

Base de datos Multidimensionales

Joel Molina^{L00385791}

Universidad de las Fuerzas Armadas
Jamolina13@espe.edu.ec

Abstract

En el presente documento se tiene como finalidad presentar un documento que permita al lector conocer ciertos conceptos básicos sobre las bases de datos multidimensionales, en donde se crearán las tablas de dimensiones y una tabla de hechos que permitirán crear una base de datos multidimensional en sus dos esquemas más utilizados, los cuales son: el esquema estrella y el esquema copo de nieve. En este documento también se manejarán diferentes gestores de bases de datos, tales como SQLite y SQL Server Managment Studio. Adicional a esto también se realizan diferentes procesos que nos permiten enviar datos desde una base de datos relacional a una base de datos multidimensional. Todo lo mencionado anteriormente está debidamente documentado y explicado en el presente documento.

1 Introducción

Existen diferentes maneras de almacenar datos, desde datos almacenados en archivos planos hasta datos almacenados en bases de datos no estructuradas. Dentro de todas estas maneras de almacenar datos también existen subdivisiones, entre las bases de datos SQL existen tipos como la relacional que es muy famosa, ya que busca mantener una buena integridad y poca redundancia en su base de datos. Sin embargo, también existen otros tipos de datos SQL como las bases de datos multidimensionales que buscan mejorar la manera en la que se obtienen las consultas y permiten cierto grado de redundancia. En este laboratorio se iniciará desde una base de datos relacional y mediante procesos ETL se buscará terminar con un resultado de una base de datos Multidimensional.

Como es de conocimiento general, una base de datos puede almacenar datos de una o diversas fuentes de datos. Estos datos no siempre van a tener la misma estructura o tipo de dato que se necesita para la base de datos final. Es por esto que en este laboratorio se realizan los procesos ETL. ETL son las siglas que hacen referencia a los procesos de Extracción, Transformación y Carga de datos en un sistema. Según menciona [1], "ETL tiene una gran importancia al momento de la implementación de una data warehouse, puede significar el éxito o fracaso de dicha implementación, estos procesos pueden significar cerca de un 70 por ciento de los recursos destinados al proyecto". Estos procesos tienen gran importancia dentro de una empresa, debido a que buscan una mejora en la integridad de sus datos, esto se logra mediante un proceso en el cual se eliminan datos que no tienen el mismo formato que la empresa lo requiere.

Contexto y motivación Hoy en día para un administrador de base de datos es esencial saber que es el modelo de datos multidimensionales y como funciona, ya que dicho modelo proporciona

varias técnicas para poder modelar bases de datos. En otras palabras, para realizar un excelente modelado de base de datos y que el usuario visualice eficientemente la información que existe en el modelo de base de datos. Además, de aplicar procesos que ayuden al manejo de cada tipo de estos datos y conservar su integridad, por ello surge como una solución el proceso ETL. Para así dar un buen mantenimiento a la empresa y que dicha empresa se mantenga a flote en el mercado.

Problema de estudio Hoy en día las empresas deben tomar decisiones muy importantes sobre los procesos de su negocio, por lo que dicha empresa debe contar con herramientas de soporte eficientes para la toma de decisiones, por esta razón es esencial tener un buen modelado de base de datos de la empresa. Por ese motivo surge como una solución el modelado multidimensional.

Trabajo relacionado Según menciona [2], realizó un análisis y desarrollo de una solución OLAP para la toma de decisiones de la gerencia en el proceso de compras y ventas de la empresa DISMERO S.A, de la provincia de los Ríos. Por lo que, el objetivo principal es crear un modelamiento multimimensional (OLAP) para tratar la información de manera más eficiente, rápida y concreta en la toma de decisiones, y es por ello que los directivos han optado por el desarrollo e implementación de una solución OLAP para agilizar el manejo ligado a estos procesos de toma de decisiones.

Según menciona [3], en su investigación, al momento de realizar los procesos ETL se deben de tomar en cuenta algunas consideraciones, entre ellas destacan: conocer el objetivo y necesidades de la empresa, tener claro que herramienta se utilizará para estos procesos y realizar algunas pruebas de errores con el fin de corregir errores en el caso de que existieran.

Resumen de contribución Por medio de este proyecto se desarrolla el modelado multidimensional de la base de datos 'Northwind_large', utilizando el Notebook Jupyter, el cual trabaja sobre el lenguaje de programación Python.

2 Antecedentes

Lenguaje de programación Python

En cuanto al lenguaje de programación escogido, es porque Python es un lenguaje de programación el cual tiene una sintaxis y tipificación fácil, que inclusive una persona con conocimientos básicos de programación podría comprender. Este lenguaje, además, está orientado a objetos, por lo que tenemos la posibilidad de hacer uso de los pilares de la programación dirigida a objetos para el desarrollo de programa. Por otro lado, este lenguaje igual soportar multiparadigma, por lo cual, se puede desarrollar orientado a objetos, de manera servible y además organizada. Por otro lado, el motivo primordial por la que usamos Python ha sido pues Python cuenta con muchas bibliotecas para el desarrollo del proceso ETL, por lo cual poseemos más posibilidades para añadir varias funcionalidades [4].

