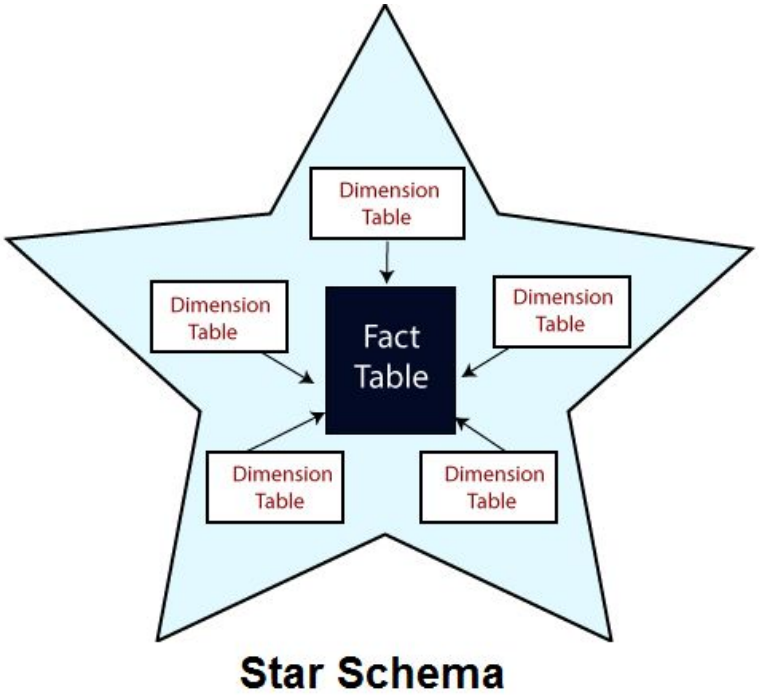
OLAP Data Warehouse with Star Schema

MoSIG M1 Database. PhD. Bahareh Afshinpour
Students: Phan Manh Tung, Louis Choules

Introduction

- The world of business is becoming increasingly data-driven, and for good reason. Data provides critical insights to make better decisions, understand customers' behaviours, and optimize business operations.
- Businesses leveraging the power of data to make informed decisions, will gain a competitive edge and identify opportunities for growth.
- Therefore, businesses turn to OLAP Data Warehouse projects. Unlike traditional normalized database, it provides faster query performance, enabling users to quickly retrieve data and gain insights.

