Lab Guide: Market Basket Analysis with DuckDB UI

1. Objective

In this lab you will:
- Work with the DuckDB UI (the downloadable version with persistent storage).
- Create and manage a persistent DuckDB database file.
- Convert CSV datasets directly into Parquet format (optimized storage).
- Create tables based on Parquet files only.
- Explore customer purchasing patterns.
- Perform business-oriented analyses (customer behavior, product demand, cross-selling).
- Export query results.

2. Setup

1. Download DuckDB UI:
   - Install DuckDB ≥ v1.4 from http s://duckdb.org/.
   - Run in your terminal:
     ```
     duckdb -ui
     ```
     or inside an open session:
     ```
     CALL start_ui();
     ```
   - This will open a web-based UI in your browser (usually http://localhost:4213).

2. Create a persistent database file (command line or UI):
   ```
   duckdb instacart.duckdb -ui
   ```
   This ensures all changes are saved in instacart.duckdb.

3. Download the dataset Instacart Market Basket Analysis and unzip the CSVs:
`orders.csv, products.csv, order_products__prior.csv, order_products__train.csv, aisles.csv, departments.csv.`

3. Convert CSVs Directly to Parquet

Instead of creating tables from CSV first, we directly convert each dataset into Parquet.

```
COPY (
  SELECT * FROM read_csv_auto('C:/Projects/Instacart/CSV/orders.csv')
) TO 'C:/Projects/Instacart/Parquet/orders.parquet' (FORMAT 'parquet');
```

```
COPY (
  SELECT * FROM read_csv_auto('C:/Projects/Instacart/CSV/products.csv')
) TO 'C:/Projects/Instacart/Parquet/products.parquet' (FORMAT 'parquet');


COPY (
  SELECT * FROM
read_csv_auto('C:/Projects/Instacart/CSV/order_products__prior.csv')
) TO 'C:/Projects/Instacart/Parquet/order_products_prior.parquet' (FORMAT
'parquet');


COPY (
  SELECT * FROM
read_csv_auto('C:/Projects/Instacart/CSV/order_products__train.csv')
) TO 'C:/Projects/Instacart/Parquet/order_products_train.parquet' (FORMAT
'parquet');


COPY (
  SELECT * FROM read_csv_auto('C:/Projects/Instacart/CSV/aisles.csv')
) TO 'C:/Projects/Instacart/Parquet/aisles.parquet' (FORMAT 'parquet');


COPY (
  SELECT * FROM read_csv_auto('C:/Projects/Instacart/CSV/departments.csv')
) TO 'C:/Projects/Instacart/Parquet/departments.parquet' (FORMAT 'parquet');
```

4. Create Tables from Parquet Files

Now that all datasets are stored in Parquet, we create tables directly from them.

```
CREATE OR REPLACE TABLE orders AS
SELECT * FROM 'C:/Projects/Instacart/Parquet/orders.parquet';


CREATE OR REPLACE TABLE products AS
SELECT * FROM 'C:/Projects/Instacart/Parquet/products.parquet';
```

```
CREATE OR REPLACE TABLE order_products_prior AS

SELECT * FROM 'C:/Projects/Instacart/Parquet/order_products_prior.parquet';


CREATE OR REPLACE TABLE order_products_train AS

SELECT * FROM 'C:/Projects/Instacart/Parquet/order_products_train.parquet';


CREATE OR REPLACE TABLE aisles AS

SELECT * FROM 'C:/Projects/Instacart/Parquet/aisles.parquet';


CREATE OR REPLACE TABLE departments AS

SELECT * FROM 'C:/Projects/Instacart/Parquet/departments.parquet';
```

5. First Exploration

```
SELECT COUNT(*) FROM orders;
SELECT * FROM products LIMIT 10;
```

6. Exercises

Exercise 1 – Most Popular Products

```
SELECT p.product_name, COUNT(*) AS times_ordered
FROM order_products_prior opp
JOIN products p ON opp.product_id = p.product_id
GROUP BY p.product_name
ORDER BY times_ordered DESC
LIMIT 10;
```

Exercise 2 – Reorder Behavior

```
SELECT CASE WHEN reordered = 1 THEN 'Reordered' ELSE 'First Time' END AS
order_type,
       COUNT(*) AS count_orders
FROM order_products_prior
GROUP BY order_type;
```

Exercise 3 – Aisle and Department Insights

```
SELECT d.department, COUNT(*) AS num_products
FROM products p
JOIN departments d ON p.department_id = d.department_id
GROUP BY d.department
ORDER BY num_products DESC;
```

Exercise 4 – Basket Size Distribution

```
SELECT order_id, COUNT(*) AS basket_size
```

```
FROM order_products_prior
GROUP BY order_id
ORDER BY basket_size DESC
LIMIT 10;
```

Exercise 5 – Frequently Bought Together

```
SELECT p1.product_name AS product_A,
       p2.product_name AS product_B,
       COUNT(*) AS times_bought_together
FROM order_products_prior o1
JOIN order_products_prior o2
     ON o1.order_id = o2.order_id AND o1.product_id < o2.product_id
JOIN products p1 ON o1.product_id = p1.product_id
JOIN products p2 ON o2.product_id = p2.product_id
GROUP BY product_A, product_B
ORDER BY times_bought_together DESC
LIMIT 10;
```

Exercise 6 – Customer Behavior Over Time

```
SELECT order_number, COUNT(*) AS num_orders
FROM orders
GROUP BY order_number
ORDER BY order_number
LIMIT 20;
```

7. Updating Tables

```
ALTER TABLE orders ADD COLUMN order_type STRING;

UPDATE orders
SET order_type = CASE
    WHEN order_number = 1 THEN 'First Order'
    ELSE 'Repeat'
END;
```

8. Exporting Results

Export to CSV:

```
COPY (
  SELECT p.product_name, COUNT(*) AS times_ordered
  FROM order_products_prior opp
  JOIN products p ON opp.product_id = p.product_id
  GROUP BY p.product_name
  ORDER BY times_ordered DESC
  LIMIT 20
) TO 'C:/Projects/Instacart/Results/top_products.csv' (HEADER, DELIMITER ',');
```

Export to Parquet:

```
COPY (
  SELECT * FROM orders LIMIT 1000
) TO 'C:/Projects/Instacart/Results/orders_sample.parquet' (FORMAT PARQUET);
```

9. Instacart Dataset Background

The Instacart Online Grocery Shopping Dataset is a large-scale, anonymized collection of grocery orders made through Instacart, a grocery delivery service in the United States. It was originally released for a Kaggle competition focused on predicting customer purchase behavior.

The dataset contains millions of records and provides detailed information on users' grocery orders, the products they purchased, and the organizational structure of the catalog (aisles and departments). Because of its scale and organization, it is widely used in teaching and research for SQL practice, exploratory data analysis, consumer behavior studies, and machine learning tasks such as recommender systems.

Tables in the dataset

- **orders**
  Contains metadata about each order: order ID, user ID, order number, day of week, time of day, and days since prior order.

- **products**
  A catalog of all products, including product ID, name, aisle ID, and department ID.

- **aisles**
  Lookup table mapping aisle IDs to descriptive aisle names (e.g., "fresh vegetables", "frozen foods").

- **departments**
  Lookup table mapping department IDs to descriptive department names (e.g., "produce", "beverages").

- **order_products_prior**
  Contains the products associated with customers' *previous* orders (used for training in the competition). Includes order ID, product ID, add-to-cart order, and a flag for whether the item was reordered.

- **order_products_train**
  Contains the products associated with customers' *training set* orders (used for evaluation in the competition). Same structure as order_products_prior.