SQLite SQLite es un instrumento de programa independiente, que posibilita guardar información en dispositivos de una manera simple y eficaz. Cabe recalcar que SQLite trabaja en equipos escasas habilidades de hardware, como podría ser una PDA o un teléfono celular. Esto posibilita que SQLite soporte a partir de las consultas más primordiales hasta las más complicadas del lenguaje SQL, y lo de mayor relevancia es que se puede utilizar tanto en dispositivos móviles como en sistemas de escritorio, sin necesidad de hacer procesos complicados de importación y exportación de datos, ya existente compatibilidad total [4]. Por este motivo hacemos uso de SQLite, ya que no consume demasiados recursos para los equipos como otros softwares.

SQL Server Management Studio SQL Server Management Studio es un ámbito incluido para regir cualquier infraestructura de SQL. Además, nos permite entrar, configurar, gestionar, regir y desarrollar todos los elementos de SQL Server. Cabe recalcar, que SQL Server Management Studio da una sola utilidad integral que combina un extenso conjunto de herramientas gráficas con varios editores de secuencias de comandos enriquecidos para brindar ingreso a SQL Server a desarrolladores y administradores de bases de datos de todos los niveles [5].

Notebook Jupyter

Jupyter Notebook es una aplicación web de código abierto que nos posibilita generar y compartir código y documentos. En donde además, nos proporciona un entorno amigable en donde podemos desarrollar Notebooks de Python, siendo así fácil su implementación [6].

Transact SQL Avanzado Para la comprensión de Transact SQL toca tener claro que es SQL, el cual es un lenguaje de programación que nos ayuda a solucionar problemas relacionados con la manipulación e integridad de la información que se almacena en una base de datos [7]. Mientras que, el lenguaje de programación Transact SQL es proporcionado por Microsoft SQL Server, el cual nos ayuda a extender el SQL normal con otro tipo de sintaxis [7].

3 Método

Paso 1: Descarga de Python en el sitio web oficial. Primero que nada, debemos tener en claro el lenguaje de programación por cuál vamos a trabajar, en nuestro caso como grupo vamos a desarrollar este proyecto por medio del lenguaje de programación Python. Así que, ingresamos al navegador web para posteriormente redireccionarnos al sitio web de Python y descargar para el sistema operativo Windows tal y como se puede observar en la Figura 1.

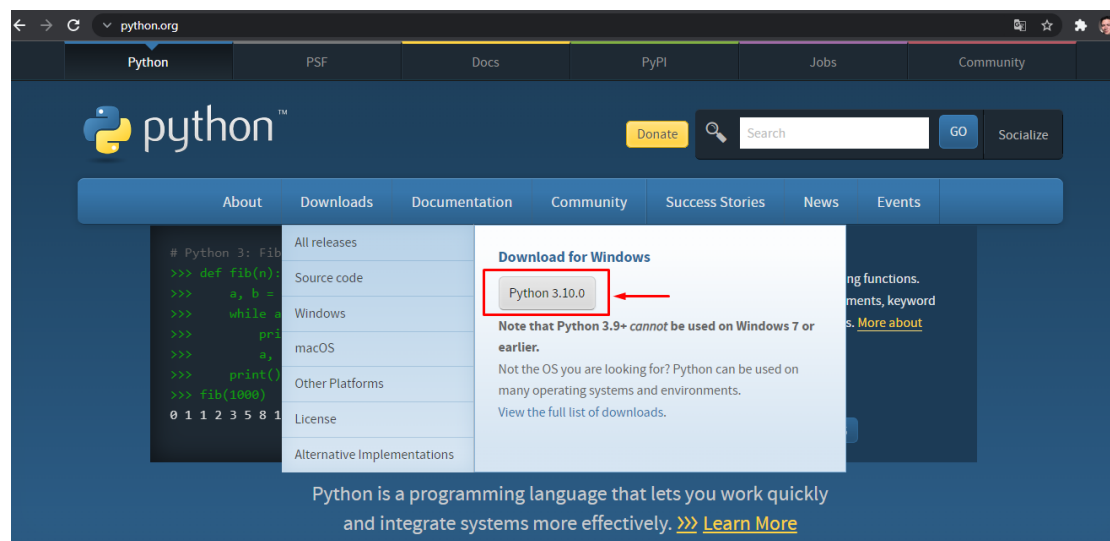


Figure 1: Descarga de Python python.

Paso 2: Ejecución del programa Python.

Una vez descargado el ejecutable con extensión “.exe” procederemos a ejecutarlo como administrador. Tal y como lo muestra la figura 2

Paso 3: Instalación de Python 3.10.0.

Nombre	Fecha de modificación	Tipo	Tamaño
mongodb-windows-x86_64-5.0.4-signed....	23/11/2021 9:43	Paquete de Windo...	284.208 KB
python-3.10.0-amd64.exe	28/11/2021 19:10	Aplicación	27.653 KB

Figure 2: Ejecución del programa Python.

Una vez ejecutado el programa Python procederemos a instalarlo marcando la opción que se muestra en la Figura 3.