Structure: Star Schema



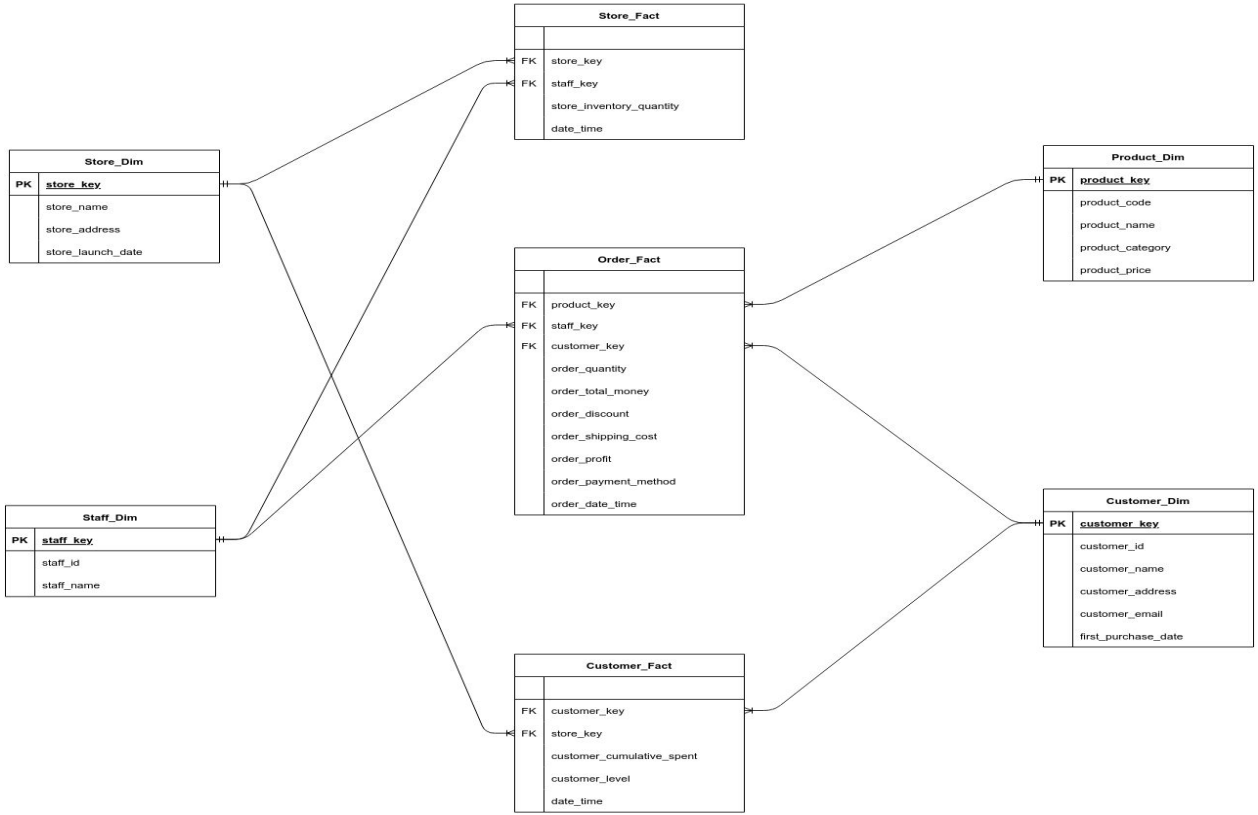
Principles

Fact tables	Dimensions tables
Numeric data, or measures, such as sales revenue, quantity sold, or profit margin.	Descriptive attributes: characteristics of a business entity, such as product, customer, or time.
Only foreign keys that links to dimension tables.	A primary key that uniquely identifies each dimension record.
Has a "grain": the level of detail or granularity at which its data is captured and stored. Atomic level	Do not has a “grain”, as it contains descriptive attributes, not numerical
Optimized for query performance, including appropriate indexing and partitioning strategies.	Denormalized to reduce the number of joins required for queries. Redundant information is added.

Denormalized vs. Normalized

Normalized	Denormalized
Minimize data redundancy, maintaining data integrity.	Intentionally add redundant data to a database schema.
Ensures data consistency and reduces anomalies that can arise from data updates, deletions, and insertions.	Improve query performance, simplify data retrieval, or achieve other performance-related goals.
Smaller tables that are more efficient to update.	Larger tables that are less efficient to update.
Requires more joins to retrieve data.	Requires less joins to retrieve data.
Increases query complexity and performance.	Simplifies queries and improves query performance.
More flexible and adaptable to changes.	Less flexible and less adaptable to changes.

Designed Schema



Import data

- Kaggle E-Commerce dataset for the order_fact, real name, locations for the descriptive attributes in staff_dim, customer_dim and store_dim.
- Data manipulation with Python pandas, then export to 7 .csv files.
- Feed these .csv files into the database: PHP SQL database and Sqlite3 my_data.db.

	order_fact	store_fact	customer_fact	customer_dim	store_dim	staff_dim	product_dim
# rows	51290	10543	51233	38997	298	8764	42

