

# The Power of Data: How ETL Reshaped Data Management

## El poder de los datos: Cómo ETL reformuló la gestión de datos

Leonardo J. Robles Angeles<sup>1</sup> , Marving B. Robles Angeles<sup>1</sup> ,  
Leonor Angeles Hernández<sup>2\*</sup>  y Mónica Leticia Acosta Miranda<sup>3</sup> 

<sup>1</sup>Universidad del Valle de México, Campus San Rafael.

Sadi Carnot 57, San Rafael, Cuauhtémoc, 06470 Ciudad de México, CDMX

<sup>2</sup>Departamento de Sistemas y Computación, Tecnológico Nacional de México, Campus Cuautla  
Libramiento Cuautla-Oaxaca S/N Juan Morales, 62745 Cuautla, Mor.

<sup>3</sup>Coordinación de Posgrado, Tecnológico Nacional de México, Campus Cuautla  
Libramiento Cuautla-Oaxaca S/N Juan Morales, 62745 Cuautla, Mor.  
monica.acosta@cuautla.tecnm.mx, 0000-0001-5564-8523.

[leonor.angeles@cuautla.tecnm.mx](mailto:leonor.angeles@cuautla.tecnm.mx)

### KEYWORDS:

API, Database Client,  
data model, data  
warehouse, Entity  
Relationship Diagram,  
SQL, Extract,  
Transform and Load  
(ETL)

### ABSTRACT

Effective data management is essential for businesses beyond the supply chain because it influences decision-making, customer experience, marketing strategies, product development, operational efficiency, compliance, and risk mitigation. By recognizing the significance of data and implementing robust data management practices, companies can unlock valuable insights, drive innovation, and maintain a competitive advantage in the digital age. ETL is a process used to collect data from various sources. Its implementation could unlock business potential by automating data-driven insights. By harnessing the capabilities of ETL, organizations can overcome data fragmentation, achieve data integrity, and gain a competitive edge in today's data-centric landscape. The objective of this paper is to demonstrate how the extraction, transformation and loading of sales for a bulb manufacturer was automated. In which there was a record of more than 2,000 weekly sales orders both in physical stores and on the website. Performing this task manually presents several challenges, however, through ETL it was automated to collect specific information from all stores in a single repository in a matter of seconds.

### PALABRAS

#### CLAVE:

API, Base de datos de  
cliente, modelo de  
datos, data warehouse,  
Diagrama de entidad-  
relación, SQL, Extraer,  
Transformar y Cargar  
(ETL)

### RESUMEN

La gestión eficaz de datos es esencial para las empresas más allá de la cadena de suministro porque influye tanto en la toma de decisiones, la experiencia del cliente, las estrategias de marketing, el desarrollo de productos, la eficiencia operativa, el cumplimiento y la mitigación de riesgos. Al reconocer la importancia de los datos e implementar prácticas sólidas en su gestión, las empresas podrían identificar información valiosa, impulsar la innovación y mantener una ventaja competitiva en la era digital. ETL es un proceso utilizado para recopilar datos de diversas fuentes. Su implementación podría desbloquear el potencial comercial al automatizar los conocimientos basados en datos. Al aprovechar las capacidades de ETL, las organizaciones pueden superar la fragmentación de datos, lograr la integridad de los datos y obtener una ventaja competitiva en el panorama actual centrado en los datos. El objetivo de este artículo es demostrar cómo se automatizó la extracción, transformación y carga de las ventas de una manufacturera de bulbos. En la que se tenía registro de más de 2000 órdenes de ventas semanales tanto en tiendas físicas y página web. Realizar esta tarea de forma manual presenta varios desafíos, sin embargo, mediante ETL se logró su automatización para recopilar información específica de todas las tiendas en un sólo repositorio en cuestión de segundos.

• Recibido: 29 de junio 2023

• Aceptado: 31 de agosto 2023

• Publicado en línea: 7 de noviembre de 2023



## 1. INTRODUCTION

It is through the meticulous combination, transformation, and analysis of this data that organizations can unlock powerful knowledge, make informed decisions, and drive significant business growth [1]. Behind every machine learning model, dashboard, or business insight lies there is data that was meticulously gathered from various sources, harmoniously integrated, and meticulously analyzed to extract its true value and deliver actionable insights to the organization [2].

During this paper, we delved into a real-world scenario where the strategic application of ETL (Extract, Transform, Load) technology brought remarkable transformations in the way data was managed for a prominent bulb manufacturer. This manufacturer was confronted with a formidable challenge: the management of thousands of weekly sales orders, originating from both physical stores and their online e-commerce platform.

Managing such a substantial influx of sales data manually presented a host of intricate challenges [1]. The sheer scale of the task, with data pouring in from multiple sources and in varied formats, made it a time-consuming and error-prone endeavor. Moreover, the urgency to access up-to-date and accurate sales information for informed decision-making and business operations needed a more efficient approach [2].

The strategic use of ETL allowed the company to automate the once labor-intensive process of collecting sales data. This automation significantly reduced the time and effort required to gather and consolidate information from physical stores and their online platform. Rather than relying on manual data entry, which is prone to human error and time-consuming, ETL enabled the rapid extraction of specific sales information, including product details, transaction records, and customer information, from all sales outlets.