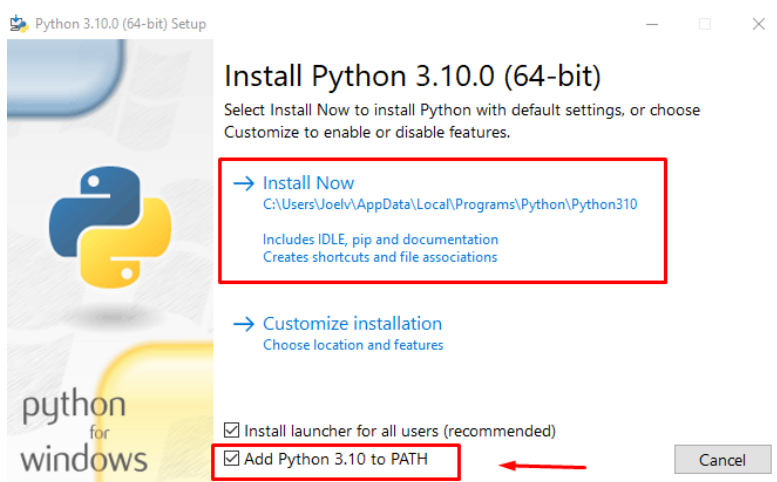


Figure 3: Instalación de Python.

Paso 4: Progreso de la configuración finalizado.

Finalmente, se llegó al apartado en donde nos menciona que la instalación fue exitosa. Tal y como lo muestra la figura 4



Figure 4: Instalación de Python finalizado.

Paso 5: Instalación de Anaconda Navigator

Debemos tener instalado el ambiente de trabajo 'Anaconda Navigator' que nos permite abarcar una serie de aplicaciones y librerías para el desarrollo de ciencia de datos con Python **cita9**. En caso de no tener instalado "Anaconda Navigator", debemos ingresar a la siguiente URL: <https://www.anaconda.com/products/individual/> e instalarlo. Una vez instalado "Anaconda Navigator", lo iniciamos teniendo así la siguiente interfaz que se detalla en la Figura 5.

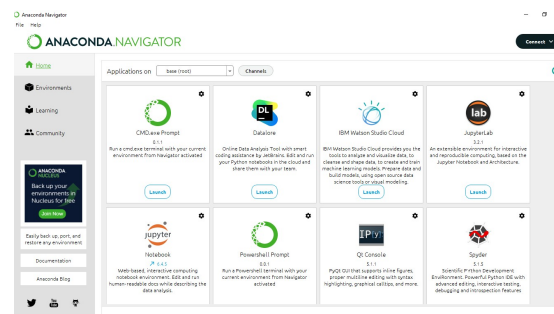


Figure 5: Interfaz de Anaconda Navigator.

Paso 6: Inicialización de Jupyter

Seguidamente, iniciamos 'Anaconda Navigator' y buscamos la aplicación 'Jupyter Notebook' y la iniciamos. Dicha aplicación nos permite crear y compartir documentos web en formato JSON. En la Figura 6 se detalla la interfaz de "Jupyter Notebook".

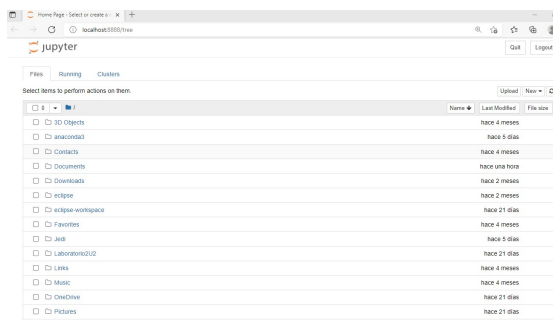


Figure 6: Interfaz de Jupyter Notebook.

Paso 7: Creación del Notebook en Jupyter

Después de haber inicializado 'Jupyter Notebook', creamos un nuevo notebook con Python, el cual nos servirá para realizar el proceso ETL para archivos planos. En la Figura 7 se detalla el notebook creado.

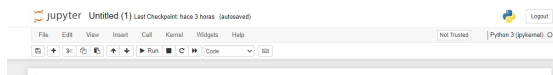


Figure 7: Notebook.

Paso 8: Descarga de SQLite

Descargamos SQLite desde la siguiente URL: <https://www.sqlite.org/index.html>

Seguidamente, inicializamos SQLite y procedemos a abrir la base de datos 'Northwind_large'. En la siguiente Figura 8. se detalla la realización de este paso.

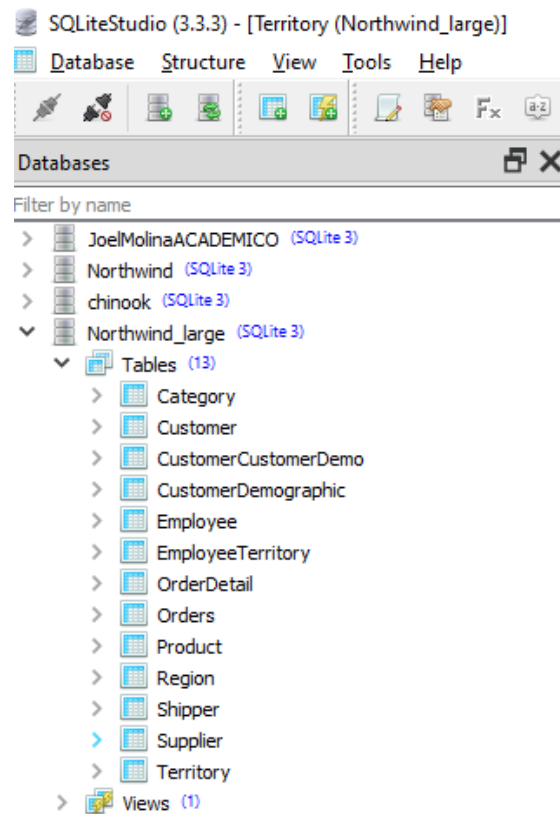


Figure 8: Interfaz SQLite.

