DATA STORAGE SOLUTIONS FOR DATA ANALYTICS

- PRESENTED BY
- Peter Ibeabuchi (studentID: 20007349)
- Elisha Johnson Kyanchat (Student ID: 20002405)
- Prashanth Periannan (StudentID: 20001940)

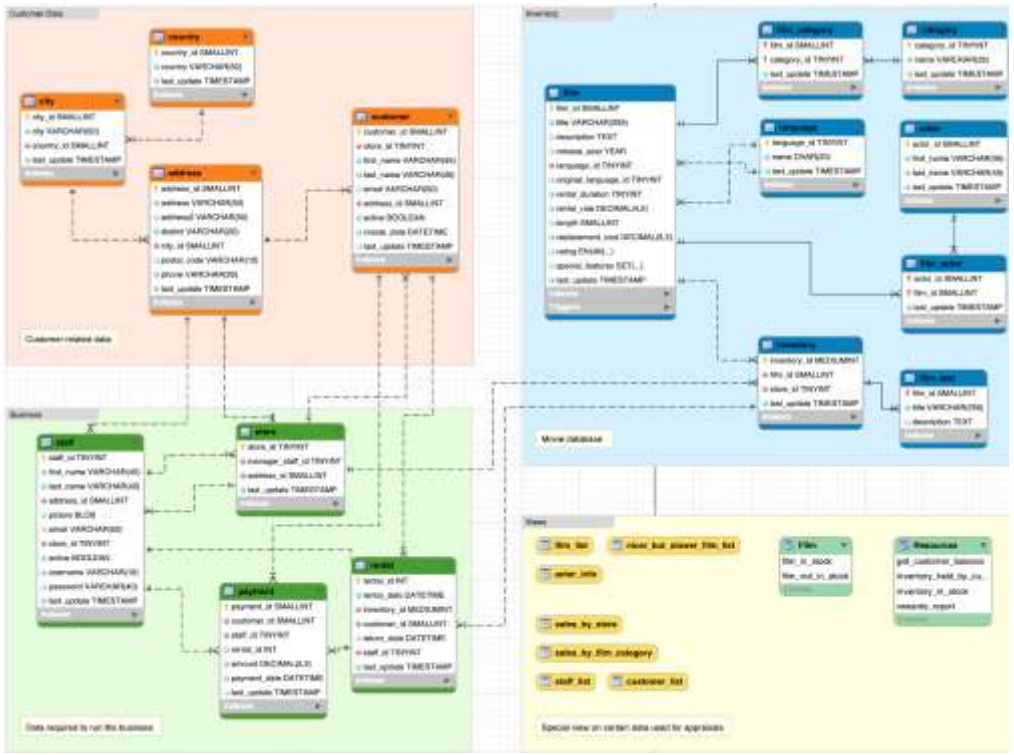
TO

BERNIE LYDON

April, 2024

The Database

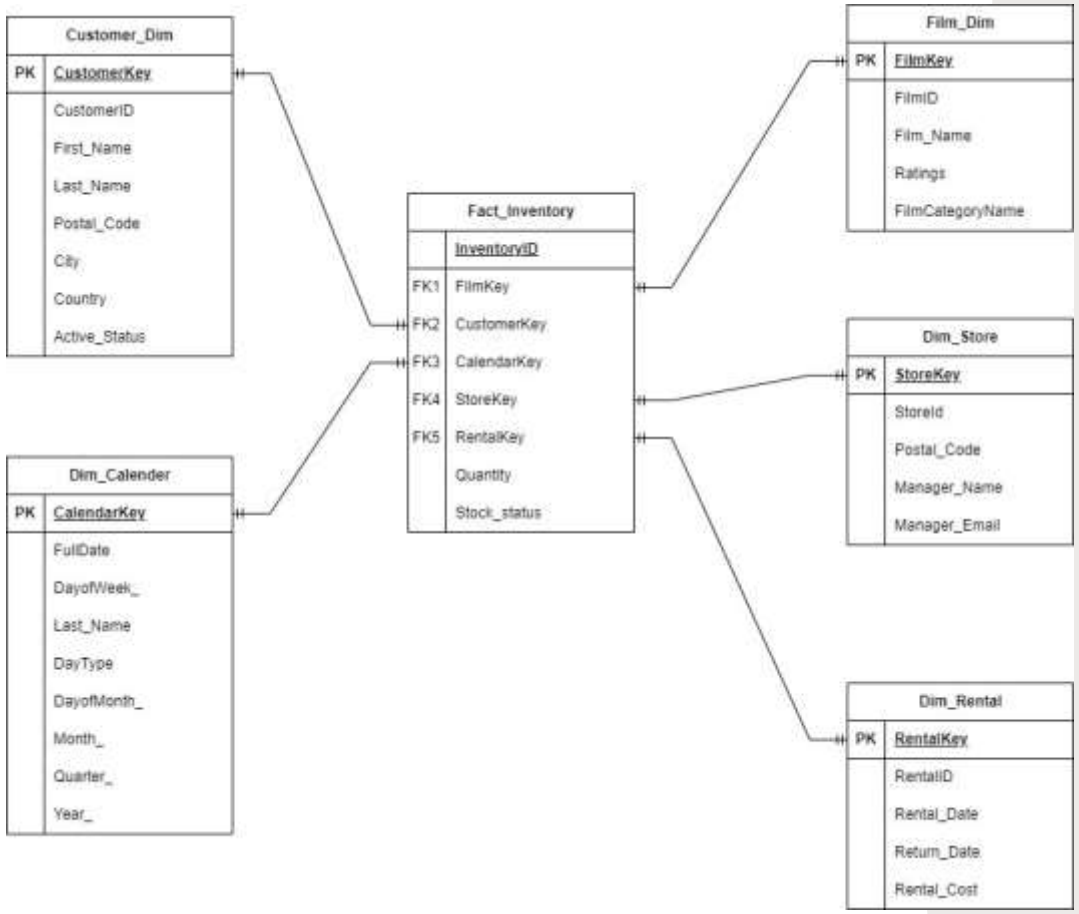
- This project focuses on creating a data warehouse for the Sakila database. The sakila database is a popular MySQL database showing a business film rental activity. It records the movies being rented out, the payments for each rental, the customer details, and many other records. See more about it here.



Business Purpose and Objectives

THE GOAL IS TO DEVELOP A DATA WAREHOUSE THAT INTEGRATES VARIOUS DIMENSIONS LIKE CUSTOMERS, FILMS, RENTALS, CALENDAR, AND STORE TO HAVE A UNIFIED VIEW OF THE BUSINESS OPERATIONS. THIS INTEGRATION WILL ALLOW FOR EFFICIENT INVENTORY MANAGEMENT, ENHANCED CUSTOMER EXPERIENCE, AND OPTIMIZED RENTAL PROCESSES.

THE DATA WAREHOUSE ANSWERS ONE MOST IMPORTANT QUESTION AMONGST OTHERS; HOW CAN WE EFFICIENTLY MONITOR AND MANAGE MOVIE INVENTORY ACROSS ALL STORE LOCATIONS TO ENSURE OPTIMAL AVAILABILITY, AND TRACKING OF MOVIE STOCK?



Inventory Data Warehouse Schema

- The **film dimension** table allows us capture what films we have in the database, the total quantities, and their availability status.
- The **rental dimension** table allows us capture what movies are currently rented out, the cost for rentals, the return date for every rental, etc. and so on.
- The **calendar dimension table** helps us monitor the dates rental activities in the database. With this we can tell the total inventory available at every given date.
- The **customer table** helps us monitor what customers have sets of the films across the globe, this makes it easy to track each movie rented out.
- The **store dimension table** helps us monitor the inventory in each store at every given time.

