Proyecto Base de Datos: Entrega II

Integrantes: Jose Joaquin Álvarez, Enrique Molinare León
Fecha: 26/05/21
Sección: 1
Grupo: 131

2.1 Crear modelo

Antiguo modelo:

- **Usuarios**(id, nombre, rut, edad)
- **Direcciones_usuarios**(id, id_usuario, id_direccion, id_comuna)
- **Compras**(id, id_comprador, id_direccion, id_tienda)
- **Productos_por_compra**(id, id_compra, id_producto, cantidad)
- **Productos**(id, nombre, precio, descripción, tipo, ancho, alto, largo, peso, fecha_de_expiracion, categoria, metodo_conservacion, duracion_ambiente)
- **Tiendas**(id, id_direccion, nombre, id_comuna, id_jefe)
- **Despacho_Tiendas**(id, id_tienda, id_direccion_despacho, id_comuna)
- **Productos_tienda**(id, id_producto, id_tienda)
- **Personal**(id, nombre, rut, sexo, id_tienda, edad)
- **Direcciones**(id, dirección)
- **Comunas**(id, comuna)

Luego de realizar análisis de dependencias, el modelo quedo asi:

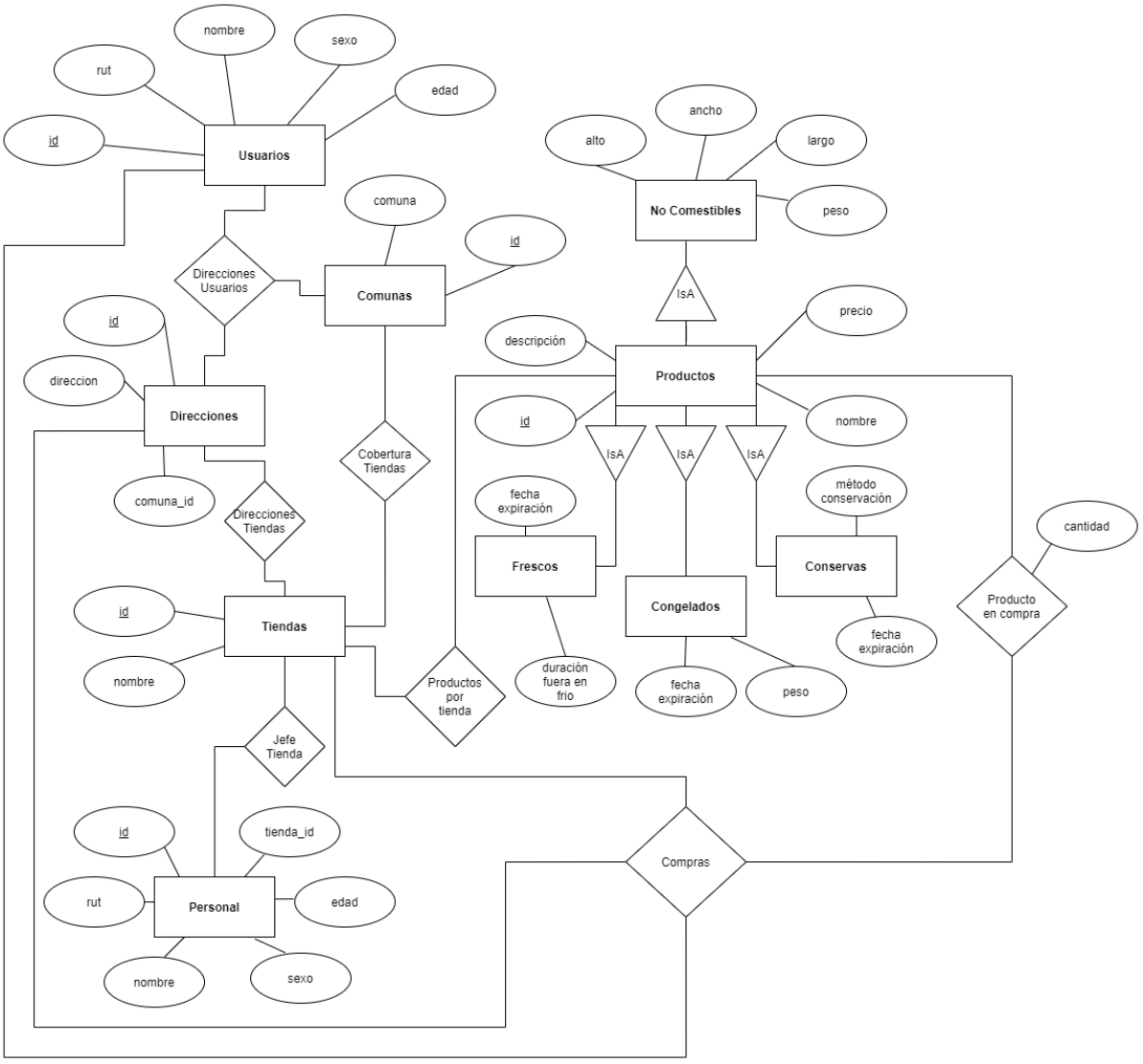

Figura 1: Diagrama Entidad - Relación.

Ahora, presentamos nuestro esquema relacional:

- Usuarios(id int, nombre varchar(150), rut varchar(150), edad int, sexo varchar(20))
    - Llave Primaria: id
- Direcciones_usuarios(usuario_id int, direccion_id int)
    - Llave primaria (usuario_id, direccion_id)
    - LLave Foránea (usuario_id) Referencia Usuarios (id)
    - Llave Foránea (direccion_id) Referencia Direcciones (id)
- Comunas(id int, comuna varchar (50))
    - Llave Primaria (id)