Order Fact

	product_key	staff_key	customer_key	order_quantity	order_total_money	order_discount	order_shipping_cost	order_profit	order_payment_method	order_date_time
0	0	83	108	1.0	102.6	0.3	4.6	46.0	credit_card	2018-01-02 10:56:33
1	1	8265	21853	1.0	158.9	0.3	11.2	112.0	credit_card	2018-07-24 20:41:37
2	2	1120	33714	5.0	529.6	0.1	3.1	31.2	credit_card	2018-11-08 08:38:49
3	3	8358	8606	1.0	85.2	0.3	2.6	26.2	credit_card	2018-04-18 19:28:06
4	4	6128	24314	1.0	191.0	0.3	16.0	160.0	credit_card	2018-08-13 21:18:39
...	...	...	...	...	...	...	...	...	...	...
51285	39	7728	4479	4.0	349.1	0.3	1.9	19.2	money_order	2018-02-28 22:59:50
51286	40	3933	4467	5.0	281.4	0.2	1.4	14.0	credit_card	2018-02-28 13:19:25
51287	41	4022	4489	1.0	97.1	0.3	4.0	39.7	credit_card	2018-02-28 10:25:07
51288	32	6522	4475	1.0	186.0	0.2	13.2	131.7	credit_card	2018-02-28 10:50:08
51289	33	3766	4586	5.0	748.4	0.3	9.9	99.4	credit_card	2018-02-28 11:09:40

51290 rows × 10 columns

What product most sell throughout the year: TOP 10?

```
import sqlite3
import pandas as pd
con =
sqlite3.Connection('/content/gdrive/MyDrive/database_project/my_(

# What product most sell throughout the year: TOP 10
query = """
SELECT product_name, product_category, SUM(order_quantity) AS
total_sales_quantity
FROM product_dim A
JOIN order_fact B ON A.product_key=B.product_key
GROUP BY product_name
ORDER BY total_sales_quantity DESC
LIMIT 10;
"""
observations = pd.read_sql(query, con)
observations
```

	product_name	product_category	total_sales_quantity
0	Titak watch	Fashion	6254.0
1	Formal Shoes	Fashion	6154.0
2	Sports Wear	Fashion	6093.0
3	Running Shoes	Fashion	6064.0
4	Fossil Watch	Fashion	6050.0
5	Sneakers	Fashion	6049.0
6	Casula Shoes	Fashion	6035.0
7	Shirts	Fashion	6012.0
8	Suits	Fashion	5996.0
9	T - Shirts	Fashion	5986.0

Most sold products - TOP 3 for each category

```
query = """
SELECT product_name, product_category, total_sales_quantity
FROM (SELECT product_name, product_category, total_sales_quantity, RANK()
      OVER (PARTITION BY product_category ORDER BY total_sales_quantity DESC)
AS rank

      FROM (SELECT product_name, product_category, SUM(order_quantity)
AS total_sales_quantity
            FROM product_dim A
            JOIN order_fact B ON A.product_key=B.product_key
            GROUP BY product_name, product_category)

      GROUP BY product_name, product_category)

WHERE rank <= 3
"""
observations = pd.read_sql(query, con)
observations
```

	product_name	product_category	total_sales_quantity
0	Car Body Covers	Auto & Accessories	2040.0
1	Tyre	Auto & Accessories	2023.0
2	Car Pillow & Neck Rest	Auto & Accessories	2013.0
3	Speakers	Electronic	581.0
4	Fans	Electronic	523.0
5	Samsung Mobile	Electronic	501.0
6	Titak watch	Fashion	6254.0
7	Formal Shoes	Fashion	6154.0
8	Sports Wear	Fashion	6093.0
9	Beds	Home & Furniture	3908.0
10	Dinning Tables	Home & Furniture	3874.0
11	Sofa Covers	Home & Furniture	3852.0

Best sellers of March : Highest total sales

```
import pandas as pd
# 1> Highest order total money

query = """
SELECT staff_name, staff_id, SUM(order_total_money) AS
total_sales_money
FROM staff_dim A
JOIN order_fact B ON A.staff_key=B.staff_key
WHERE date(B.order_date_time) BETWEEN '2018-03-01'
AND '2018-03-31'
GROUP BY staff_id
ORDER BY total_sales_money DESC
LIMIT 10;
"""
observations = pd.read_sql(query, con)
observations
```

	staff_name	staff_id	total_sales_money
0	Paul Crowley	41347	1971.6
1	Denia Caballero	43678	1648.2
2	Marius Sabaliauskas	39982	1590.7
3	Maura Hopkins	44102	1533.9
4	Simon Eldershaw	43975	1490.5
5	Wolfgang Larrazábal	43822	1430.3
6	Dejan Drakul	43004	1422.2
7	Nele Jansegers	44445	1356.7
8	John Dooley	39797	1355.2
9	John Harris	43468	1253.7