Business Objectives and Key StakeHolders

Business Objectives

From the data we seek to understand the following insights

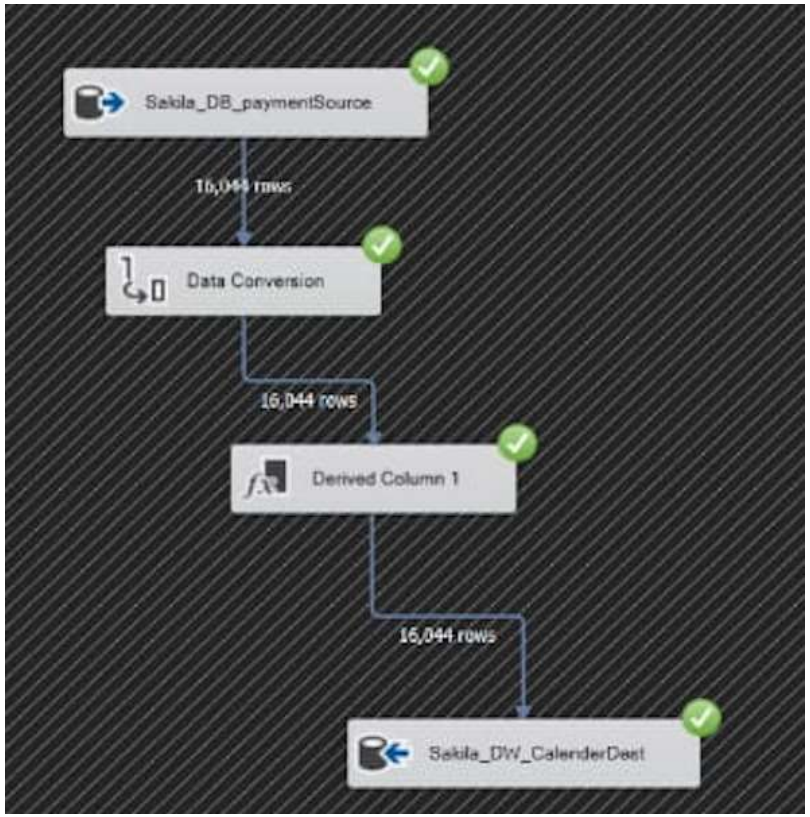
1. Determine the current inventory levels for each film in the database to facilitate efficient stock management.
2. Calculate the total inventory across various store locations to optimize distribution and ensure sufficient availability.
3. Analyze the geographic distribution of films to identify market trends and tailor inventory strategies accordingly.
4. Identify movies with the highest quantities in the database to understand popular demand and inform purchasing decisions.
5. Monitor stock levels to identify films at risk of running low and take proactive measures to replenish inventory.

•

key stakeholders

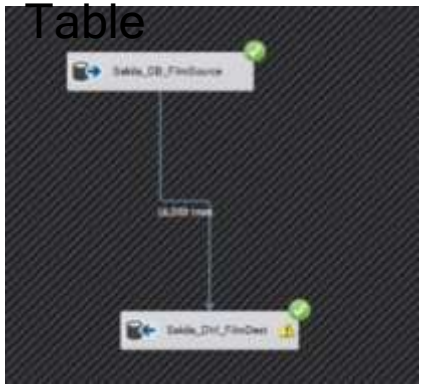
1. **Business Analysts:** They analyze the data for insights into customer preferences and behavior.
2. **Management Team:** They utilize the insights derived from the data to make strategic decisions.
3. **Marketing Team:** Insights into customer demographics and movie preferences can inform targeted marketing campaigns to promote specific movies to likely renters.
4. **Store Managers:** Data from the warehouse can empower store managers to optimize inventory levels at their specific stores, identify popular genres among their customer base, and tailor promotions accordingly.

Calendar Table



Store Table

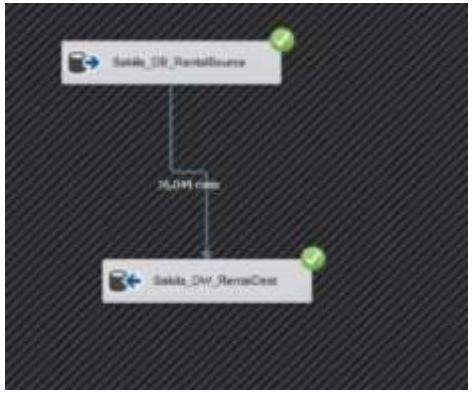
Film Table



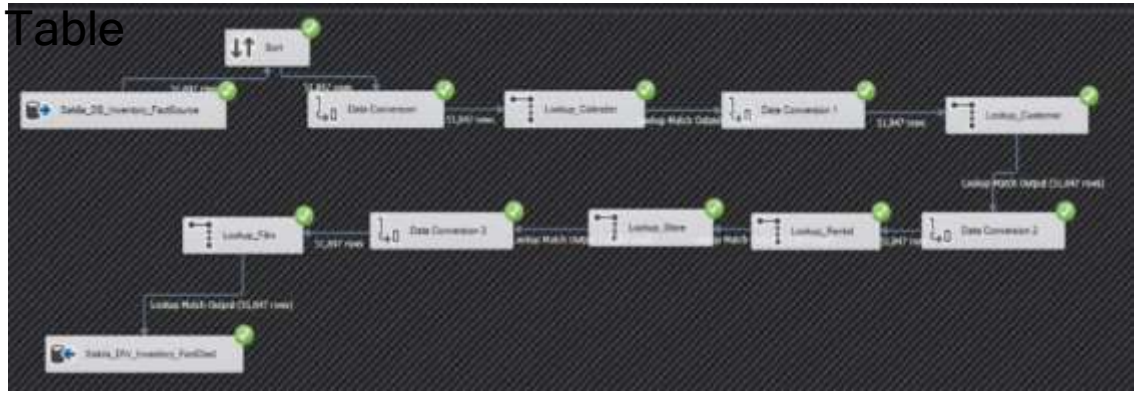
Extract Transform Load

The data was extracted from the databse using SQL Server Integration services(SSIS).