- Direcciones(id int, dirección varchar(150), comuna_id int)
    - Llave primaria (id)
    - Llave Foránea (comuna_id) Referencia comunas (id));
- Tiendas(id int, nombre varchar (100), direccion_id int)
    - Llave primaria (id)
    - Llave Foránea (direccion_id) Referencia direcciones(id));
- Direcciones_tiendas(tienda_id int, direccion_id int)
    - Llave primaria (tienda_id, direccion_id)
    - Llave Foránea (tienda_id) Referencia tiendas (id)
    - Llave Foránea (direccion_id) Referencia direcciones (id));
- Cobertura_Tiendas(tienda_id int, comuna_cobertura_id int)
    - Llave primaria (tienda_id, comuna_cobertura_id)
    - Llave Foránea (tienda_id) Referencia tiendas (id)
    - Llave Foránea (comuna_cobertura_id) Referencia comunas (id));
- Productos(id int, nombre varchar(150), descripción varchar(150), precio real)
    - Llave primaria (id)
- No_Comestibles(producto_id int, alto real, largo real, ancho real, peso real)
    - Llave primaria (producto_id)
    - Llave Foránea (producto_id) Referencia productos (id));
- Congelados(producto_id int, fecha_de_expiracion date, peso real)
    - Llave primaria (producto_id)
    - Llave Foránea (producto_id) Referencia productos (id));
- Conservas(producto_id int, fecha_de_expiracion date, metodo_de_conservacion varchar (100))
    - Llave primaria (producto_id)
    - Llave Foránea (producto_id) Referencia productos (id));
- Frescos(producto_id int, fecha_de_expiracion date, duracion_fuera_frio int)
    - Llave primaria (producto_id)

