Bike Store

Descripción BD:

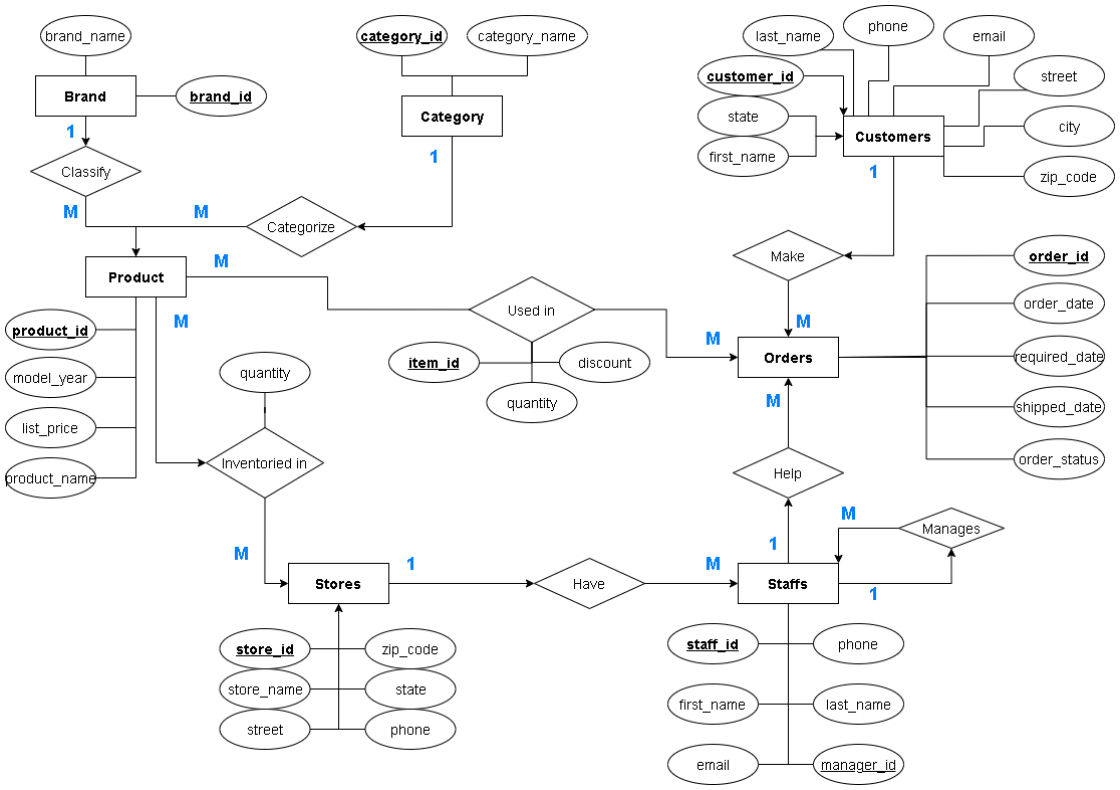
La base de datos de ejemplo es una tienda de bicicletas que tiene ya estructurado las tablas que deben ser usadas. A continuación, se escribe sobre los atributos de las entidades/tablas:

1. La tabla sales.stores incluye la información de la tienda. Cada tienda tiene un nombre de tienda, información de contacto como teléfono y correo electrónico, y una dirección que incluye calle, ciudad, estado y código postal.
2. La tabla sales.staffs
    a. Almacena la información esencial del personal, incluido el nombre y el apellido. También contiene la información de comunicación, como correo electrónico y teléfono.
    b. Un empleado trabaja en una tienda especificada por el valor en la columna store_id. Una tienda puede tener uno o más empleados.
    c. Un empleado informa a un gerente de tienda especificado por el valor en la columna manager_id. Si el valor en manager_id es nulo, entonces el personal es el gerente superior.
    d. Si un personal ya no trabaja para ninguna tienda, el valor de la columna activa se establece en cero.
3. La tabla production.categories
    a. La tabla production.categories almacena las categorías de bicicletas, como bicicletas para niños, bicicletas de confort y bicicletas eléctricas.
4. La tabla production.brands
    a. La tabla production.brands almacena la información de la marca de las bicicletas, por ejemplo, Electra, Haro y Heller.
5. La tabla production.products
    a. La tabla production.products almacena la información del producto, como el nombre, la marca, la categoría, el año del modelo y el precio de lista.
    b. Cada producto pertenece a una marca especificada por la columna brand_id. Por lo tanto, una marca puede tener cero o muchos productos.
    c. Cada producto también pertenece a una categoría especificada por la columna category_id. Además, cada categoría puede tener cero o muchos productos.
6. La tabla sales.customers
    a. La tabla sales.customers almacena la información del cliente, incluido el nombre, apellido, teléfono, correo electrónico, calle, ciudad, estado y código postal.
7. La tabla sales.orders
    a. La tabla sales.orders almLa tabla sales.customers almacena la información del cliente, incluido el nombre, apellido, teléfono, correo electrónico, calle, ciudad, estado y código postal.acena la información del encabezado del pedido de ventas, incluido el cliente, el estado del pedido, la fecha del pedido, la fecha requerida y la fecha de envío.
    b. También almacena la información sobre dónde se creó la transacción de venta (tienda) y quién la creó (personal).
    c. Cada pedido de venta tiene una fila en la tabla sales_orders. Un pedido de ventas tiene uno o varios artículos de línea almacenados en la tabla sales.order_items.

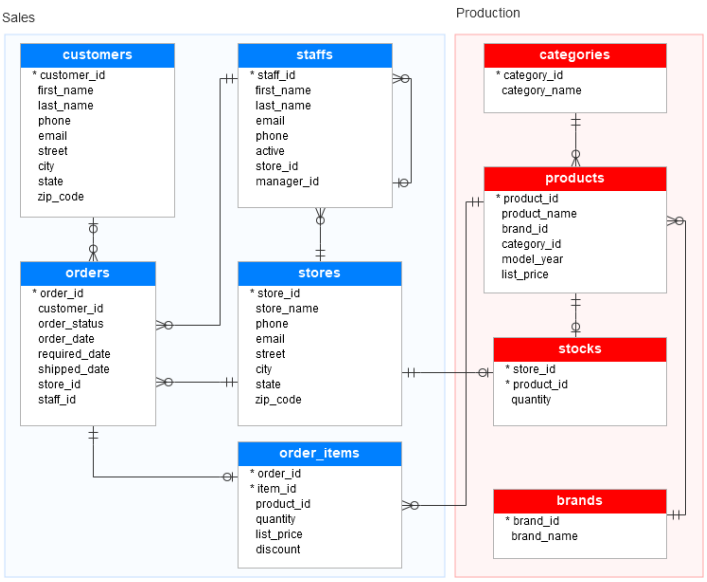
8. La tabla sales.order_items
    a. La tabla sales.order_items almacena los artículos de línea de un pedido de ventas. Cada artículo de línea pertenece a un pedido de ventas especificado por la columna order_id.
    b. Un artículo de línea de pedido de ventas incluye producto, cantidad de pedido, precio de lista y descuento.
9. La tabla production.stocks
    a. La tabla de existencias de producción almacena la información de inventario, es decir, la cantidad de un producto en particular en una tienda específica.

Link de la documentación: Pagina web base: https://www.sqlservertutorial.net/sql-server-sample-database/

Desafío II: Construyendo el modelo Entidad-Relación



Listado de Tablas:



Listado de columnas por tablas:

Tabla I: Brands

brand_id	brand_name
String	String

Tabla II: Products

product_id	product_name	brand_id	category_id	model_year	list_price
String	String	String	String	Integer/Whole Number	Currency / Decimal

Tabla III: Category

category_id	category_name
String	String

Tabla IV: Stock

store_id	product_id	quantity
String	String	Integer/Whole Number

Tabla V: Order_Items

order_id	item_id	product_id	quantity	list_price	discount
String	String	String	Integer/Whole Number	Currency / Decimal	Percentage

Tabla VI: Stores

store_id	store_name	phone	email	street	city	state	zip_code
String	String	String	String	String	String	String	String

Tabla VII: Orders

order_id	customer_id	order_status	order_date	required_date	shipped_date	store_id	staff_id
String	String	String	Date	Date	Date	String	String

Tabla VIII: Staffs

staff_id	first_name	last_name	email	phone	active	store_id	manager_id
String	String	String	String	String	String	String	String

Tabla IX: Customers

customer_id	first_name	last_name	phone	email	street	city	state	zip_code
String	String	String	String	String	String	String	String	String

Modelo relacional en Power BI

Se extraen los datos de SQL y se copian a excel, posteriormente se cargan a Power BI y se transforman los datos para que coincidan con el tipo definido en la entrega anterior.