Stores with lowest inventory, at which specific date

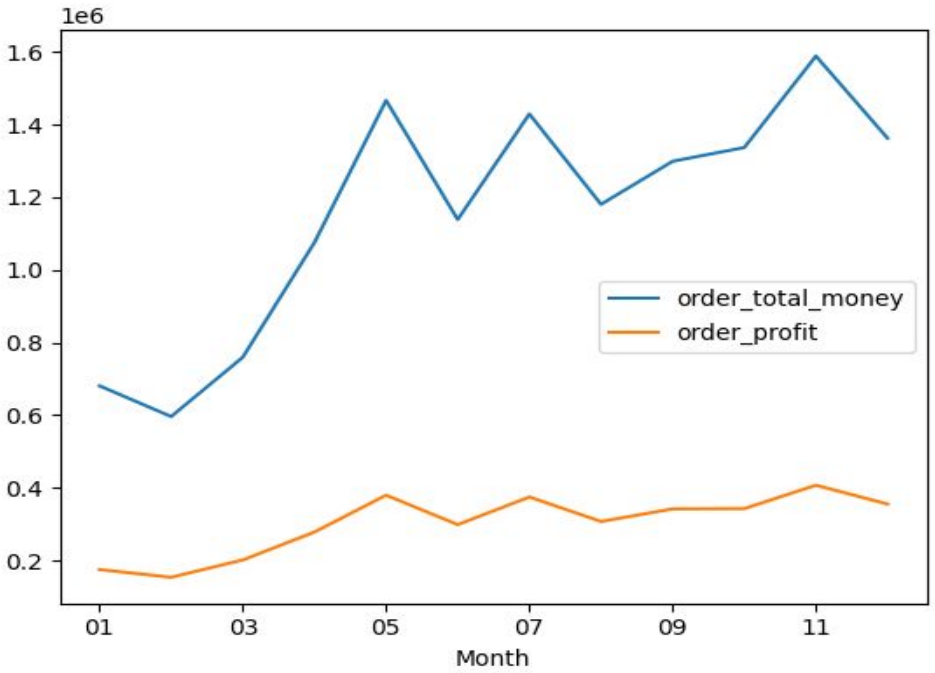
```
query = """
SELECT store_name, store_address,
MAX(store_inventory_quantity) AS lowest_monthly_inventory,
date(date_time) AS date
FROM store_dim A
JOIN store_fact B ON A.store_key=B.store_key
WHERE date BETWEEN '2018-01-01' AND
'2018-12-31'
GROUP BY store_name
ORDER BY lowest_monthly_inventory ASC
LIMIT 10;
"""
observations = pd.read_sql(query, con)
observations
```

	store_name	store_address	lowest_monthly_inventory	date
0	TOMPKINSVILLE	TOMPKINSVILLE, KY, USA	1400	2018-07-01
1	BOULDER CITY	BOULDER CITY, NV, USA	1500	2018-01-01
2	CREVE COEUR	CREVE COEUR, MO, USA	1600	2018-05-01
3	EAU CLAIRE	EAU CLAIRE, WI, USA	1600	2018-05-01
4	GREENCASTLE	GREENCASTLE, IN, USA	1600	2018-01-01
5	HILLSBORO	HILLSBORO, OH, USA	1600	2018-04-01
6	LYNN	LYNN, MA, USA	1600	2018-02-01
7	MAHNOMEN	MAHNOMEN, MN, USA	1600	2018-02-01
8	SUN CITY WEST	SUN CITY WEST, AZ, USA	1600	2018-05-01
9	ALPINE	ALPINE, TX, USA	1700	2018-03-01

Total sales and profit per month

```
query = """
SELECT strftime('%m', order_date_time) as Month,
SUM(order_total_money) AS order_total_money,
SUM(order_profit) AS order_profit
FROM order_fact
GROUP BY Month;
"""
observations = pd.read_sql(query, con)

observations.plot(x="Month",
y=['order_total_money', 'order_profit'])
```