Furthermore, the data transformation capabilities of ETL played a pivotal role in ensuring data accuracy and consistency. Through defined transformation rules and processes, raw data was refined and standardized, making it ready for analysis and reporting [4]-[6]. This critical step not only eliminated data inconsistencies but also enhanced the overall quality of the data.

## 2. METHODOLOGY

To demonstrate the importance of ETL to automate the analysis of data in relational databases, the following methodology and steps were implemented.

- Analyze a database of incandescent bulbs manufacturers regarding their sales. A database was used to get real world information regarding the sales of bulb manufacturers. Data was analyzed according to the predefined format already established in the database. For example:

1. Quantitative or Qualitative values.
2. Column attributes type.
3. Relationship within the tables through Primary/Foreign/Candidate keys.
4. Number of decimals.
5. String length.
6. Date Format

- Develop the Entity-Relationship Diagram. The entity-relationship diagram was designed and created according to the database. It provides a visual representation of different data elements and shows how they relate to one another. It also demonstrates how the database system is structured [3]. This structure helps to understand how data is stored, accessed, updated, and queried within the database. It was normalized to:

1. Reduce data duplication.

2. Avoid data modification implications.
3. Simplify queries

- Analysis of results.

- Generate a database according to the ER Diagram. The script was performed using Jupyter Notebook and MySQL Workbench through the database client Python/Connector. Everything is documented in the following pages. MySQL Workbench is a unified visual tool for database modeling and management. Is used to build a data model diagram or even to create and populate databases [7].
- Make the script in Python through an API. Before working with the database client, it was necessary to install the proper libraries and establish a connection between the database (local or cloud) and the client.
- Implement the ETL process to load new orders to the database. ETL is an acronym for "Extract, Transform, and Load" [4]. It is an essential part of the data staging phase in the data warehouse architecture [6]. ETL describes the set of processes that extract data from different source systems, transform the data into a suitable format, and load it into a data warehouse repository for data analytic [4]-[6].

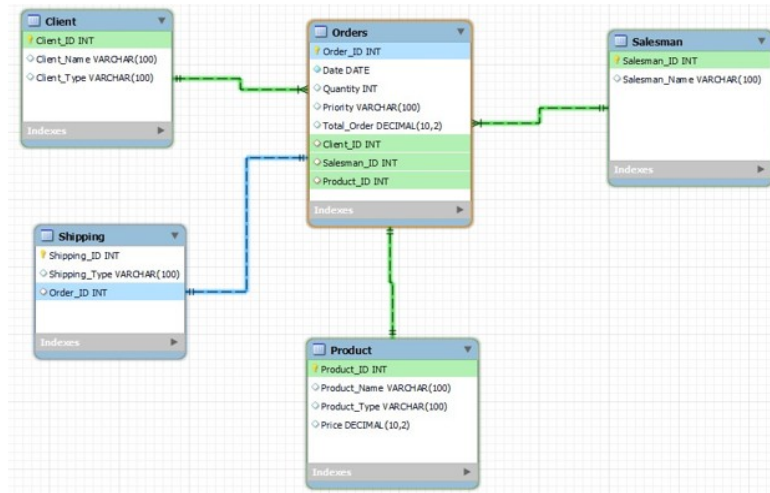
## 4. RESULTS AND ANALYSIS

Before implementing the database, a diagnosis was made of the amount of data to be handled, as well as the digital transformations that should be implemented before its creation. Subsequently, all the data marts (special version of the data warehouse sitting on a subject or specific business area) that would be used for data processing were identified. These could be of any type: other databases (SQL, NoSQL), applications, social networks, surveys, or, as in this case, Excel files. Once this was done, the ETL process (Extraction, Transformation and Load) was performed to extract the data from the different sources, to transform it and to load it into the database. Information about the manufacturer regarding type of customers, shipments, products and number of orders made is shown in Table 1. Which also shows partial records of the sales, 26, but in total this file has information of thousands of sales. And the proposed ER-diagram to represent the data structure, relationships, and rules for storage and manipulation of the manufacturer sales [3], is shown in Schema 1.