3.1 Diseño

Para el desarrollo del diseño, primero se planteó cuáles eran los factores determinantes a tener en cuenta para el modelamiento multidimensional de la base de datos 'Northwind_large', en el esquema estrella y el esquema copo de nieve. Dando como resultado el esquema estrella que se detalla en la Figura 9. Mientras que en la 10 se detalla el esquema copo de nieve.

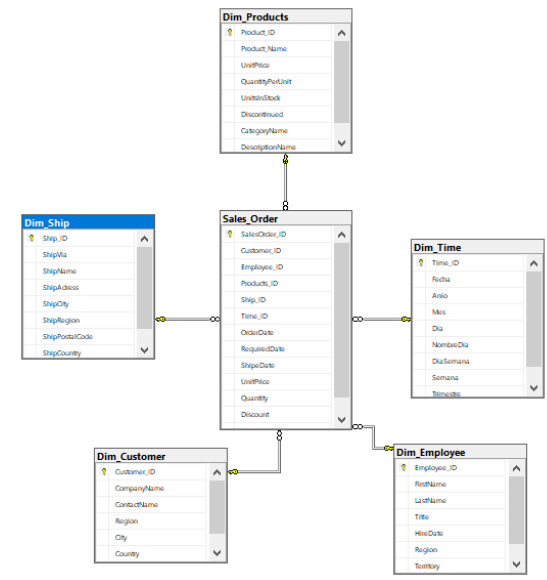


Figure 9: Esquema estrella de la base de datos 'Northwind_large'.

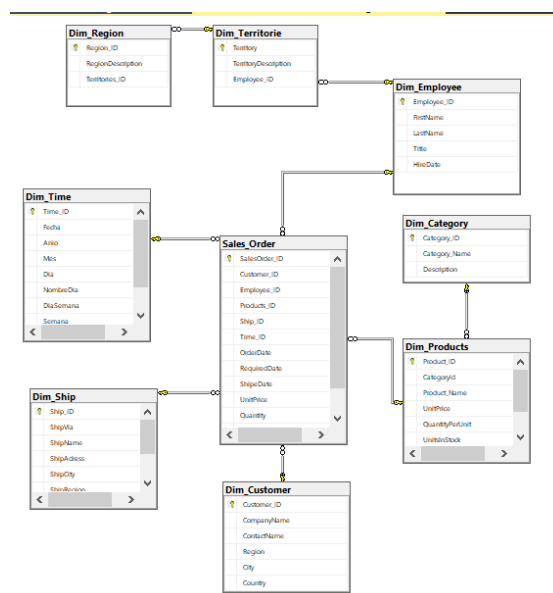


Figure 10: Esquema copo de nieve de la base de datos 'Northwind_large'.

3.2 Análisis

En este proceso de modelamiento multidimensional y los procesos ETL se realizaron una serie de preguntas antes de realizarlas, estas preguntas tenían como objetivo principal el generar

respuestas a las principales dudas que suele tener un gerente de una empresa, de tal manera que puedan realizar un buen análisis con los datos que tiene de resultado de estos procesos realizados.

3.3 Optimización

En el diseño del modelado multidimensional se utilizaron dos de los esquemas más utilizados al momento de realizar el modelado multidimensional, los cuales son: el modelo estrella y el modelo copo de nieve. Dando como resultado el modelamiento correcto de la base de datos 'Northwind_large'.

Los procesos ETL se optimizaron mediante una buena estructuración de código, de esta manera se busca evitar la redundancia y una buena lectura de código al momento de realizar los procesos ETL.

3.4 Implementación

Para la elaboración del modelo multidimensional estrella primero se analizó profundamente los campos que se necesitaban si se requería que el foco de la base de datos sea la venta. Es por esto que solo se realizaron 5 dimensiones, Ship, Products, Time, Employee y Customer. En cuanto a la tabla de hechos, se procedió a agregar todas las llaves primarias de las dimensiones y ciertos campos que miden el negocio, como por ejemplo unitPrice, Quantity, entre otros. Todo lo mencionado anteriormente se lo puede visualizar en la figura 11

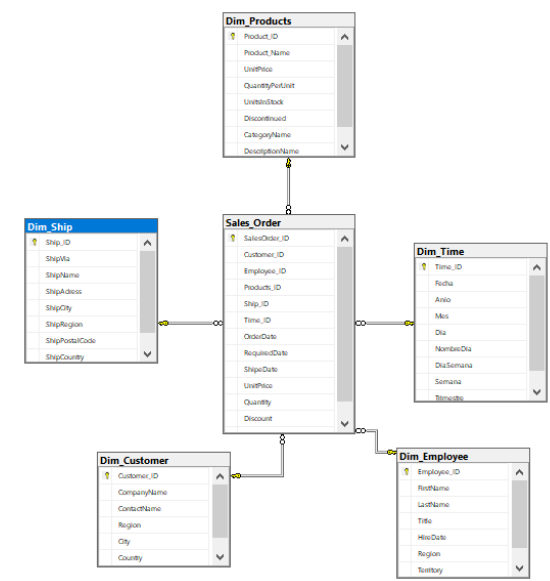


Figure 11: Diagrama Estrella.