    - Llave Foránea (producto_id) Referencia productos (id));
- Productos_tienda(tienda_id int, producto_id int)
    - Llave primaria (tienda_id, producto_id)
    - Llave Foránea (tienda_id) Referencia tiendas (id)
    - Llave Foránea (producto_id) Referencia productos (id));
- Compras(id int, comprador_id int, direccion_id int , tienda_id int)
    - Llave primaria (id)
    - Llave Foránea (tienda_id) Referencia tiendas (id),
    - Llave Foránea (comprador_id) Referencia usuarios (id)
    - Llave Foránea (direccion_id) Referencia direcciones (id))
- Producto_por_compra(compra_id int, producto_id int, cantidad rea)
    - Llave primaria (compra_id, producto_id)
    - Llave Foránea (compra_id) Referencia compras(id)
    - Llave Foránea (producto_id) Referencia productos(id));
- Personal(id int, rut varchar(20), nombre varchar(100), edad int, sexo varchar(100), tienda_id int)
    - Llave primaria (id)
    - Llave Foránea (tienda_id) Referencia tiendas (id));
- Jefe_tiendas(tienda_id int , personal_id int)
    - Llave primaria (tienda_id)
    - Llave Foránea (tienda_id) Referencia tiendas (id)
    - Llave Foránea (personal_id) Referencia personal (id))

2.2 Justificar modelo

Analizaremos el modelo presentado en la primera entrega, veremos sus dependencias y arreglaremos las anomalías que poseen sus relaciones, para luego dejar el esquema en su forma normal de Boyce-Codd (BCFN).

Modo incorrecto

Usuarios(id,rut,nombre,sexo,edad,direccion,comuna)

id → rut, nombre, sexo, edad, direccion, comuna
rut → nombre, sexo, edad

MALO: Genera un problema de redundancia porque una persona puede tener N direcciones.

Tiendas(id, nombre, jefe, dirección, comuna, cobertura)

SUPUESTOS: Una persona solo puede ser jefe de una sola tienda, dos tiendas no pueden tener el mismo nombre.

id → nombre, jefe, dirección, comuna, cobertura
nombre → jefe, dirección, comuna

MALO: posee redundancia al tener M coberturas.

Productos(id, nombre, precio, descripción, tipo, ancho, alto, largo, peso, fecha_de_expiracion, categoria, metodo_conservacion, duracion_ambiente)

id → nombre, precio, descripción, …, duracion_ambiente

MALO: existirán productos que serán comestibles y otros no, lo que implicaría que si dejamos la relación tal como está, existirán NULLS.

Compras(id, comprador, dirección, producto, cantidad, tienda)

id → comprador, dirección, producto, cantidad, tienda

MALO: Genera redundancia dado que la relación entre comprador y producto es N : N.

Modo Correcto

Usuarios(<u>id</u>, rut, nombre, sexo, edad)

rut → id, nombre, sexo, edad
id → rut, nombre, sexo, edad

Direcciones_Usuarios(<u>usuario_id, direccion_id</u>)

usuario_id, direccion_id → usuario_id, direccion_id

Comunas(<u>id</u>, comuna)

id → comuna
comuna → id

Direcciones(<u>id</u>, dirección, comuna_id)

id → dirección, comuna_id
dirección, comuna_id → id

En conclusión:

usuario_id, direccion_id → nombre, rut, sexo, edad, direccion, comuna

Dado que en todas las dependencias de cada relación, su lado izquierdo corresponden a superllaves (es decir, que son suficientes para abarcar toda su información), estas relaciones están de la forma BCNF.

Tiendas(<u>id</u>, nombre, direccion_id)

id → nombre, direccion_id
nombre, direccion_id → id

Direcciones_Tiendas(<u>tienda_id, direccion_id</u>)

tienda_id, direccion_id → tienda_id, direccion_id

Cobertura_Tiendas(<u>tienda_id, comuna_cobertura_id</u>)

tienda_id, comuna_cobertura_id → tienda_id, comuna_cobertura_id

En conclusión:

tienda_id → nombre, jefe, direccion_id, comuna_cobertura_id

el lado izquierdo de las dependencias de las relaciones son superllaves, por lo tanto, son BCNF.