- Los ID se definen como texto
- Los ZIPCODE se definen como texto
- Los precios se definen como moneda
- Los estados del staff se cambia a texto

Se cargan los datos y el modelo relacional queda automáticamente creado correctamente.

Generación de campos y visualizaciones

a. Se crean las columnas:

En la tabla Order_Items:

- `Subtotal = Order_Items[quantity]*Order_Items[list_price]`
- `Total = Order_Items[Subtotal]*(1-Order_Items[discount])`

En la tabla Orders:

- Order_Time = Orders[shipped_date]-Orders[order_date]
- = Table.AddColumn(#"Changed Type", "StatusName", each if [order_status] = "1" then "Pending" else if [order_status] = "2" then "Processing" else if [order_status] = "3" then "Rejected" else if [order_status] = "4" then "Completed" else null)

En la tabla Clients:

= Table.AddColumn(#"Changed Type", "Client full Name", each Text.Combine({[customer_id], "- ", [first_name], " ", [last_name]}), type text)

b. Se crean las medidas:

En la tabla Order_Items

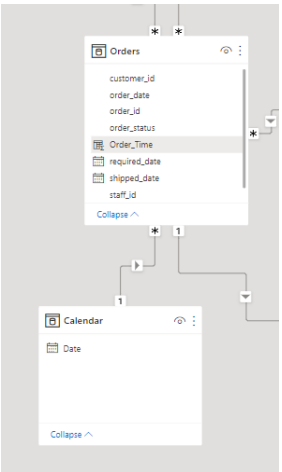
- Discount per product = SUM('Order_Items'[list_price]) * SUM('Order_Items'[discount])

- Total Discount = SUM('Order_Items'[Total]) - SUM('Order_Items'[Subtotal])

- Max Sell per product = MAXX(KEEPFILTERS(VALUES('Products'[product_name])), CALCULATE(SUM('Order_Items'[Total])))

- Total running total in order_id =
```
CALCULATE(
    SUM('Order_Items'[Total]),
    FILTER(
       ALLSELECTED('Order_Items'[order_id]),
        ISONORAFTER('Order_Items'[order_id], MAX('Order_Items'[order_id]), DESC)
    )
)
```

c. **Generar tabla calendario:** Calendar = calendarauto()
d. **Implementar 2 KPIs**
e.



CREACIÓN DE MEDIDAS AVANZADAS CALCULADAS

- Medida calculada con función de agregación
  - **MEDIDA**: Total_Sales / En la tabla Order_Items
  - **Finalidad de la medida:** Hacer la suma de todas las ventas
  - **Código**:

```
Total_Sales =
    VAR Total = CALCULATE(
        SUM(Order_Items[Total])
        )
    RETURN Total
```

- Medida con dos variables, siendo una de agregación y otra de inteligencia de tiempo
  - **MEDIDA**: %Year Change / En la tabla Order_Items
  - **Finalidad de la medida:** Encontrar el cambio porcentual entre el año actual y el anterior
  - **Código**:

```
%Year Change =
    VAR YearPrev = DATEADD('Calendar'[Date].[Date], -1, YEAR)
    VAR  PREV_YEAR =
CALCULATE(SUM('Order_Items'[Total]),YearPrev)

    RETURN
        DIVIDE(SUM('Order_Items'[Total]) - PREV_YEAR,
PREV_YEAR)
```

- Medida calculada con parámetro.

- Se crea un parámetro de ObjetivoVentas que equivale al total presupuestado de USD vendidos.

```
ObjetivoVentas = GENERATESERIES(CURRENCY(5000000), CURRENCY(10000000), CURRENCY(100000))
```

- **MEDIDA**: %ExecutionUSD / En la tabla Order_Items
- **Finalidad de la medida:** Encontrar el cumplimiento a las ventas de USD de manera porcentual
- **Código**:

```
%ExecutionUSD = SUM( Order_Items[Total] )/ObjetivoVentas[ObjetivoVentas Value]
```