Al igual que en el modelo estrella, en el modelo de copo de nieve se analizó y se determinó de que a partir del modelo estrella se normalizarían dos tablas. Las tablas que se normalizaron en este modelo fueron las de Employee y Products. Todo lo mencionado anteriormente se puede visualizar en la figura 12.

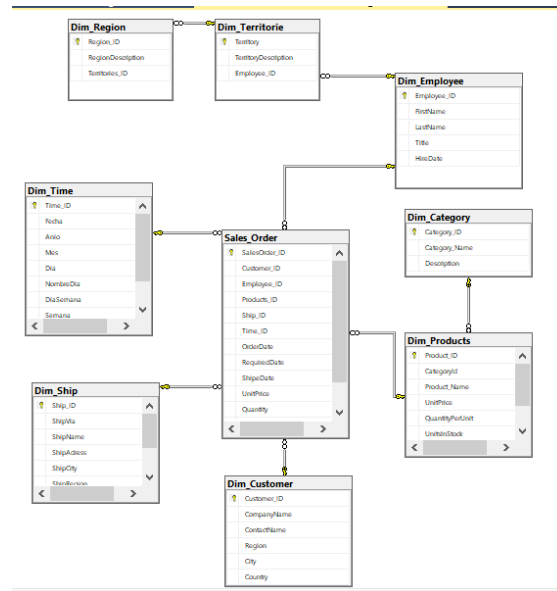


Figure 12: Diagrama Estrella.

Como se mencionó anteriormente, para esta práctica se necesita del lenguaje de programación Python, adicional a esto se debe de tener instalado Jupyter en la computadora o hacerlo desde internet con la aplicación de Jupyter online. Para la realización de esta práctica se necesitan algunas librerías y módulos del lenguaje de programación Python, entre las cuales cabe destacar a pandas, una librería muy útil al momento de analizar y estructurar datos. Otra librería de vital importancia es sqlalchemy, ya que con esta librería se podrá conectar con la base de datos multidimensional. Las librerías y módulos se pueden visualizar en la figura 13

```
] : import sqlalchemy
import pymysql
from sqlalchemy import create_engine
import pandas as pd
import pyodbc
```

Figure 13: Librerías.

Cabe recalcar que el proceso que se documentará será el de ETL de una base de datos relacional a una base de datos multidimensional de modelo Estrella, el modelo copo de nieve no está documentado debido a que son muy parecidos los pasos a seguir. Como primer paso para los procesos ETL se procede a crear una conexión con la base de datos northwind para posteriormente poder manipularla mediante código de python. La conexión con la base de datos northwind se puede observar en la figura 14. Para la conexión se utiliza la librería sqlalchemy mediante el método createengine(), en este método se le especifica que la base de datos es de sqlite y se le especifica la ruta en la que se encuentra.

```
northwind = create_engine('sqlite:///Northwind_larges.sqlite')
northwind
```

Figure 14: Conexión con base de datos northwind.

Por objetivos de este proyecto se necesita crear una nueva tabla en la base de datos relacional northwind, esta tabla es una llamada tiempo que maneja fechas desde el 2009 hasta el 2030. Para poder crear estos datos se hace uso de la librería pandas mediante el método date_range. Este método permite crear un rango de fechas y se le pasa como parámetros una fecha de inicio y una fecha de salida, tal y como lo muestra la figura 15. Luego de esto se le asignan campos al dataframe, entre los cuales deberá mostrar la fecha, año, mes, día, nombre del día, día de semana y trimestre a los que pertenecen. Todo este proceso se lo guardó dentro de un método y retorna un dataframe.

```
def create_date_table(start='2009-01-01', end='2030-12-31'):
    df = pd.DataFrame({'date': pd.date_range(start, end)})
    df['date_id'] = df.index + 1
    df['año'] = df.date.dt.year
    df['mes'] = df.date.dt.month
    df['dia'] = df.date.dt.day
    df['nombre_dia'] = df.date.dt.day_name()
    df['dia_semana'] = df.date.dt.dayofweek
    df['semana'] = df.date.dt.weekofyear
    df['trimestre'] = df.date.dt.quarter

    df = df[['date_id', 'date', 'año', 'mes', 'dia', 'nombre_dia', 'dia_semana', 'semana', 'trimestre']]

    return df
```

Figure 15: Dataframe de Tiempo.

Adicional al dataframe, también se necesita crear una tabla en la base de datos northwind que recibirá estos datos. En la figura 16 se puede visualizar el código que se realiza para crear la tabla. En este código se puede ver que se agrega una clave primaria que es autoincrementable y los campos que tiene el dataframe.

```
fecha_inicio='2009-01-01'
fecha_fin='2030-12-31'
dataFrameDate = pd.DataFrame({'fecha': pd.date_range(fecha_inicio, fecha_fin)})
dataFrameDate['anio'] = dataFrameDate.fecha.dt.year
dataFrameDate['mes'] = dataFrameDate.fecha.dt.month
dataFrameDate['dia'] = dataFrameDate.fecha.dt.day
dataFrameDate['nombredia'] = dataFrameDate.fecha.dt.day_name()
dataFrameDate['diasemana'] = dataFrameDate.fecha.dt.dayofweek
dataFrameDate['semana'] = dataFrameDate.fecha.dt.weekofyear
dataFrameDate['trimestre'] = dataFrameDate.fecha.dt.quarter
dataFrameDate = dataFrameDate[['fecha', 'anio', 'mes', 'dia', 'nombredia', 'diasemana', 'semana', 'trimestre']]
```

Figure 16: Dataframe de Tiempo.