Table 1. Partial overview of the excel database

#	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
	ID Orden	Fecha	Cantidad	Prioridad	Tipo Envío	Salesman Name	Salesman ID	Client Name	Client ID	Tipo Cliente	Shipping ID	Product Name	Product	Product Type	Product Price	Total Order
1	1864	22/01/2022	1856	Media	Terrestre	Jose	3	Emily	41	Distribuidor	186441	Halogen Bulb	101	Exterior/Interior	\$5	\$8,380
2	1151	08/07/2020	2530	Alta	Terrestre	Mike	2	Alfredo	32	Mayoreo	115132	Tree Lamp	107	Decoration	\$1,200	\$1,036,000
3	1045	18/12/2021	2552	No especifica	Express Aéreo	Valeria	1	Tina	39	Mayoreo	104539	Halogen Bulb	101	Exterior/Interior	\$5	\$12,760
4	1053	13/09/2022	2620	Media	Regular Aéreo	Jose	3	Marvin	36	Distribuidor	105336	Arched Lamp	103	Decoration	\$350	\$917,000
5	1714	14/01/2020	1081	Baja	Regular Aéreo	Mike	2	Boyer	33	Distribuidor	171433	Smart Bulb	102	Exterior/Interior	\$100	\$108,100
6	1727	03/01/2021	1054	Media	Terrestre	Jose	3	Lilly	31	Mayoreo	172731	Tower Lamp	105	Lamps	\$1,000	\$1,054,000
7	1728	18/03/2022	1619	Media	Terrestre	Valeria	1	Jake	30	Mayoreo	172830	Halogen Bulb	101	Exterior/Interior	\$5	\$8,095
8	1188	24/06/2020	3106	Alta	Terrestre	Jose	3	Renne	42	Otros	118842	LED Bulb	100	Exterior/Interior	\$5	\$15,530
9	1039	27/07/2021	2971	Baja	Terrestre	Jose	3	Boyer	33	Distribuidor	103933	Halogen Bulb	101	Exterior/Interior	\$5	\$14,855
10	2106	14/06/2022	1903	Alta	Regular Aéreo	Mike	2	Ian	37	Mayoreo	210637	Smart Bulb	102	Exterior/Interior	\$100	\$190,300
11	1167	30/07/2021	3072	No especifica	Terrestre	Mike	2	Miley	34	Otros	116734	Halogen Bulb	101	Exterior/Interior	\$5	\$15,560
12	1323	27/10/2021	1550	Media	Express Aéreo	Valeria	1	Renne	42	Otros	132342	Smart Bulb	102	Exterior/Interior	\$100	\$155,000
13	1295	04/11/2020	1230	Media	Terrestre	Mike	2	Lilly	31	Mayoreo	129531	Halogen Bulb	101	Exterior/Interior	\$5	\$6,100
14	1629	13/06/2021	1773	Alta	Express Aéreo	Mike	2	Marvin	36	Distribuidor	162936	Tower Lamp	105	Lamps	\$1,000	\$1,773,000
15	1312	17/03/2020	3530	No especifica	Regular Aéreo	Mike	2	Renne	42	Otros	131242	LED Bulb	100	Exterior/Interior	\$5	\$17,600
16	1508	04/08/2020	3567	Alta	Terrestre	Ana	4	Renne	42	Otros	150842	Tree Lamp	107	Decoration	\$1,200	\$4,280,400
17	1883	01/01/2022	1403	Alta	Terrestre	Mike	2	Lilly	31	Mayoreo	188331	Tree Lamp	107	Decoration	\$1,200	\$1,683,600
18	1202	23/05/2021	1736	Alta	Terrestre	Mike	2	Jake	30	Mayoreo	120230	Halogen Bulb	101	Exterior/Interior	\$5	\$8,680
19	1118	21/10/2020	1279	Alta	Terrestre	Ana	4	Renne	42	Otros	111842	Tree Lamp	107	Decoration	\$1,200	\$1,534,800
20	1265	09/07/2022	2612	No especifica	Terrestre	Jose	3	Renne	42	Otros	126542	Smart Bulb	102	Exterior/Interior	\$100	\$261,200
21	1063	02/08/2021	1576	Baja	Terrestre	Ana	4	Marvin	36	Distribuidor	106336	Tower Lamp	105	Lamps	\$1,000	\$1,576,000
22	1997	06/02/2021	1811	Alta	Regular Aéreo	Valeria	1	Tina	39	Mayoreo	199739	Tree Lamp	107	Decoration	\$1,200	\$2,197,200
23	1811	17/01/2022	1971	No especifica	Terrestre	Jose	3	Boyer	33	Distribuidor	181133	Tower Lamp	105	Lamps	\$1,000	\$1,971,000
24	1899	26/02/2020	1959	Media	Express Aéreo	Jose	3	Renne	42	Otros	189942	Tower Lamp	105	Lamps	\$1,000	\$1,959,000
25	1124	01/10/2022	3676	Media	Regular Aéreo	Jose	3	Renne	42	Otros	112442	Smart Bulb	102	Exterior/Interior	\$100	\$187,600
26	1429	24/06/2020	2288	Alta	Terrestre	Mike	2	Renne	42	Otros	142942	Tree Lamp	107	Decoration	\$1,200	\$2,745,600

Schema 1. Logical data model.



The following figures show the steps described in the methodology. Firstly, a successful database connection was implemented using Python Connector with MySQL Workbench and Jupyter Notebook using the proper credentials: username, password and port. as shown in Fig. 1. Then, a cursor was created to interact between MySQL and Python. A try-except code was implemented to catch any error during the login, this is shown in Fig. 2.

```
try:
    connection=connector.connect(host="localhost",
                                user="root", # use your own
                                password="root", # use your own
                                port= 3307,
                                use_pure=True
                                )
    print("Connection between MySQL and Python is established.\n")
except connector.Error as er:
    print("Error code: ", er.errno)
    print("Error message: ", er.msg)
```

Figure 1. Database Client Connection.

```
cursor = connection.cursor(buffered=True)
print("Cursor is created to communicate with the MySQL using Python.\n")

try:
    cursor.execute("CREATE DATABASE Bulb_Sales")
except:
    cursor.execute("drop database Bulb_Sales")
    cursor.execute("CREATE DATABASE Bulb_Sales")
print("The database Bulb_Sales was created.\n")
cursor.execute("USE Bulb_Sales")
print("Bulb_Sales Database ready to use.\n")
```

Figure 2. Database "Bulb\_Sales" creation.