Advantages vs Disadvantages

Advantages	Disadvantages
Fast query response time	Complex technology
Flexible data analysis	Data quality issues
Aggregated views	Cost
Enhanced reporting capabilities	Limited transactional capabilities
Scalability	Data security and privacy concerns

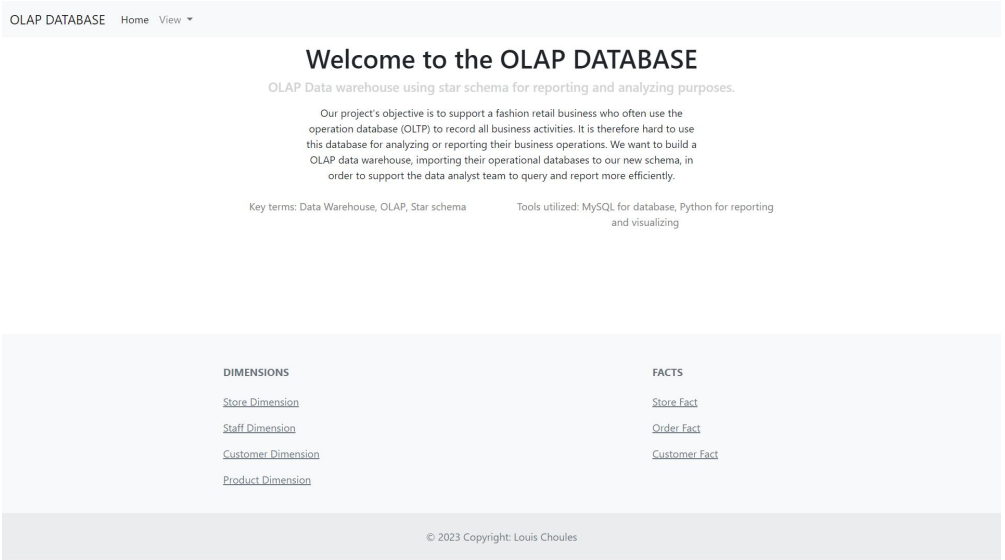
Website

Basically the design of the website divided into 3 part

- Dashboard
- Dimension Page
  - View
  - Insert
  - Update
  - Delete
- Fact Page
  - View

Website - Dashboard

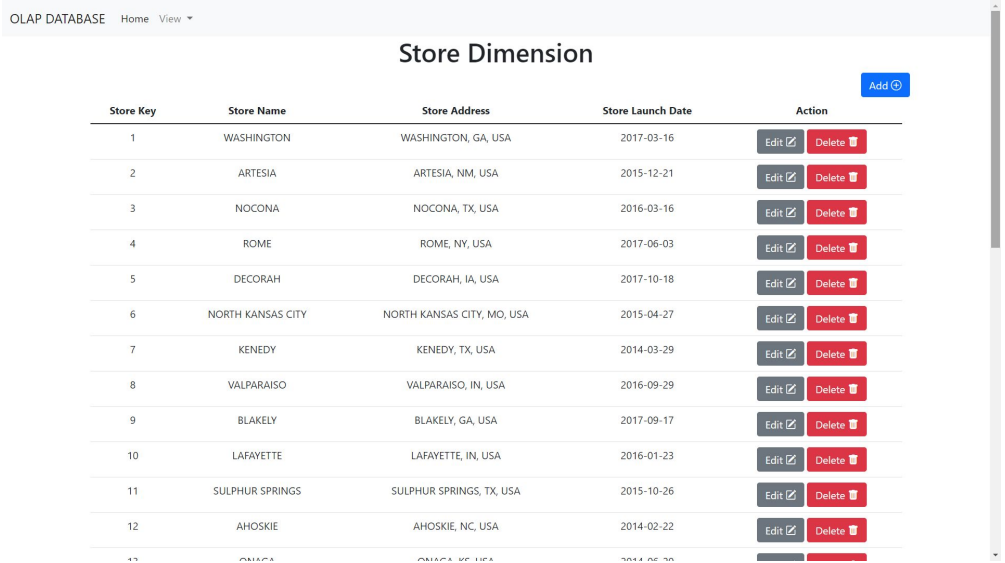
It will display a brief information about the website.