Luego de obtener un dataframe con las fechas requeridas y de haber creado una tabla tiempo en la base de datos northwind, se procede a subir el dataframe a la base de datos northwind. Tal y como se muestra en la figura 17 se hace uso del método `to_sql()`, el cual permite subir datos a una tabla sql siempre y cuando los datos ingresados sean correctos. Este método recibe como parámetros el nombre de la tabla a la cual se va a subir, la conexión con la base de datos, un condicional que evalúa si la base de datos está creada y toma una decisión a partir de esta evaluación y el último parámetro que es para no crear un índice.

```

1 CREATE TABLE Dim_Time (
2     TimeId      INTEGER      PRIMARY KEY AUTOINCREMENT,
3     fecha       DATE,
4     año         INTEGER,
5     mes         INTEGER,
6     día         INTEGER,
7     nombre_día  VARCHAR (50),
8     día_semana  INTEGER,
9     semana      INTEGER,
10    trimestre   INTEGER
11 );
12

```

Figure 17: Carga del dataframe de tiempo.

Una vez que se realizaron los pasos anteriores, se procede a extraer datos que van a ser llenados en las tablas de dimensiones de la base de datos multidimensional. La manera en la que se extraen estos datos es mediante la librería pandas y su método `read_sql_query()`, a este método se le pasan dos parámetros. El primer parámetro que se le pasa es la consulta sql y el segundo parámetro es la conexión con la base de datos. Como se puede visualizar en la figura 18 Cada dataframe obtiene los campos que necesita y algo que cabe recalcar es que no se extraen llaves primarias porque en la base de datos multidimensional estas llaves fueron configuradas como autoincrementables, por tal motivo ese campo no se llena manualmente, el gestor de base de datos lo hace automático.

```

frameShips = pd.read_sql_query("""select ShipVia, ShipName, ShipAddress as ShipAddress, ShipCity, ShipRegion,
ShipPostalCode, ShipCountry from Orders;""",
con=northwind.connect())

frameCustomer = pd.read_sql_query("""select Id as Customer_ID, CompanyName, ContactName, Region, City, Country from Customer;""",
con=northwind.connect())

frameProduct = pd.read_sql_query("""select ProductName as Product_Name, UnitPrice, QuantityPerUnit,
UnitsInStock as UnitsInStock, Discontinued, Category.CategoryName, Category.Description as DescriptionName
from Product inner join Category
on Product.CategoryId = Category.Id;""",
con=northwind.connect())

```

Figure 18: Extracción de datos.

Para la extracción de datos que servirán para la tabla de hechos se realiza un proceso un poco diferente en comparación a la extracción para las dimensiones. Como se puede observar en la figura 19, se crea un dataframe que contiene diferentes consultas con inner joins, de esta manera se busca obtener los datos correspondientes. Al final de la consulta se puede observar que en la dimensión tiempo no se compara mediante los ids, debido a que como es una tabla nueva no tiene relaciones con otras tablas. La comparación se la hace por la fecha de la tabla orders, de esta manera se obtienen los datos correctos. Finalmente, se muestra el dataframe obtenido con los campos que se necesitan.

```

frameOrderSales = pd.read_sql_query("""SELECT Customer.Id as Customer_ID, EmployeeId as Employee_ID, ProductId as Products_ID,
dim_ships.Id as Ship_ID, Dim_Time.TimeId as Time_ID, Orders.OrderDate, Orders.RequiredDate, Orders.ShippedDate as ShipDate,
OrderDetail.UnitPrice, OrderDetail.Quantity,
OrderDetail.Discount, (OrderDetail.UnitPrice * OrderDetail.Quantity) - OrderDetail.Discount as Total
FROM Product INNER JOIN OrderDetail
ON Product.Id = OrderDetail.ProductId INNER JOIN Orders
on OrderDetail.OrderId = Orders.Id INNER JOIN Employee
on Orders.EmployeeId = Employee.Id INNER JOIN Customer
on Orders.CustomerId = Customer.Id INNER JOIN dim_ships
on Orders.Id = dim_ships.Id INNER JOIN Dim_Time
ON Orders.OrderDate = substr(dim_time.fecha,0,11)""",
con=northwind.connect())

```

Figure 19: Extracción de datos.

Una vez que se realizó el proceso de extracción se procede con el proceso de transformación. En esta ocasión se necesita transformar los datos nulos de las tablas en datos reales. Para realizar esto se hizo uso del método `fillna()`, este método puede manipularse de algunas maneras, pero en todas se busca reemplazar los datos nulos por algún valor. En este caso se le pide al método que reemplace todos los valores nulos por el valor no nulo de abajo, tal y como lo muestra la figura 20. En este proceso se realizan estos cambios para todos los datos nulos de todos los dataframes.

```

frameShips.fillna(method="bfill", inplace = True)
frameCustomer.fillna(method="bfill", inplace = True)
frameProduct.fillna(method="bfill", inplace = True)
Employee.fillna(method="bfill", inplace = True)
frameOrderSales.fillna(method="bfill", inplace = True)
dataFrameDate.fillna(method="bfill", inplace = True)

```

Figure 20: Limpieza de datos.