After the creation of the database. The tables were implemented according to the ER-Diagram and the keys (foreign keys,

primary keys) and data types predefined. Fig. 3, shows the logical diagram to create the tables Client and Salesman which were implemented successfully into MySQL Workbench. Once the tables were created for the database. These were populated with the data from the bulb manufacturers sales (Table 1). Fig. 4 represents the Initial Load, which is a partial load to record the sales information partially into the database to eventually implement an "Incremental Load" to record only new and updated values into the database.

```
# Clients table
create_client_table="""
CREATE TABLE Clients (
    Client_ID INT AUTO_INCREMENT,
    Client_Name VARCHAR(100) NOT NULL,
    Client_Type VARCHAR(100) NOT NULL,
    PRIMARY KEY (Client_ID)
);"""

cursor.execute(create_client_table)
print("Clients table was created")

# Salesman table
create_salesman_table="""
CREATE TABLE Salesman(
    Salesman_ID INT AUTO_INCREMENT,
    Salesman_Name VARCHAR(100) NOT NULL,
    PRIMARY KEY (Salesman_ID)
);"""

cursor.execute(create_salesman_table)
print("Salesman table was created")
```

Figure 3. "Clients" and "Salesman".

```
#Inserting values to Salesman Table
insert_salesman="""
INSERT INTO Salesman (Salesman_ID, Salesman_Name)
VALUES
(1, 'Valeria'),
(2, 'Mike'),
(3, 'Jose'),
(4, 'Ana')
;"""

cursor.execute(insert_salesman)
connection.commit()
print("Values inserted into Salesman Table")
print("There are 4 Salesman: Valeria, Mike, Jose and Ana ")
```

Figure 4. Initial load of Salesman table.

For reviewing purposes, once the tables were populated an inspection of the data inserted was performed. For that reason, many stored procedures were implemented, not only to show the data after insertions, but also to automate the process or need to write many lines of code every time it was required to check the full content of each table. Fig. 5 shows the code used to create the stored procedures for the Salesman table. The structure to create the stored procedures for each table of the database is the same, but with different arguments.

```
#Show Salesman Table
cursor.execute("DROP PROCEDURE IF EXISTS Salesman;")

# Creating the procedure
GetSalesman = """
CREATE PROCEDURE Salesman()
BEGIN
    SELECT * FROM Salesman;
END
"""

# Execute the query
cursor.execute(GetSalesman)

# Calling the store procedure
cursor.callproc('Salesman')

# Retrieving records in a dataset
results = next(cursor.stored_results())
dataset = results.fetchall()

# Retrieve column names using List comprehension in a 'for' loop
for column_id in cursor.stored_results():
    columns = [ column[0] for column in column_id.description ]

# Print column names
print(columns)

# Print data
for data in dataset:
    print(data)

['Salesman_ID', 'Salesman_Name']
(1, 'Valeria')
(2, 'Mike')
(3, 'Jose')
(4, 'Ana')
```

Figure 5. Stored procedure structure.

To perform ETL process and load the sales information from another file into the “Bulb Sales” database. Firstly, the file was converted

to CSV format and PANDAS library was downloaded and installed in the Notebook to enable analysis and data manipulation. Fig. 6 shows that Pandas was properly in the notebook.

```
#ETL PROCESS

#Importing PANDAS library to perform data analysis and manipulation
!pip install pandas
import pandas as pd
print("pandas installed and imported to the instance")

Requirement already satisfied: pandas in c:\users\lorus\appdata\local\programs\python\python310\lib\site-packages (2.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\lorus\appdata\local\programs\python\python310\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: tzdata>=2022.1 in c:\users\lorus\appdata\local\programs\python\python310\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: numpy>=1.21.0 in c:\users\lorus\appdata\local\programs\python\python310\lib\site-packages (from pandas) (1.25.0)
Requirement already satisfied: pytz>=2020.1 in c:\users\lorus\appdata\local\programs\python\python310\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\lorus\appdata\local\programs\python\python310\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
pandas installed and imported to the instance
```

Figure 6. Pandas installed.

During data extraction, raw data was copied or exported from source locations to the staging area. Data management teams extracted data from a variety of data sources, which could be structured or unstructured data formats. Those sources included but were not limited to [4]–[6]:

- 1 SQL or NoSQL servers
- 2 CRM and ERP systems
- 3 Flat files

Data was extracted from the “Data” file by creating a data frame and using Panda’s function “read.csv”. The second line shows the content of the Data.csv file. The records are not sorted in any order as shown in Fig. 7.

```
#DATA EXTRACTION
data_frame = pd.read_csv('data.csv')
print("Data extracted from CSV FILE")
```

Data extracted from CSV FILE

Figure 7. CSV file data extraction

During the staging area, the raw data underwent data processing. Here, the data was transformed and consolidated for its intended analytical use case. This phase involved the following tasks [4]:

- 1 Filtering, cleansing, de-duplicating, validating, and authenticating the data.



