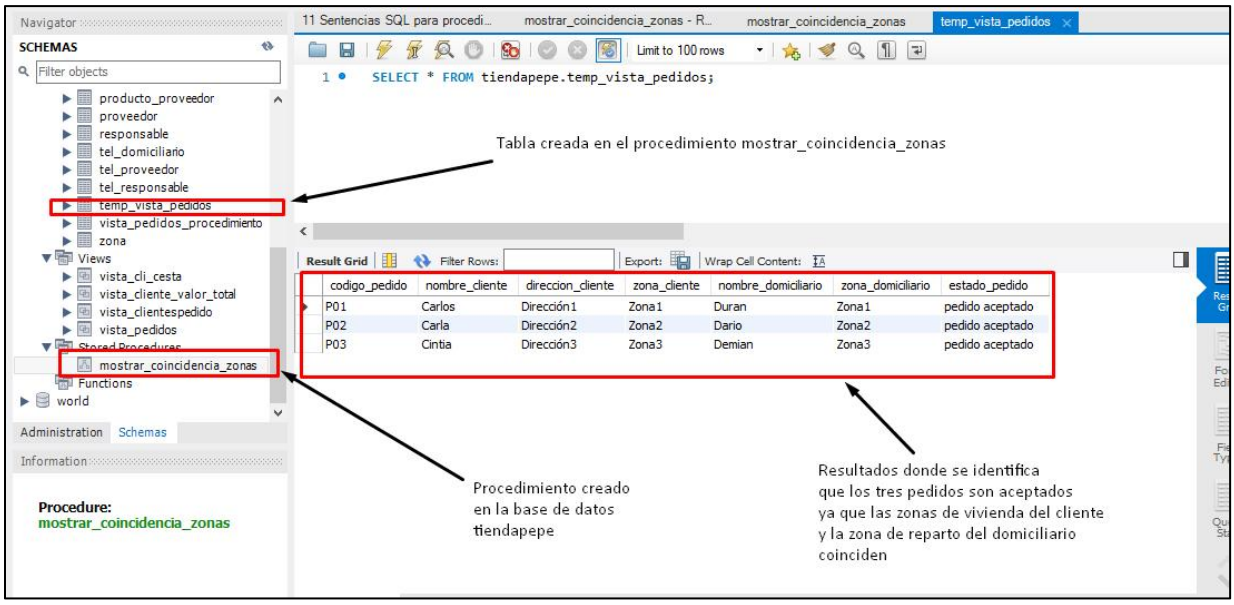
- **Generar al menos 4 procedimientos almacenados.**

Procedimiento 1: mostrar_coincidencia_zonas

Es requerido comparar las zonas en las cuales el cliente vive y el domiciliario realiza domicilios para este caso se crea un procedimiento almacenado para comparar las dos zonas y devolver un resultado indicando si es un “pedido aceptado” o “no es posible aceptar pedido” y devuelve una tabla como sigue:

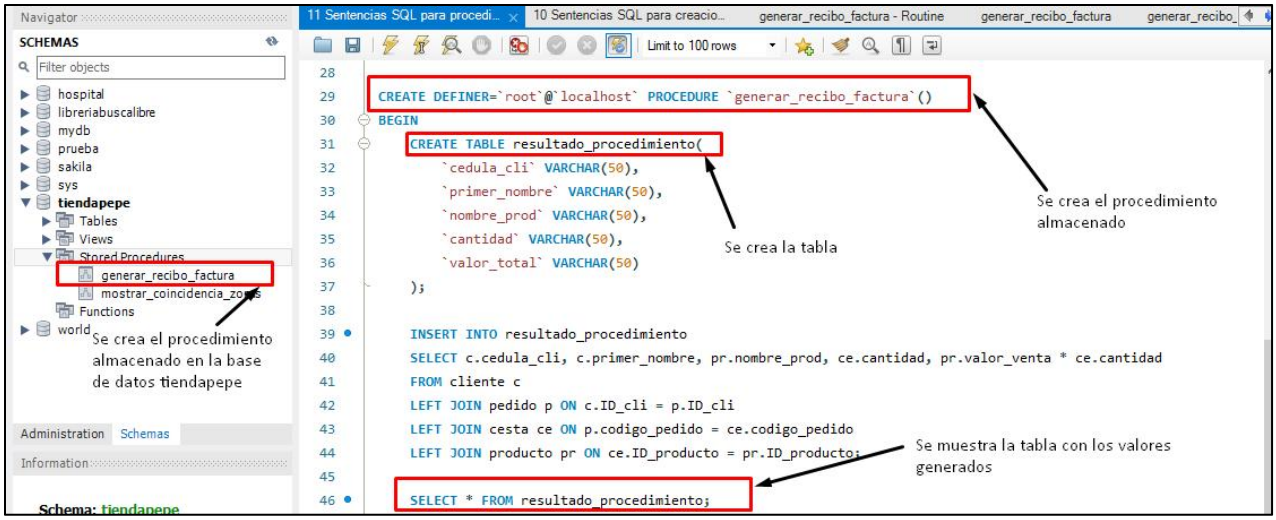


En consecuencia se generan los siguientes resultados producto de la ejecución del procedimiento almacenado:

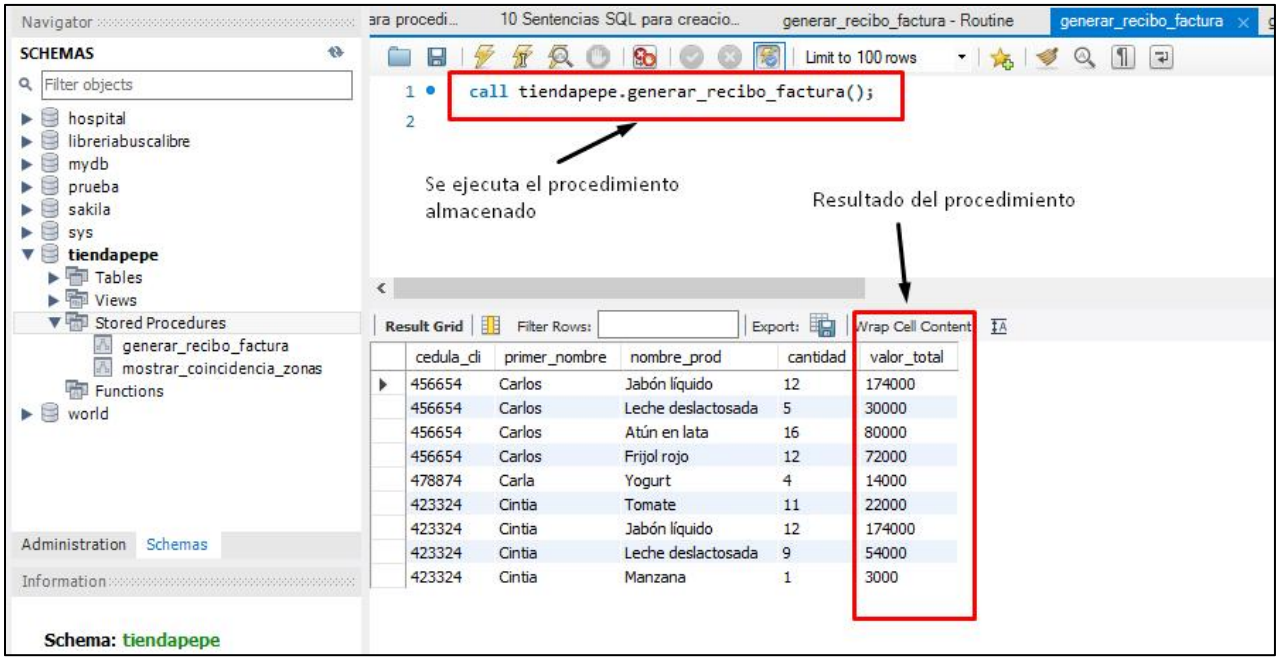


Procedimiento 2: generar_recibo_factura

Con el proposito de generar una factura completa al cliente es conveniente dar el detallado de la cantidad de productos de su pedido y el valor de la multiplicación individual del número de productos por el valor de venta de cada producto. En consecuencia se generó el siguiente procedimiento almacenado:

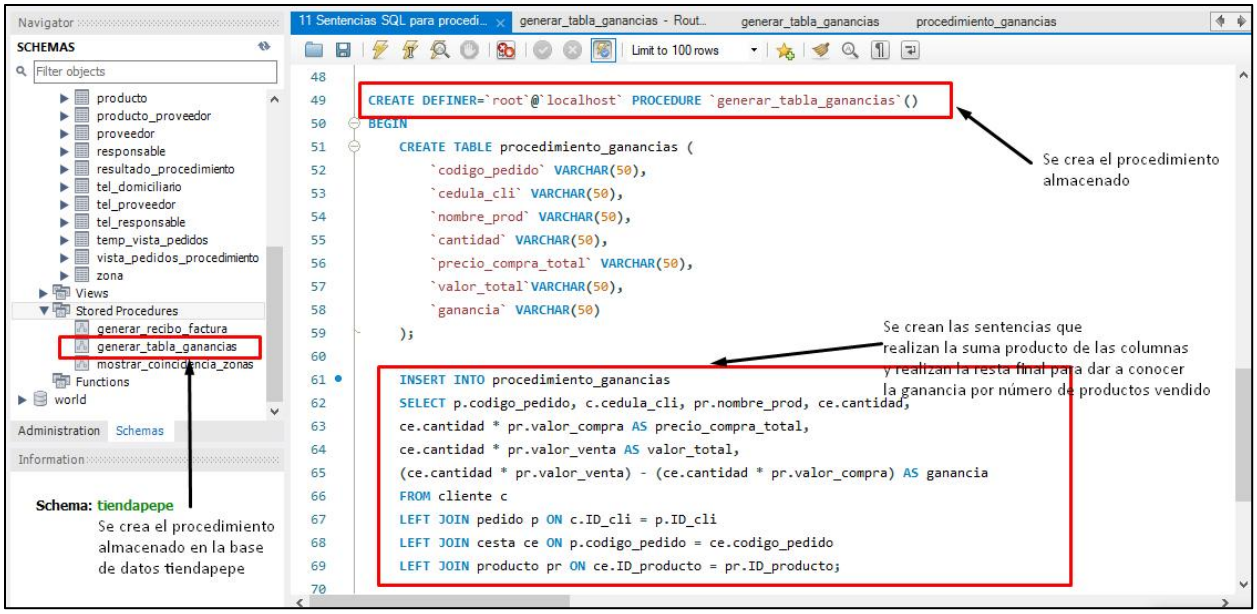


Se generan los siguientes resultados producto de la ejecución del procedimiento almacenado:

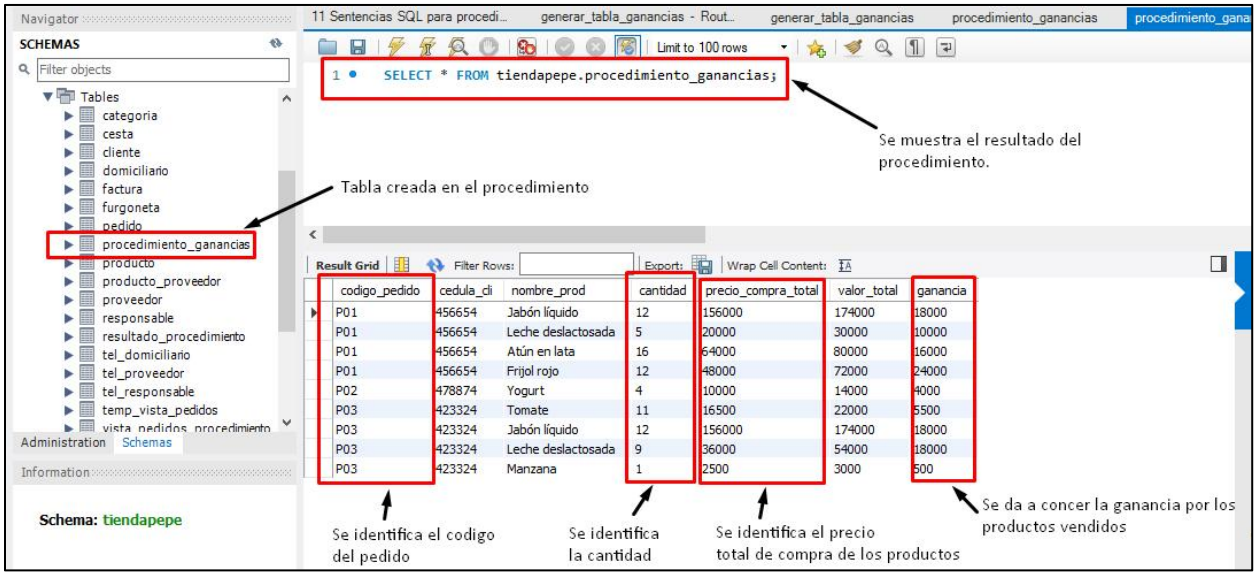


Procedimiento 3: generar_tabla_ganancias

Por otra parte es de interes de don Pepe el conocer la ganancia de cada grupo de productos vendido de cada pedido. De esta manera se puede observar cuales son los productos que generan más venta y más ganancia para la tienda. En consecuencia se generó el siguiente procedimiento almacenado para conocer la ganancia de cada producto:

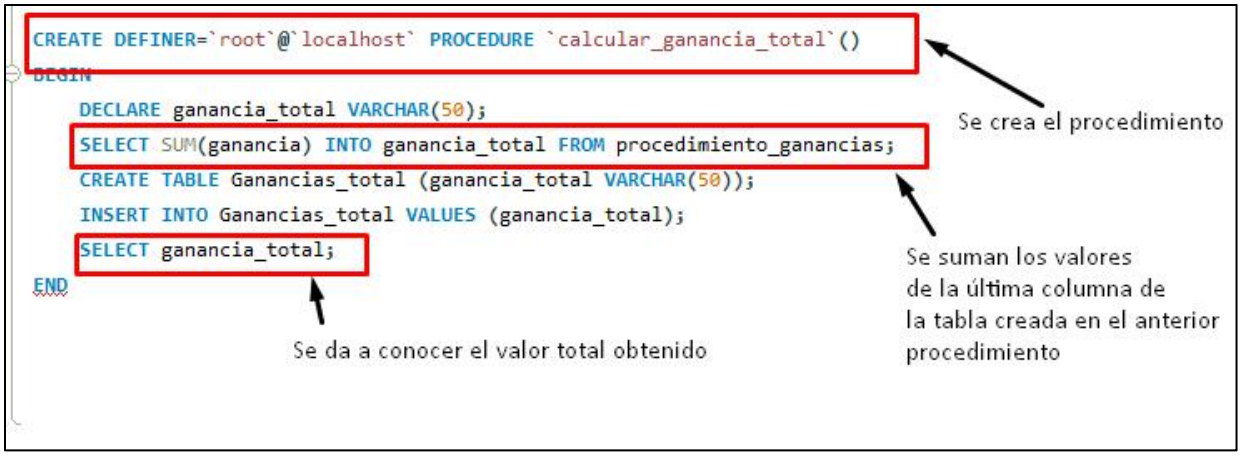


Se generan los siguientes resultados producto de la ejecución del procedimiento almacenado:

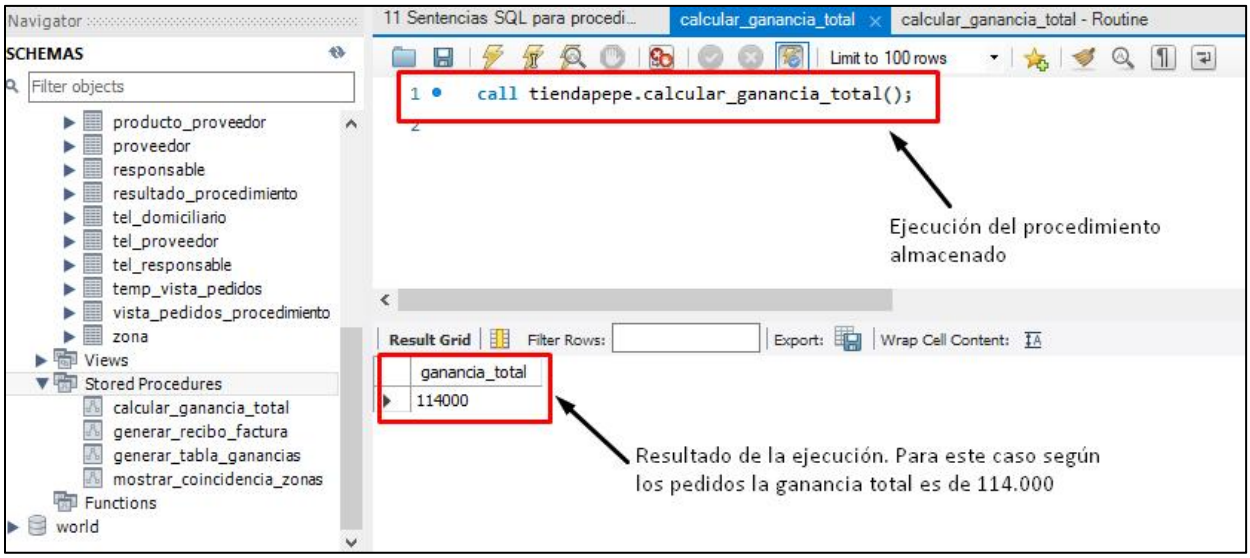


Procedimiento 4: calcular_ganancia_total

Por otra parte es de interes de don Pepe el conocer la ganancia total de todos los pedidos en curso. De esta manera se puede observar la ganancia total por la venta de todos los productos en la tienda. En consecuencia se generó el siguiente procedimiento almacenado para conocer la ganancia total por los pedidos:



Se genera el siguiente resultado producto de la ejecución del procedimiento almacenado:



- ## Generar al menos 4 triggers

Trigger 1: Actualizar_stock_producto

Con este trigger se actualiza la cantidad de productos disponibles en la tabla producto en función de la cantidad solicitada por el cliente en la tabla cesta.

```
CREATE TRIGGER actualizar_stock_producto
AFTER INSERT ON cesta
FOR EACH ROW
BEGIN
    UPDATE producto
    SET unidades_disponibles = unidades_disponibles - NEW.cantidad WHERE ID_producto = NEW.ID_producto
END;
```

Trigger 2: Actualizar_stock_producto

Con este trigger se impide eliminar un cliente de la base de datos tienda pepe si este tiene un pedido con estado pendiente.

```
CREATE TRIGGER impide_eliminacion_cliente
BEFORE DELETE ON cliente
FOR EACH ROW
BEGIN
    DECLARE num_pedidos INT
    SELECT COUNT(*) INTO num_pedidos FROM pedido WHERE ID_cli = OLD.ID_cli AND estado = 'Pendiente';
    IF num_pedidos > 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'No se puede eliminar el cliente debido a que tiene pedidos pendientes';
    END IF;
END;
```

Trigger 3: verificar_precio_venta_producto

Con este trigger se da un aviso a la persona que este ingresando nueva información sobre un producto que el precio de venta del producto al cliente es menor al precio de compra al proveedor.

```
CREATE TRIGGER verificar_precio_venta_producto
BEFORE INSERT OR UPDATE ON producto
FOR EACH ROW
BEGIN
    IF NEW.valor_venta < NEW.valor_compra THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'El precio de venta debe ser mayor o igual al precio de compra';
    END IF;
END;
```

Trigger 4: borrar_registros_relacionados

Con este trigger cuando se elimina un pedido se borran los datos asociados en la tabla canasta, de esta forma se le indica al responsable en el almacen que ya no se deben empacar los productos asociados a esa cesta.

```
CREATE TRIGGER borrar_registros_relacionados
AFTER DELETE ON pedido
FOR EACH ROW
BEGIN
    DELETE FROM cesta WHERE codigo_pedido = OLD.codigo_pedido;
END;
```

- **Poblar la base de datos (50 registros por tabla) utilizando una conexión desde Java.**
- **Al terminar el ejercicio responda ¿ Está conforme con el resultado obtenido según el contexto o cree que hubiera obtenido un mejor resultado con una base de datos no relacional?**

Si estoy conforme con el resultado, el uso base de datos relacionadas permite generar interacciones entre las tablas, ahorrar tiempo en las consultas, generar información nueva a partir de los datos originales, evitar la repetición de registros, entre otras que con una base de datos no relacional no sería posible.