Rental Table

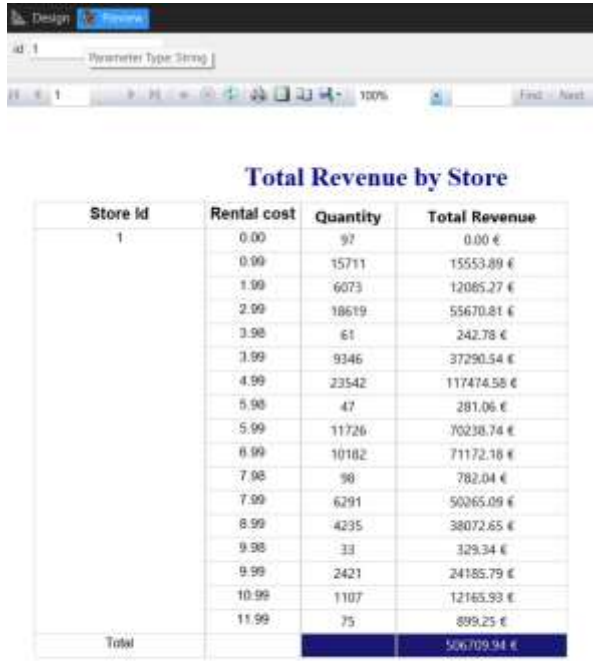


Fact Table



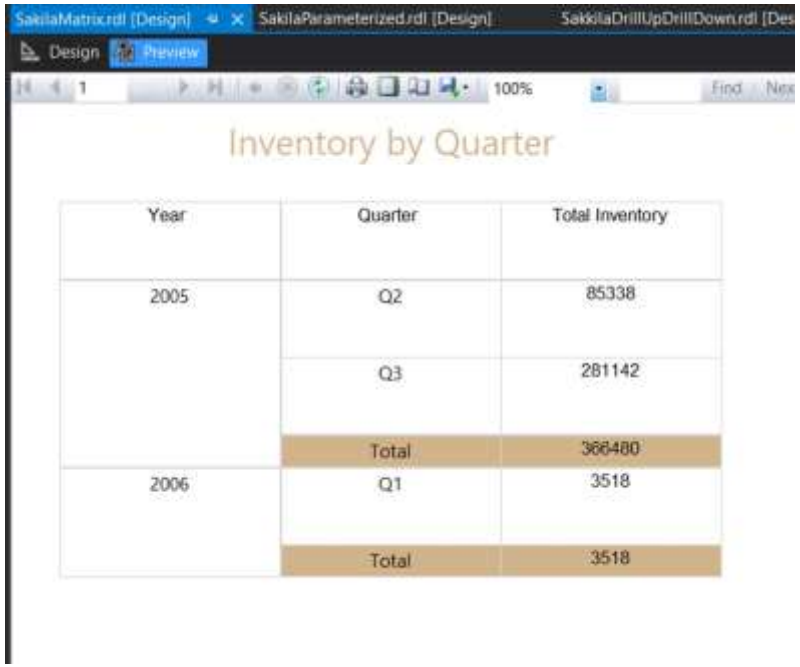
Analytical Report

SQL SERVER REPORT SERVICES(SSRS)



