**3.- Genera una estructura para cargar los datos transformados desde PostgreSQL como parte del proceso ETL.**

Para desarrollar un proceso ETL (Extracción, Transformación y Carga) que maneje la carga de datos desde PostgreSQL a otra base de datos, probablemente MongoDB o una solución de data warehouse, primero necesitas establecer una estructura que facilite esta operación, especialmente considerando la tabla normalizada que has proporcionado.

**1. Extracción de Datos (Extract)**

El primer paso de cualquier proceso ETL es extraer los datos de la fuente.

En este caso, puedes utilizar una consulta SQL para extraer datos de las tablas normalizadas:

sql

Copy code

-- Consultar los datos de cada tabla

SELECT \* FROM customers;

SELECT \* FROM products;

SELECT \* FROM orders;

**2. Transformación de Datos (Transform)**

Durante la fase de transformación, es posible que desees aplanar la estructura para hacerla más adecuada para cargas en un formato NoSQL o simplemente para mejorar la analítica. Aquí tienes un ejemplo sobre cómo puedes transformar los datos:

**2.1. Mapear Clientes a Pedidos**

Por ejemplo, puede ser útil combinar datos de clientes y pedidos para una major comprensión:

sql

Copy code

SELECT

o.order\_id,

c.customer\_id,

c.customer\_name,

p.product\_id,

p.product\_name,

p.product\_price,

o.order\_date

FROM

orders o

JOIN

customers c ON o.customer\_id = c.customer\_id

JOIN

products p ON o.product\_id = p.product\_id;

**3. Carga de Datos (Load)**

Para cargar los datos transformados en la nueva plataforma de destino (puede ser MongoDB, un archivo CSV, etc.), definimos cómo estructurarlos. A continuación, se muestra un ejemplo de cómo podría ser la estructura final en función de la información combinada:

**3.1. Estructura de Carga JSON para MongoDB**

Aquí tienes un ejemplo de cómo se vería el JSON que podrías cargar en una colección de MongoDB:

json

Copy code

[

{

"order\_id": 1,

"customer": {

"customer\_id": 1,

"customer\_name": "John Doe"

},

"product": {

"product\_id": 1,

"product\_name": "Laptop",

"product\_price": 1200.50

},

"order\_date": "2024-01-01T10:00:00Z"

},

{

"order\_id": 2,

"customer": {

"customer\_id": 2,

"customer\_name": "Jane Smith"

},

"product": {

"product\_id": 2,

"product\_name": "Smartphone",

"product\_price": 800.75

},

"order\_date": "2024-01-02T11:30:00Z"

},

{

"order\_id": 3,

"customer": {

"customer\_id": 3,

"customer\_name": "Alice Johnson"

},

"product": {

"product\_id": 3,

"product\_name": "Tablet",

"product\_price": 300.00

},

"order\_date": "2024-01-03T15:45:00Z"

},

{

"order\_id": 4,

"customer": {

"customer\_id": 4,

"customer\_name": "Bob Brown"

},

"product": {

"product\_id": 4,

"product\_name": "Monitor",

"product\_price": 150.99

},

"order\_date": "2024-01-04T09:20:00Z"

},

{

"order\_id": 5,

"customer": {

"customer\_id": 5,

"customer\_name": "Carol White"

},

"product": {

"product\_id": 5,

"product\_name": "Keyboard",

"product\_price": 50.00

},

"order\_date": "2024-01-05T14:10:00Z"

}

]

**4. Implementación del Proceso ETL**

Dependiendo de la tecnología y el lenguaje que quieras usar, puedes implementar el proceso ETL de diversas maneras. Aquí te presento una forma general de hacerlo:

**4.1. Herramienta ETL**

Puedes usar herramientas como Apache NiFi, Talend, o crear un script personalizado en Python o Node.js. Aquí hay un esbozo de lo que podría ser un script ETL en Python con pandas para manejar la carga en MongoDB:

python

Copy code

import pandas as pd

from sqlalchemy import create\_engine

from pymongo import MongoClient

# Configuración de la base de datos PostgreSQL

pg\_engine = create\_engine('postgresql://user:password@localhost:5432․․․