Indexes in the Data Warehousing Project

Project team members:

1- **Yassin Gamal Elhussiny**
2- **Mina Samir Atallah**
3- **Youssef Mohamed Khirallah**
4- **George Ayad Naguib**

1. Primary Key Indexes:

Purpose: These indexes are automatically created for primary key constraints in all dimension and fact tables. They ensure the uniqueness of rows and improve performance for queries that rely on these keys for joins and filtering.

Examples:

1. Flight_ID in Flights_Dimension
2. Fact_ID in Flight_Facts, Reservation_Facts, and Customer_Care_Facts
3. Customer_ID in Customers_Dimension
4. Date_ID in Dates_Dimension

Usage: These indexes enable quick lookups and efficient join operations, which are common in star schema designs.

2. Foreign Key Indexes:

Purpose: These indexes correspond to foreign keys that link fact tables to dimension tables, enhancing join performance between these tables.

Examples:

1. Flight_ID in Flight_Facts and Reservation_Facts
2. Channel_ID in Reservation_Facts
3. Customer_ID in Customer_Care_Facts
4. Interaction_ID in Customer_Care_Facts

Usage: Foreign key indexes optimize the execution of star-join queries by reducing the cost of joining large fact tables with smaller dimension tables.

3. Clustered Indexes:

Purpose: Clustered indexes physically sort the data in the table according to the index key.

Examples:

1. Date_ID in Dates_Dimension
2. Flight_ID in Flights_Dimension

Usage: Clustered indexes facilitate efficient data retrieval when the data is queried in the order of the index, such as fetching all flights from a specific airline or all entries for a particular date.

4. Non-Clustered Indexes:

Purpose: These indexes are created on frequently queried columns that are not part of the primary key, providing additional performance optimization for specific queries.

Examples:

1. Airline, Origin, and Destination in Flights_Dimension
2. Revenue and Number_Of_Flights in Flight_Facts
3. Profit in Reservation_Facts

Usage: Non-clustered indexes improve query performance for aggregations, filtering, and sorting on non-key columns.

5. Composite Indexes:

Purpose: Composite indexes are created on multiple columns and used in queries involving multi-column filters or joins.

Examples:

1. A composite index on Year, Month, and Day in Dates_Dimension
2. A composite index on Channel_ID and Flight_ID in Reservation_Facts

Usage: These indexes optimize queries requiring filtering or grouping by multiple columns, such as fetching reservations for a specific channel on a specific flight.