Para finalizar el proceso ETL, se procede a realizar el proceso de carga de datos. Para esto se necesita conectar con la base de datos multidimensional que está en SQL server. Tal y como lo muestra la figura 21, se crea una conexión mediante el método `createEngine()` y se lo guarda en una variable. Una vez que se realizó esta conexión se procede a enviar todos los dataframes que tenemos a SQL, esto se hace mediante el método `toSql()` de pandas. A este método se le especifica el nombre de la dimensión o tabla de hechos a la que se exportará, se le pasa la conexión con la base de datos, se condiciona para que evalúe si existe o no la tabla, si no existe, se crea una tabla con el nombre especificado y si existe, añade los datos como datos nuevos, finalmente se le dice que no cree un índice.

```

engine = sqlalchemy.create_engine('mssql+pyodbc://localhost/ESQUEMAESTRELLA3?driver=SQL+Server+Native+Client+11.0')
try:
    frameShips.to_sql(name='Dim_Ship', con=engine, if_exists='append', index=False)
    frameCustomer.to_sql('Dim_Customer', con=engine, if_exists='append', index=False)
    frameProduct.to_sql('Dim_Products', con=engine, if_exists='append', index=False)
    Employee.to_sql('Dim_Employee', con=engine, if_exists='append', index=False)
    dataFrameDate.to_sql('Dim_Time', con=engine, if_exists='append', index=False)
    frameOrderSales.to_sql('Sales_Order', con=engine, if_exists='append', index=False)
except Exception as ex:
    print("Ocurrió un error: {}".format(ex))

```

Figure 21: Carga de datos a SQL Server.

4 Evaluación Experimental

Mediante Jupyter Notebook se desarrolla la implementación del proceso ETL en la base de datos 'Northwind_large'. Par así, poder crear nuestras Data Warehouse respectivas para cada modelo (estrella y copo de nieve) del modelo multidimensional de la misma base de datos.

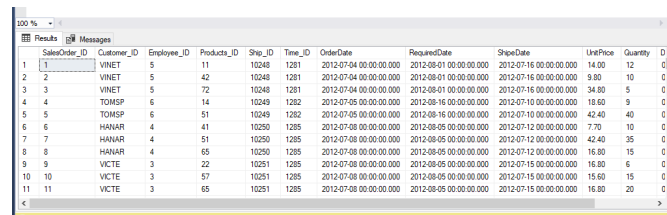
4.1 Configuración experimental

Basándose en los objetivos a cumplir, se implementan varios procesos para poder realizar el modelamiento multidimensional de la base de datos 'Northwind_large', aplicando también las formas de normalización para un modelamiento multidimensional eficaz.

Además, se aplican las fases que componen el proceso ETL, para así implementar su funcionalidad. Por lo cual realizamos varias funciones de las cuales nos permitirán extraer los datos de la base de datos 'Northwind_large', transformarlos y cargarlos a los respectivos esquemas del modelado multidimensional.

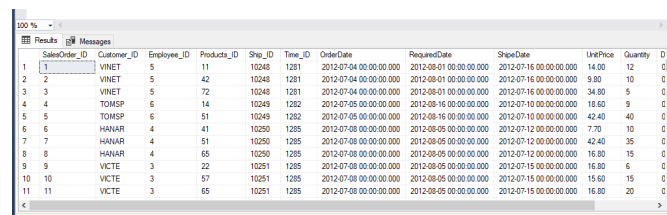
4.2 Resultados

Como resultado del proyecto, tenemos la carga de los datos exitosa a los diferentes esquemas multidimensionales (esquema estrella y copo de nieve), tal y como se detalla en la 22 los datos de la tabla de hechos del esquema estrella. Mientras que en la 23 se detallan los datos de la tabla de hechos del esquema copo de nieve. Cabe recalcar que dichas tablas de hechos son iguales, ya que lo que cambian son las dimensiones de cada esquema.



SalesOrder_ID	Customer_ID	Employee_ID	Products_ID	Ship_ID	Time_ID	OrderDate	RequiredDate	ShipDate	UnitPrice	Quantity
1	VINET	5	11	10248	1281	2012-07-04 00:00:00.000	2012-08-01 00:00:00.000	2012-07-16 00:00:00.000	14.00	12
2	VINET	5	42	10248	1281	2012-07-04 00:00:00.000	2012-08-01 00:00:00.000	2012-07-16 00:00:00.000	9.90	10
3	VINET	5	72	10248	1281	2012-07-04 00:00:00.000	2012-08-01 00:00:00.000	2012-07-16 00:00:00.000	34.80	5
4	TOMSP	6	14	10249	1282	2012-07-05 00:00:00.000	2012-08-16 00:00:00.000	2012-07-10 00:00:00.000	18.60	9
5	TOMSP	6	51	10249	1282	2012-07-05 00:00:00.000	2012-08-16 00:00:00.000	2012-07-10 00:00:00.000	42.40	40
6	HANAR	4	41	10250	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-12 00:00:00.000	7.70	10
7	HANAR	4	51	10250	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-12 00:00:00.000	42.40	35
8	HANAR	4	65	10250	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-12 00:00:00.000	16.80	15
9	VICTE	3	22	10251	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-15 00:00:00.000	16.80	6
10	VICTE	3	57	10251	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-15 00:00:00.000	15.60	15
11	VICTE	3	65	10251	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-15 00:00:00.000	16.80	20

Figure 22: Tabla de hechos Sales_Order del esquema estrella.