- 2 Performing calculations, translations, or summarizations based on the raw data. Like changing row and column headers for consistency, converting currencies or other units of measurement, editing text strings, and more.
- 3 Conducting audits to ensure data quality and compliance.
- 4 Removing, encrypting, or protecting data governed by industry or governmental regulators.
- 5 Formatting the data into tables or joined tables to match the schema of the target data warehouse.

For this case, it was not necessary to perform any transformation of the data because it was formatted to the data model characteristics. However, "ID Orden" was sorted in ascending order for visualizing purposes. This is shown in Fig. 8. Once the new variable was sorted, an inspection was made to double check there were no errors as shown in Fig. 9. And finally, the new variable was saved as "sorted\_data.csv" before implementing the Load process in the database, as shown in Fig. 10.

```
#Transformation Process
#Sorting ID ORDEN in a new variable
data_frame2 = data_frame.sort_values('ID Orden')
data_frame2 = pd.DataFrame(data_frame2)
```

Figure 8. Sorting the new variable.

```
# printing sorted file
data_frame2
```

ID Orden	Fecha	Cantidad	Prioridad	Tipo Envío	Salesman_Name	Salesman_ID	Client_Name	Client_ID	Tipo Cliente	Shipping_ID	Pr
1024	2021-12-09	1787	Baja	Regular Aéreo	Jose	3	Renne	42	Otros	102442	
1185	2021-05-13	3789	Media	Regular Aéreo	Valeria	1	Renne	42	Otros	118542	
1231	2021-12-05	1535	Media	Terrestre	Jose	3	Tina	39	Mayoreo	123139	
1309	2020-06-05	3645	Baja	Express Aéreo	Jose	3	Alfredo	32	Mayoreo	130932	
1383	2020-06-30	2372	Alta	Express Aéreo	Ana	4	Renne	42	Otros	138342	
1385	2020-02-12	3111	Media	Terrestre	Mike	2	Ian	37	Mayoreo	138537	
1468	2022-02-21	3130	Alta	Terrestre	Jose	3	Alfredo	32	Mayoreo	146832	
1558	2020-01-06	1490	Media	Regular Aéreo	Jose	3	Marvin	36	Distribuidor	155836	
1927	2021-03-06	1083	Media	Terrestre	Mike	2	Ian	37	Mayoreo	192737	
2070	2020-12-27	1853	Baja	Express Aéreo	Jose	3	Renne	42	Otros	207042	
2093	2022-08-01	1065	Media	Regular Aéreo	Jose	3	Miley	34	Otros	209334	

Figure 9. Sorted data.

```
#saving the new variable
data_frame2.to_csv('sorted_data.csv')
```

Figure 10. Saving the new instance

During the load the transformed data was moved from the staging area into a target data warehouse. This involved an initial loading of all data, followed by periodic loading of incremental data changes to fill missing or updated data in the repository [2]-[6].

The incremental load was performed using MySQL Workbench and the file generated in the data transformation "sorted data" was loaded to the existing database. Fig. 11 shows the records after the initial load in the Orders table.

1 • SELECT \* FROM bulb\_sales.orders;

Order_ID	Date	Quantity	Priority	Total_Order	Client_ID	Salesman_ID	Product_ID
1629	2021-08-13	1773	Alta	1773000.00	36	2	105
1663	2021-08-02	1576	Baja	1576000.00	36	4	105
1714	2020-01-14	1081	Baja	108100.00	33	2	102
1727	2021-01-03	1054	Media	1054000.00	31	3	105
1728	2022-03-18	1619	Media	8095.00	30	1	101
1864	2022-01-22	1656	Media	8280.00	41	3	101
1883	2022-01-01	1403	Alta	1683600.00	31	2	107
1997	2021-02-06	1831	Alta	2197200.00	39	1	107
2106	2022-08-14	1903	Alta	190300.00	37	2	102

Figure 11. "Orders" table records.

The file "sorted\_data.csv" was imported using the existing tables, so no new information was overwritten and the updated values were refreshed for visualization purposes (Incremental Load).

Fig. 12 shows the encoded type used "utf-8" and the columns selected to perform the Load. Values from "Columns" matched the values in "Dest Column" in column type to import the data. Fig. 13 shows there were no errors occurred during the load. Finally, it

was verified that the “Orders” table was updated with the new data. Before the Load the second highest ID was 1997, as shown in Fig. 14, and after the Load, the second and third highest IDs are: 2093 and 2070. This verifies that new data was added to the Orders table, as shown in Fig. 15. Finally, the stored procedure “Orders” was called to double check that new data was inserted in the table and no information was overwritten, as shown in Fig. 16.

Figure 12. Import settings.

Figure 13. Import logs.