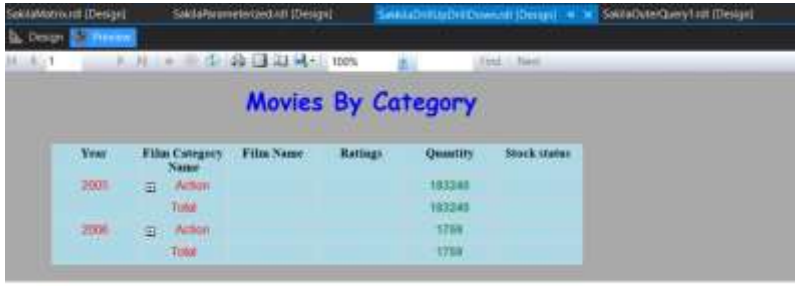
Total Revenue by Store

Store Id	Rental cost	Quantity	Total Revenue
1	0.00	97	0.00 €
	0.99	15711	15553.89 €
	1.99	6073	12085.27 €
	2.99	18619	55670.81 €
	3.98	61	242.78 €
	3.99	9346	37290.54 €
	4.99	23542	117474.58 €
	5.98	47	281.06 €
	5.99	11726	70238.74 €
	6.99	10182	71172.18 €
	7.98	98	782.04 €
	7.99	6291	50265.09 €
	8.99	4235	38072.65 €
	9.98	33	329.34 €
	9.99	2421	24185.79 €
	10.99	1107	12165.93 €
	11.99	75	899.25 €
Total			506709.94 €



Inventory by Quarter

Year	Quarter	Total Inventory
2005	Q2	85338
	Q3	281142
	Total	366480
2006	Q1	3518
	Total	3518

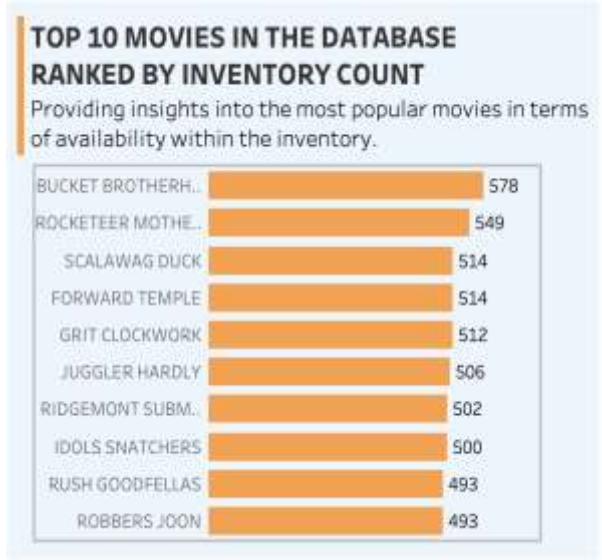


Movies By Category

Year	Film Category Name	Film Name	Ratings	Quantity	Stock status
2005	Action			183240	
	Total			183240	
2006	Action			1759	
	Total			1759	

Analytical REPORT

TABLEAU REPORTS



SQL V/s Graph Databases

SQL (Structured Query Language) and graph databases are two different approaches to managing and querying data in a database. Here we compare seven various queries of both SQL and graph database to get an understanding of how both works.

QUESTION 1: A simple select statement to show the details on the Menu Table

SQL→

```
3 •  SELECT
4    product_id,
5    product_name,
6    price
7    FROM dannys_diner.menu
```

00% 6:6

Result Grid Filter Rows: Search

product_id	product_na...	price
1	sushi	10
2	curry	15
3	ramen	12

Cypher Query→

```
1  Match(me:Menu)
2  RETURN me.product_id, me.product_name, me.price
```

	me.product_id	me.product_name	me.price
1	1	"sushi"	10.0
2	2	"curry"	15.0
3	3	"ramen"	12.0

SQL V/s Graph Databases

Question 2: **A query to show the total revenue generated by each customer.**

SQL

```
3 •  SELECT
4    S.customer_id AS Customers,
5    sum(M.price) AS Revenue
6    From sales AS S
7    Join menu AS M
8    on S.product_id= M.product_id
9    Group by S.customer_id
```

100% 18:1

Result Grid Filter Rows: Search Export:

Customers	Revenue
A	76
B	74
C	36

Cypher Query

```
1 // Total revenue generated from each customer
2 MATCH (s:Sales)⟶(me:Menu)
3 RETURN s.customer_id as customer, SUM(me.price) as total_amount_spent
```

Table Text Code

	customer	total_amount_spent
1	"B"	74.0
2	"A"	76.0
3	"C"	36.0

SQL V/s Graph Databases

Question 3: **A query to show the total revenue generated by each product.**

SQL

```
3 •  SELECT
4    m.product_name AS Products,
5    m.price As 'Unit Price',
6    sum(price) As Revenue
7    From sales AS s
8    Join menu as m
9    on s.product_id= m.product_id
10   Group by m.product_name, m.price
```

100% 22:6

Result Grid | Filter Rows: Search | Export:

Products	Unit Price	Revenue
sushi	10	30
curry	15	60
ramen	12	96

Cypher Query

```
1  // Total revenue generated by each product
2  MATCH (s:Sales)⟶(me:Menu)
3  RETURN me.product_name as product,
4  me.price as unit_price,
5  Sum(me.price) as Revenue
```

Table | Text | Code

	product	unit_price	Revenue
1	"sushi"	10.0	30.0
2	"curry"	15.0	60.0
3	"ramen"	12.0	96.0

SQL V/s Graph Databases

Question 4: **A query to show the number of times each product was purchased**

SQL

```
3 •  SELECT
4    m.product_name AS Products,
5    Count(s.product_id) AS 'Nubmber of times purchased'
6    From sales AS s
7    Join menu as m
8    on s.product_id= m.product_id
9    Group by product_name
```

100% 18:1

Result Grid Filter Rows: Search Export:

Products	Nubmber of times purcha...
sushi	3
curry	4
ramen	8

Cypher Query

```
1  // Number of Time each product was purchased
2  MATCH (s:Sales)⟶(me:Menu)
3  RETURN me.product_name as product,
4  count(s.product_id) as Nubmber_of_times_purchased
```

Table Text Code

	product	Nubmber_of_times_purchased
1	"sushi"	3
2	"curry"	4
3	"ramen"	8

SQL V/s Graph Databases

Question 5: A query to show the amount spent by each customer before they became a member.

SQL

```sql
Select
s.customer_id,
Count(s.product_id) as 'Total item',
sum(m.price) as 'Amount spent'
From sales as s
Join menu as m
on s.product_id = m.product_id
Join members as me
on s.customer_id = me.customer_id
AND s.order_date > me.join_date
```

100% 18:1

Result Grid | Filter Rows: Search | Export:

customer_id	Total item	Amount spe...
B	3	34
A	3	36

Cypher Query

```
// Amount spent by each customer before they became a member
MATCH (s:Sales)-->(m:Member)
MATCH (s:Sales)-->(me:Menu)
WHERE s.order_date > m.join_date
RETURN s.customer_id AS Customer,
count(s.product_id) as Total_item,
sum(me.price) as Amount_spent
```

Table | Text | Code

	Customer	Total_item	Amount_spent
1	"A"	3	36.0
2	"B"	3	34.0

SQL V/s Graph Databases

Question 6: **A query to show the number of days each customer visited the resturant.**

SQL

```
3 •  SELECT
4    Customer_id AS Customers,
5    count(Distinct(Day(order_date))) As 'Number of days'
6    From sales
7    Group by 1;
8
```

100% 1:1

Result Grid | Filter Rows: Search | Export:

Customers	Number of da...
A	4
B	5
C	2

Cypher Query

```
1 // Number of days each customer visited the resturant
2 match(s:Sales)
3 Return s.customer_id as customer, count(DISTINCT date(s.order_date))as
  Number_of_days
```

Table | Text | Code

	customer	Number_of_days
1	"A"	4
2	"B"	6
3	"C"	2

SQL V/s Graph Databases

Question 6: **A query to show the details of the products bought on the first day.**

SQL

```
3 •  SELECT
4        s.order_date as Date,
5        m.product_name AS Products,
6        COUNT(s.product_id) AS Quantity,
7        SUM(m.price) AS Revenue
8    FROM sales AS s
9    JOIN menu AS m ON s.product_id = m.product_id
10   WHERE s.order_date = '2021-01-01'
11   GROUP BY m.product_name;
12
```

100% 25:11

Result Grid | Filter Rows: Search | Export:

Date	Products	Quantity	Revenue
2021-01-01	sushi	1	10
2021-01-01	curry	2	30
2021-01-01	ramen	2	24

Cypher Query

```
1 // Number of product bought on the first day
2 MATCH(s:Sales)⟶ (me:Menu)
3 Where s.order_date = '2021-01-01'
4 Return s.order_date as Date, me.product_name as product,COUNT(s.product_id) AS
  Quantity, sum(me.price) as Revenue
```

	Date	product	Quantity	Revenue
1	"2021-01-01"	"sushi"	1	10.0
2	"2021-01-01"	"curry"	2	30.0
3	"2021-01-01"	"ramen"	2	24.0

Conclusion

This report summarized the successful construction of a data warehouse. It detailed the process of extracting, transforming, and loading relevant data to provide valuable insights for stakeholders. Additionally, the report presented an analytical report with visuals to aid informed decision-making. Finally, it compared querying methods between structured relational databases and graph databases.