Productos(<u>id</u>, nombre, descripcion, precio)

id → nombre, descripcion, precio

No_Comestibles(<u>producto_id</u>, alto, largo, ancho, peso)

producto_id → alto, largo, ancho, peso

Congelados(<u>producto_id</u>, fecha_de_expiracion, peso)

producto_id → peso, fecha_de_expiracion

Conservas(<u>producto_id</u>, fecha_de_expiracion, metodo_de_conservacion)

producto_id → metodo_de_conservacion, fecha_de_expiracion

Frescos(<u>producto_id</u>, fecha_de_expiracion, duracion_fuera_frio)

producto_id → duracion_fuera_frio, fecha_de_expiracion

Al dividir en todas estas secciones los productos, evitaremos el uso de Nulls y la anomalía al borrar algún producto (eliminación de forma cascada al rescate). BCNF

Ahora, para clasificarlos por tienda:

Productos_tienda(<u>tienda_id, product_id</u>)

tienda_id, product_id → tienda_id, product_id

Compras(<u>id</u>, comprador_id, direccion_id, tienda_id)

Supuesto: en una dirección pueden vivir más de una persona por lo que es incorrecto direccion_id, tienda_id → comprador_id

id → comprador_id, direccion_id, tienda_id

Producto_por_compra(<u>compra_id, producto_id</u>, cantidad)

compra_id, producto_id → cantidad

Personal(<u>id</u>, rut, nombre, edad, sexo, tienda_id)

SUPUESTO: una persona no puede trabajar en dos tiendas

id → nombre, rut, edad, sexo, tienda_id
rut → id, nombre, edad sexo, tienda_id

Jefe_tiendas(tienda_id, personal_id)

tienda_id, personal_id → tienda_id, personal_id

Cumplen la regla de poseer todas sus dependencias como superllaves, por lo tanto, son todos BCNF.

2.3 Consultas en SQL

A continuación presentamos las consultas pedidas:

1) SELECT tiendas.nombre AS nombre_tienda, comunas.comuna FROM tiendas, cobertura_tiendas, comunas WHERE tiendas.id = cobertura_tiendas.tienda_id AND cobertura_tiendas.comuna_cobertura_id = comunas.id;

2) Dependerá de la comuna elegida = INPUT

   SELECT personal.nombre FROM jefe_tiendas, tiendas, direcciones_tiendas, personal, direcciones, comunas WHERE jefe_tiendas.tienda_id = tiendas.id AND jefe_tiendas.personal_id = personal.id AND direcciones_tiendas.direccion_id = direcciones.id AND direcciones.comuna_id = comunas.id AND direcciones_tiendas.tienda_id = tiendas.id AND comunas.comuna = **INPUT**;

3) Dependerá del tipo de alimento que se pida:

   SELECT tiendas.id, tiendas.nombre, CONCAT(direcciones.dirección, ‘, ’, comunas.comuna) AS direccion FROM tiendas, productos_tienda, **INPUT**, direcciones, comunas WHERE tiendas.id = productos_tienda.tienda_id AND productos_tienda.producto_id = **INPUT**.producto_id AND tiendas.direccion_id = direcciones.id AND direcciones.comuna_id = comunas.id GROUP BY tiendas.id, direcciones.dirección, comunas.comuna ORDER BY tiendas.id ASC;

   Dónde **INPUT** será igual a**:**

   - **no_comestibles** si se escogen no comestibles.

   - **SELECT productos.id AS producto_id, productos.nombre FROM productos, frescos, congelados, conservas WHERE productos.id = frescos.producto_id OR productos.id = congelados.producto_id OR productos.id = conservas.producto_id GROUP BY productos.id ORDER**

BY productos.id ASC) si se escogen todos los comestibles. NOTA: hay que agregarle a esta relación **AS comestibles**, para después llamarla como comestibles en el segundo input. (PDD: Esta consulta se demora un par de segundo mas)

- **frescos** si se escogen comestibles frescos.
- **congelados** si se escogen comestibles congelados.
- **conservas** si se escogen comestibles en conserva