1 • SELECT \* FROM bulb\_sales.orders;

Order_ID	Date	Quantity	Priority	Total_Order	Client_ID	Salesman_ID	Product_ID
2106	2022-08-14	1903	Alta	190300.00	37	2	102
1997	2021-02-06	1831	Alta	2197200.00	39	1	107
1883	2022-01-01	1403	Alta	1683600.00	31	2	107
1864	2022-01-22	1656	Media	8280.00	41	3	101
1728	2022-03-18	1619	Media	8095.00	30	1	101
1727	2021-01-03	1054	Media	1054000.00	31	3	105
1714	2020-01-14	1081	Baja	108100.00	33	2	102
1663	2021-08-02	1576	Baja	1576000.00	36	4	105
1629	2021-08-13	1773	Alta	1773000.00	36	2	105
1611	2022-01-17	1971	No Especifica	1971000.00	33	3	105

Figure 14. Orders table before the Incremental Load.

1 • SELECT \* FROM bulb\_sales.orders;

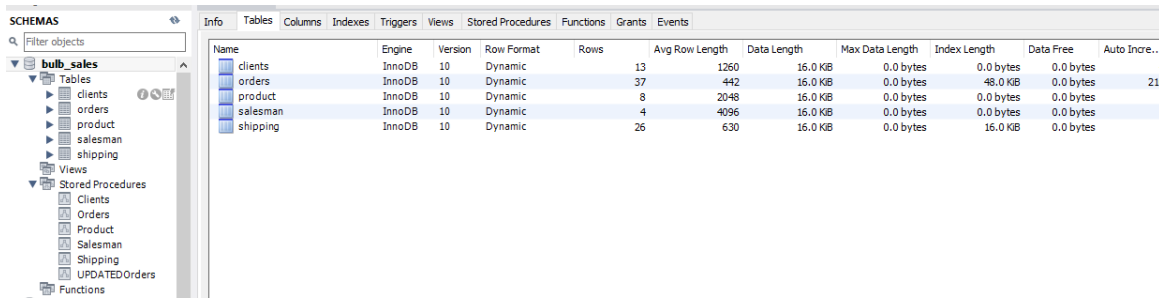
Order_ID	Date	Quantity	Priority	Total_Order	Client_ID	Salesman_ID	Product_ID
2106	2022-08-14	1903	Alta	190300.00	37	2	102
2093	2022-08-01	1065	Media	106500.00	34	3	102
2070	2020-12-27	1853	Baja	9265.00	42	3	100
1997	2021-02-06	1831	Alta	2197200.00	39	1	107
1927	2021-03-06	1083	Media	1299600.00	37	2	107
1883	2022-01-01	1403	Alta	1683600.00	31	2	107
1864	2022-01-22	1656	Media	8280.00	41	3	101
1728	2022-03-18	1619	Media	8095.00	30	1	101
1727	2021-01-03	1054	Media	1054000.00	31	3	105
1714	2020-01-14	1081	Baja	108100.00	33	2	102