Website - Dimension Page

View

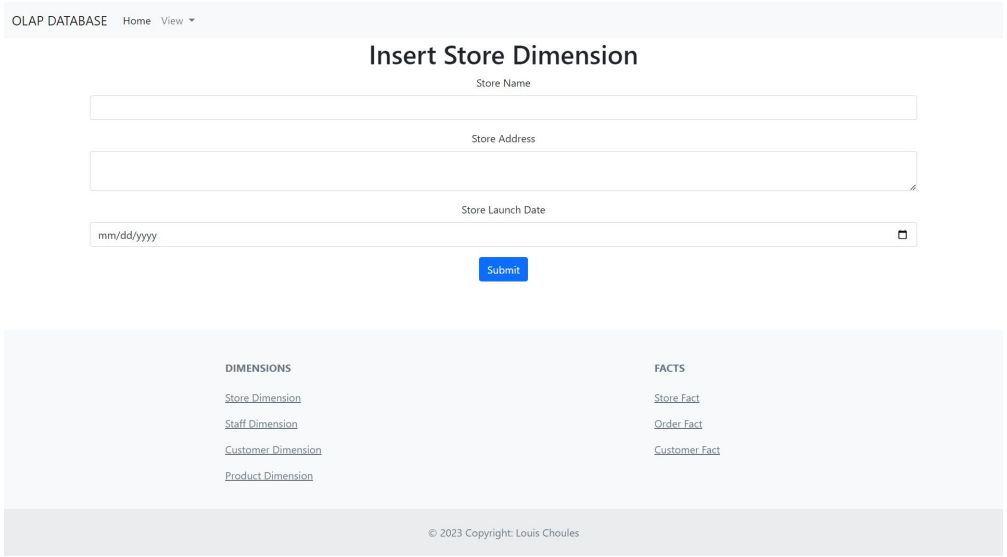
It will display the information in the chosen table and paginate the data into 25 row per page.



Website - Dimension Page

Insert

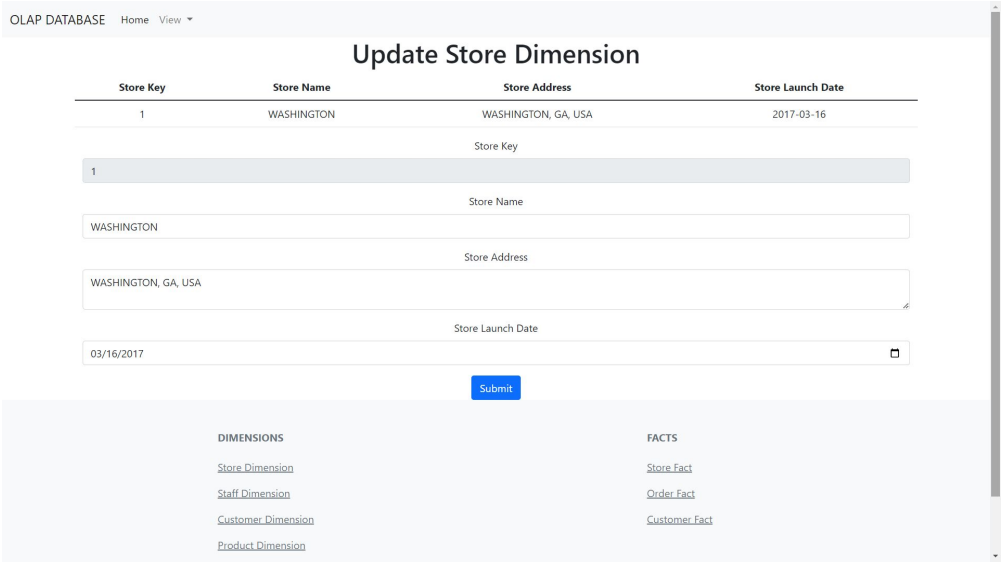
It will display a form to fill related to the dimension that you choose.