4) Depende de la desccripción ingresada en el **INPUT**

   SELECT usuarios.nombre FROM compras, producto_por_compra, usuarios, productos WHERE usuarios.id = compras.comprador_id AND producto_por_compra.compra_id = compras.id AND producto_por_compra.producto_id = productos.id AND productos.descripción = **INPUT** GROUP BY usuarios.nombre;

5) Dependerá de la comuna ingresada **INPUT**:

   SELECT AVG(personal.edad) AS promedio_edad_personal FROM tiendas, direcciones_tiendas, direcciones, comunas, personal WHERE tiendas.direccion_id = direcciones.id AND direcciones.comuna_id = comunas.id AND personal.tienda_id = tiendas.id AND comunas.comuna = **INPUT**;

6) Depende del input ingresado
   a) Si el input es no comestibles:
      i) SELECT tiendas.id AS id , tiendas.nombre, SUM(producto_por_compra.cantidad) AS Cantidad_Ventas FROM no_comestibles, producto_por_compra, compras, tiendas, productos WHERE no_comestibles.producto_id = producto_por_compra.producto_id AND compras.id = producto_por_compra.compra_id AND compras.tienda_id = tiendas.id AND no_comestibles.producto_id = productos.id GROUP BY tiendas.id ORDER BY cantidad_ventas desc LIMIT 5;
   b) Si el input es comestibles
      i) SELECT tiendas.id AS id , tiendas.nombre, SUM(producto_por_compra.cantidad) AS Cantidad_Ventas FROM producto_por_compra, compras, tiendas, productos, (SELECT productos.id AS producto_id, productos.nombre FROM productos, frescos, congelados, conservas WHERE productos.id = frescos.producto_id OR productos.id = congelados.producto_id OR productos.id = conservas.producto_id GROUP BY productos.id ORDER BY productos.id ASC) AS comestibles WHERE comestibles.producto_id = producto_por_compra.producto_id AND compras.id = producto_por_compra.compra_id AND compras.tienda_id = tiendas.id AND comestibles.producto_id =

productos.id GROUP BY tiendas.id ORDER BY cantidad_ventas desc LIMIT 5;

c) Si el input es Congelados
    i) SELECT tiendas.id AS id , tiendas.nombre, SUM(producto_por_compra.cantidad) AS Cantidad_Ventas FROM congelados, producto_por_compra, compras, tiendas, productos WHERE congelados.producto_id = producto_por_compra.producto_id AND compras.id = producto_por_compra.compra_id AND compras.tienda_id = tiendas.id AND congelados.producto_id = productos.id GROUP BY tiendas.id ORDER BY cantidad_ventas desc LIMIT 5;
d) Si el input es Frescos
    i) SELECT tiendas.id AS id , tiendas.nombre, SUM(producto_por_compra.cantidad) AS Cantidad_Ventas FROM frescos, producto_por_compra, compras, tiendas, productos WHERE frescos.producto_id = producto_por_compra.producto_id AND compras.id = producto_por_compra.compra_id AND compras.tienda_id = tiendas.id AND frescos.producto_id = productos.id GROUP BY tiendas.id ORDER BY cantidad_ventas desc LIMIT 5;
e) Si el input es Conservas
    i) SELECT tiendas.id AS id , tiendas.nombre, SUM(producto_por_compra.cantidad) AS Cantidad_Ventas FROM conservas, producto_por_compra, compras, tiendas, productos WHERE conservas.producto_id = producto_por_compra.producto_id AND compras.id = producto_por_compra.compra_id AND compras.tienda_id = tiendas.id AND conservas.producto_id = productos.id GROUP BY tiendas.id ORDER BY cantidad_ventas desc LIMIT 5;

2.5 Página Web

Se implementó una página web con PHP en el servidor, la cual se pueden chequear los archivos dentro de la carpeta /home/grupo131/Sites. Puedes ingresar a nuestra pagina a traves del siguiente link:

http://codd.ing.puc.cl/~grupo131/index.php?