Figure 15. Orders table after Incremental Load.

```
['Order_ID', 'Date', 'Quantity', 'Priority', 'Total_Order', 'Client_ID', 'Salesman_ID', 'Product_ID']
(1024, datetime.date(2021, 12, 9), 1787, 'Baja', Decimal('2144400.00'), 42, 3, 107)
(1039, datetime.date(2021, 7, 27), 2071, 'Baja', Decimal('14055.00'), 33, 3, 103)
(1045, datetime.date(2021, 12, 18), 2552, 'No Especifica', Decimal('12760.00'), 39, 1, 101)
(1053, datetime.date(2022, 9, 13), 2620, 'Media', Decimal('917000.00'), 36, 3, 103)
(1118, datetime.date(2020, 10, 21), 1279, 'Alta', Decimal('1534000.00'), 42, 4, 107)
(1124, datetime.date(2022, 10, 1), 3676, 'Media', Decimal('367600.00'), 42, 3, 102)
(1151, datetime.date(2020, 7, 8), 2530, 'Alta', Decimal('3636000.00'), 32, 2, 107)
(1167, datetime.date(2021, 7, 30), 3872, 'No Especifica', Decimal('15360.00'), 34, 2, 101)
(1185, datetime.date(2021, 5, 13), 3769, 'Media', Decimal('4522800.00'), 42, 1, 107)
(1188, datetime.date(2020, 6, 24), 3106, 'Alta', Decimal('15530.00'), 42, 3, 100)
(1202, datetime.date(2021, 5, 23), 1736, 'Alta', Decimal('8600.00'), 30, 2, 101)
(1231, datetime.date(2021, 12, 5), 1535, 'Media', Decimal('7675.00'), 39, 3, 100)
(1265, datetime.date(2022, 7, 8), 2012, 'No Especifica', Decimal('201200.00'), 42, 3, 102)
(1295, datetime.date(2020, 11, 4), 1220, 'Media', Decimal('6100.00'), 31, 2, 101)
(1309, datetime.date(2020, 6, 5), 3645, 'Baja', Decimal('18225.00'), 32, 3, 100)
(1312, datetime.date(2020, 3, 17), 3530, 'No Especifica', Decimal('17050.00'), 42, 2, 100)
(1323, datetime.date(2021, 10, 27), 1550, 'Media', Decimal('155000.00'), 42, 1, 102)
(1383, datetime.date(2020, 6, 30), 2372, 'Alta', Decimal('284400.00'), 42, 4, 107)
(1385, datetime.date(2020, 2, 12), 3111, 'Media', Decimal('1555000.00'), 37, 2, 104)
(1429, datetime.date(2020, 6, 24), 2288, 'Alta', Decimal('2745600.00'), 42, 2, 107)
(1468, datetime.date(2022, 2, 21), 2130, 'Alta', Decimal('3756000.00'), 32, 3, 107)
(1499, datetime.date(2020, 2, 26), 1959, 'Media', Decimal('1359000.00'), 42, 3, 105)
(1508, datetime.date(2020, 8, 4), 3567, 'Alta', Decimal('4280400.00'), 42, 4, 107)
(1558, datetime.date(2020, 1, 8), 1490, 'Media', Decimal('7450.00'), 36, 3, 103)
(1611, datetime.date(2022, 1, 17), 1971, 'No Especifica', Decimal('1971000.00'), 33, 3, 105)
(1629, datetime.date(2021, 8, 13), 1773, 'Alta', Decimal('1773000.00'), 36, 2, 105)
(1663, datetime.date(2021, 8, 2), 1576, 'Baja', Decimal('1576000.00'), 36, 4, 105)
(1714, datetime.date(2020, 1, 14), 1081, 'Baja', Decimal('108100.00'), 33, 2, 102)
(1727, datetime.date(2021, 1, 3), 1054, 'Media', Decimal('1054000.00'), 31, 3, 105)
(1728, datetime.date(2022, 3, 18), 1619, 'Media', Decimal('8095.00'), 30, 1, 101)
(1864, datetime.date(2022, 1, 22), 1656, 'Media', Decimal('8280.00'), 41, 3, 101)
(1883, datetime.date(2022, 1, 1), 1403, 'Alta', Decimal('1683600.00'), 31, 2, 107)
(1927, datetime.date(2021, 3, 6), 1083, 'Media', Decimal('1299600.00'), 37, 2, 107)
(1997, datetime.date(2021, 2, 6), 1831, 'Alta', Decimal('2197200.00'), 39, 1, 107)
(2070, datetime.date(2020, 12, 27), 1853, 'Baja', Decimal('9265.00'), 42, 3, 100)
(2093, datetime.date(2022, 8, 1), 1065, 'Media', Decimal('106500.00'), 34, 3, 102)
(2106, datetime.date(2022, 8, 14), 1903, 'Alta', Decimal('190300.00'), 37, 2, 102)
```

Figure 16. Data verification

The “loading” phase marked the pinnacle of transformation for the bulb manufacturer's data management strategy. During this phase, all the meticulously refined and standardized data from various sources seamlessly converged into a single, centralized repository.



Name	Engine	Version	Row Format	Rows	Avg Row Length	Data Length	Max Data Length	Index Length	Data Free	Auto Incre...
clients	InnoDB	10	Dynamic	13	1260	16.0 KiB	0.0 bytes	0.0 bytes	0.0 bytes	4
orders	InnoDB	10	Dynamic	37	442	16.0 KiB	0.0 bytes	48.0 KiB	0.0 bytes	210
product	InnoDB	10	Dynamic	8	2048	16.0 KiB	0.0 bytes	0.0 bytes	0.0 bytes	
salesman	InnoDB	10	Dynamic	4	4096	16.0 KiB	0.0 bytes	0.0 bytes	0.0 bytes	
shipping	InnoDB	10	Dynamic	26	630	16.0 KiB	0.0 bytes	16.0 KiB	0.0 bytes	

Figure 17. Resulting database.

This repository, often referred to as a data warehouse or data store, held the immense potential to revolutionize how the company conducted its day-to-day operations and strategic decision-making [2].

Once all code above was run on Jupyter the changes were reflected on MySQL Workbench. Everything was created: the database, the tables, the stored procedures and even the database was altered after the Load of the ETL process. Fig. 17 shows the results.

#### 4. CONCLUSION

The creation of this consolidated repository was nothing short of a game-changer. Instead of dispersing sales data across multiple platforms, formats, and locations, the ETL system had successfully brought order to the chaos. In a matter of seconds, the company could access a comprehensive and up-to-date treasure trove of sales insights, which proved to be instrumental in several critical aspects of their business.

First and foremost, this centralized repository became the epicenter of informed decision-making. Previously, extracting relevant data and generating meaningful reports was a time-consuming and often error-prone endeavor. With the data readily available in one location, decision-makers could swiftly access vital information, such as sales trends, customer preferences, and product performance. Armed with this knowledge, they could make strategic decisions with confidence and precision. For

instance, they could quickly identify which products were selling well, target marketing campaigns more effectively, and adjust inventory levels in real-time to meet customer demand.