SalesOrder_ID	Customer_ID	Employee_ID	Products_ID	Ship_ID	Time_ID	OrderDate	RequiredDate	ShipDate	UnitPrice	Quantity
1	VINET	5	11	10248	1281	2012-07-04 00:00:00.000	2012-08-01 00:00:00.000	2012-07-16 00:00:00.000	14.00	12
2	VINET	5	42	10248	1281	2012-07-04 00:00:00.000	2012-08-01 00:00:00.000	2012-07-16 00:00:00.000	9.90	10
3	VINET	5	72	10248	1281	2012-07-04 00:00:00.000	2012-08-01 00:00:00.000	2012-07-16 00:00:00.000	34.80	5
4	TOMSP	6	14	10249	1282	2012-07-05 00:00:00.000	2012-08-16 00:00:00.000	2012-07-10 00:00:00.000	18.60	9
5	TOMSP	6	51	10249	1282	2012-07-05 00:00:00.000	2012-08-16 00:00:00.000	2012-07-10 00:00:00.000	42.40	40
6	HANAR	4	41	10250	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-12 00:00:00.000	7.70	10
7	HANAR	4	51	10250	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-12 00:00:00.000	42.40	35
8	HANAR	4	65	10250	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-12 00:00:00.000	16.80	15
9	VICTE	3	22	10251	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-15 00:00:00.000	16.80	6
10	VICTE	3	57	10251	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-15 00:00:00.000	15.60	15
11	VICTE	3	65	10251	1285	2012-07-08 00:00:00.000	2012-08-05 00:00:00.000	2012-07-15 00:00:00.000	16.80	20

Figure 23: Tabla de hechos Sales_Order del esquema copo de nieve

5 Conclusión

Tal y como hemos podido comprobar, se implementaron los procesos ETL para poder realizar la extracción, transformación y carga de datos desde la base de datos relacional hacia la base

de datos multidimensional. Estos procesos pueden ser establecidos en muchos lenguajes de programación, sin embargo, en este laboratorio se implementó en Python debido a que este lenguaje es muy fácil de programar y cuenta con múltiples librerías y funciones que permiten la manipulación de datos. En el proceso de transformación de datos se utilizó la función `fillna` con el método `bfill`, de tal manera que los valores nulos sean reemplazados por el siguiente valor no nulo de la tabla, de esta manera se colocaron valores reales dentro de la base de datos.

Además, se cumplió el objetivo principal que era transformar una base de datos relacional en una base de datos multidimensional. Durante este proceso se tuvo algunos problemas en cuanto al diseño de la base de datos multidimensional y esto es normal debido a que originalmente esta base de datos estaba pensada para ser relacional. El diseñador de la base de datos debe tener en cuenta todos los campos necesarios y las tablas necesarias para la elaboración de la nueva base de datos de tal manera de que no se pierdan datos. Adicional a esto el diseñador también deberá elaborar bien la tabla de hechos, ya que esta es sumamente importante debido a que aquí se conectan las dimensiones y se establecen atributos que contienen los valores del negocio.

Algo a tener en cuenta, es que el modelo multidimensional parte del modelo de una base de datos transaccional, implementando así varias técnicas para poder realizar un correcto modelado de una base de datos, logrando un diseño amigable y de fácil comprensión para el destinatario, el cual es el usuario final, ya que la información es fácil de comprender. No obstante, por medio del modelo multidimensional realizado en el presente laboratorio se obtuvieron conocimientos eficaces al momento de crear un modelo multidimensional y su estructura.

Cabe recalcar, que los modelos multidimensionales creados en el presente proyecto fueron el esquema estrella y el esquema copo de nieve, siendo un poco más complejo el esquema de copo de nieve. Cabe recalcar, que cualquiera de los dos esquemas del modelo multidimensional cumplen con el mismo objetivo, el cual es crear un modelado de una base de datos de manera eficiente para la toma de decisiones de la empresa.

References

- [1] J. V. Chávez, “Marco de trabajo basado en ontologías para el proceso etl,” *Maestro en Ciencias de la Computación Aplicada, Departamento de Computación, Centro de Investigación y de estudios avanzados del Instituto Politecnico Nacional de México, México, DF*, 2011.
- [2] V. Pazmiño, “Las aplicaciones olap y su importancia en el soporte a la toma de decisiones gerenciales en los procesos de compras y ventas en la empresa dismero sa,” *Tesis de Licenciatura. Universidad Técnica de Ambato. Facultad de Ingeniería en Sistemas, Electrónica e Industrial. Maestría en Gestión de Bases de Datos.*, 2012.
- [3] T. Martinez Trujillo, “Gestión de datos empresariales utilizando procesos etl,”
- [4] J. Chamorro Rodriguez, “Aplicación web para la elaboración y gestión de procesos etl en big data,” B.S. thesis, 2016.
- [5] microsoft. “What is sql server management studio (ssms).” visitado el 2022-02-28. (), [Online]. Available: <https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver15>.
- [6] Santiago. “¿conoces jupyter notebook?” visitado el 2021-12-27. (), [Online]. Available: <https://www.ceupe.mx/blog/conoces-jupyter-notebook.html>.
- [7] P. Ramos. “Qué es y para qué sirve sql.” visitado el 2022-01-23. (2018), [Online]. Available: <https://styde.net/que-es-y-para-que-sirve-sql/>.