Website - Dimension Page

Update

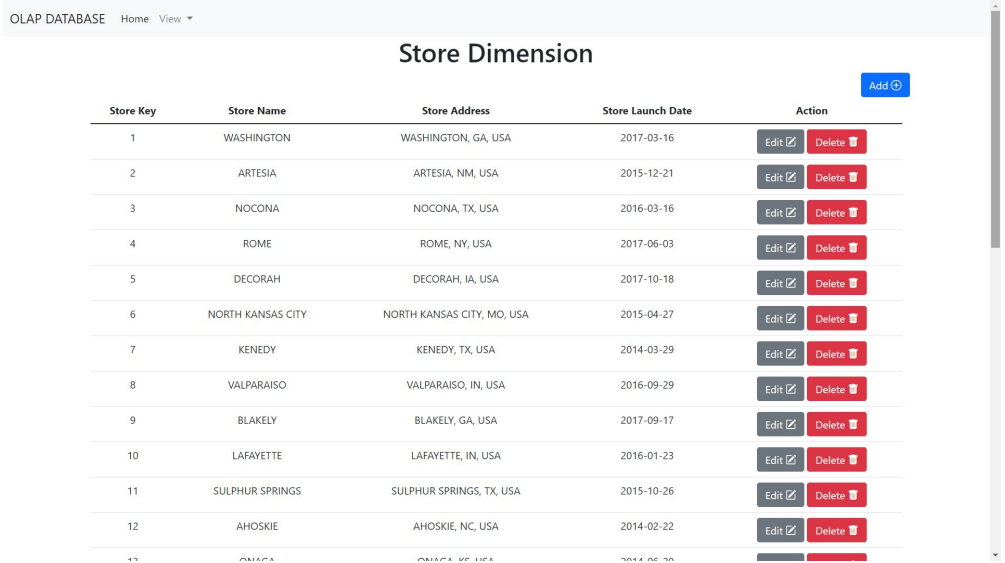
It will display the old information and a form to update the data related to the specific data that you choose.



Website - Dimension Page

Delete

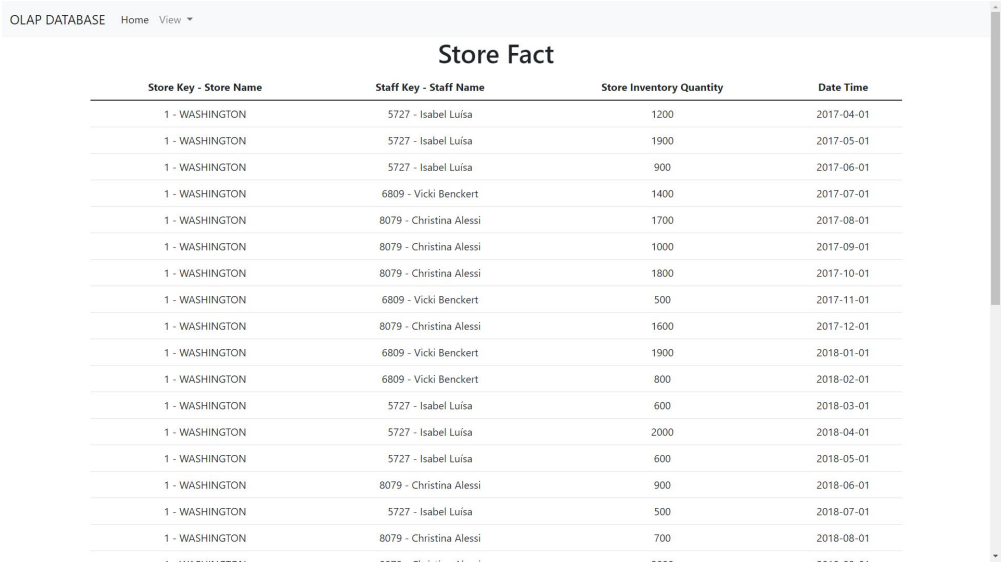
It will display a confirmation modal that can be used to delete the selected data.



Website - Fact Page

View

It will display the information in the chosen table and paginate the data into 25 row per page.



Demo!

Thank you!