Moreover, the centralized data repository had a profound impact on the company's marketing strategies. Marketing teams could now analyze customer behavior, preferences, and buying patterns in a granular manner. This insight allowed them to tailor marketing campaigns with a high degree of personalization, thereby improving customer engagement and conversion rates. By aligning marketing efforts closely with customer data, the company achieved greater efficiency and effectiveness in its marketing initiatives.

Inventory management also underwent a substantial transformation. The company was no longer reliant on manual inventory counts and guesswork to replenish stock. Instead, they could leverage real-time sales data from the centralized repository to optimize inventory levels. This not only minimized the risk of overstocking or under-stocking but also improved overall operational efficiency and cost management.

In essence, the centralized data repository, made possible by the ETL process, became the cornerstone of the company's overall business operations. It served as a dynamic source of actionable insights, driving data-driven decision-making, enhancing marketing precision, and streamlining inventory management. By harnessing the power of their data in this way, the bulb manufacturer not only thrived in a competitive market but



also laid the foundation for continued growth and innovation in the digital age.

## REFERENCES

- [1] Stedman. C. *What is data management and why is it important?* Recuperado el 5 de agosto de 2023 de <https://www.techtarget.com/searchdatamanagement/definition/data-management>, 2023.
- [2] Densmore. J. *Data Pipelines Pocket Reference*. O'Reilly. 2021.
- [3] Song, I-Y., Froehlich, K. Entity-relationship modeling, *IEEE Potentials*. 1995, 13(5), 29-34, doi: [10.1109/45.464652](https://doi.org/10.1109/45.464652).
- [4] IBM. ETL (Extraer, transformar, cargar). Recuperado el 10 de agosto de 2023 de <https://www.ibm.com/mx-es/topics/etl>, 2022.
- [5] Bernabeu, R. *Data warehousing: Investigación y sistematización de conceptos*. Recuperado el 15 agosto de 2023 de [https://www.dataprix.com/files/DWH\\_Metodologia\\_HEFESTO-V1.0.pdf](https://www.dataprix.com/files/DWH_Metodologia_HEFESTO-V1.0.pdf), 2009.
- [6] Jhawar. R. Extracción, transformación y carga de datos (ETL). Recuperado el 10 de agosto de 2023 de <https://learn.microsoft.com/es-es/azure/architecture/data-guide/relational-data/etl>, 2023
- [7] MySQL. *MySQL Documentation*. Recuperado el 15 de agosto de 2023 de <https://dev.mysql.com/doc>, 2023.

## ACERCA DE LOS AUTORES



**Leonardo J. Robles Angeles**, es Ingeniero Biomédico egresado del Tec de Monterrey y estudiante de maestría en Ciencia de Datos. Contribuciones a la Industria: (a) Desarrolló una plataforma para mejorar la trazabilidad del equipo médico y su mantenimiento en el Hospital San José, Tec salud, Monterrey, (b) Incrementó en 5% el SLA Agreement en Bees, Anheuser Busch. Su artículo actual busca resaltar cómo se utilizan y explotan los datos para impulsar la innovación y mantener una ventaja competitiva en la industria digital.



**Marving Bryan Robles Angeles**, es Ingeniero en Ciencias de la Computación y Tecnología, graduado del ITESM Monterrey, y actualmente cursa una maestría en Ciencia de Datos.

Este trabajo de investigación destaca la importancia de la implementación de ETL para automatizar la obtención de conocimientos basados en datos, lo que permite a las organizaciones superar la fragmentación de datos, lograr la integridad de los datos y obtener una ventaja competitiva en el panorama actual centrado en los datos.



**M.A. Leonor Ángeles Hernández**, es Licenciada en Informática, por el Instituto Tecnológico de Zacatepec, Maestría en Administración en la Universidad Autónoma del Estado de Morelos, Forma parte de un cuerpo académico "Gestión, Innovación y Sistematización en las organizaciones". Profesora certificada en Gestión de proyectos de Google, Informática Administrativa por ANFECA, en impartición de cursos de formación del capital humano de manera presencial grupal y Diseño de estrategias didácticas aplicando tecnologías de la información y comunicación por CONOCER. Cuenta con Reconocimiento a Perfil Deseable de PRODEP desde 2012. Ha participado en diversos Congresos, proyectos e investigaciones y realizado publicaciones en revistas, memorias y capítulos de libros.



**M.A. Mónica Leticia Acosta Miranda**, es Coordinadora de Posgrado y docente del Depto. de Ciencias Económico-Administrativas del TecNM campus Cuautla. Contador Público y Maestra en Administración por la Universidad Autónoma del Estado de Morelos. Doctora en Ciencias de la Administración por el Instituto de Estudios

Superiores (IEU) campus Puebla. Académico certificado en Contaduría Pública por ANFECA y con diversas certificaciones expedidas por CONOCER. Reconocimiento a Perfil Deseable de PRODEP desde 2013. Integrante de la Comisión de Prevención de Lavado de dinero del Colegio de Contadores Públicos de Cuautla, A.C. y de la Región Centro del Instituto Mexicano de Contadores Públicos. Ha participado en diversos Congresos a nivel nacional e internacional, proyectos e investigaciones y realizado publicaciones en revistas, memorias y capítulos de libros.