**Batch -** T7

**Assignment No. -** 6

**Student Name -** Sharaneshwar Bharat Punjal

**Student PRN -** 23520011

To design and implement a data warehouse for a customer order processing system in a company. [Use MySQL Database]

**Specifications:**

The target of data warehouse system is an enterprise that consists of a number of stores located in different cities and states. Each store holds a variety of items in various quantity. In addition, the enterprise keeps the information of the customers. There are two kinds of customers: walk-in led by tourism guide and mail-order by post address inclusive. The city location of the customer, together with the data of the customer's first order, is stored by the existing system. Each customer lives in one city only, and the enterprise will try to satisfy the customer's order items by the present stock in the city where the customer lives. Each customer order can be for any quantity of any number of items, and each order is uniquely identified by an order number.

The location of the stores is also recorded. Each store is located in one city, and there can be many stores in the city. Each city has one headquarter for coordinating all of its stores. The enterprise's goal is to meet all of the customer's requirements from stores located in the customer's city. If the requirement cannot be met, the company will turn to the other cities where the item can be found if there is any.

Some processing information is important for the enterprise. For example, the total quantity of item stored in each city. After every time an item is taken, the company needs to know the total quantities of the item in all the stores in a city.

The relational schema of the enterprise's current (operational) databases are:

**Headquarter Database;**

Relation Customer (Customer id. Customer name, City\_id, First\_order\_date)

Relation Walk-in\_customers (\*Customer id, tourism\_guide, Time)

Relation Mail\_order\_customers (Customer id, post\_address, Time)

**Sales Databases:**

Relation Headqarters (City id. City name, Headquarter\_addr, State, Time)

Relation Stores (Store id. "City\_id, Phone, Time)

Relation Items (Item id. Description, Size, Weight, Unit\_price, Time)

Relation Stored\_items (\*Store id. Item id. Qantity held. Time)

Relation Order (Order no, Order\_date, Customer\_id)

Relation Ordered\_item (Order no. Item id. Quantity\_ordered, Ordered\_price, Time)

Where underlined are primary key and "\*" prefixed are foreign keys.

**Business requirements:**

In order to meet users' demand, the data warehouse system extracts data from the existing two database into a data warehouse, and provides online analytical processing with roll up, drill down, slice and dice features according to users’ selections based on dimension tables to meet the user requirements.

Build data warehouse / OLAP which will answer the following queries:

1. Find all the stores along with city, state, phone, description, size, weight and unit price that hold a particular item of stock.

2. Find all the orders along with customer name and order date that can be fulfilled by a given store.

3. Find all stores along with city name and phone that hold items ordered by given customer.

4. Find the headquarter address along with city and state of all stores that hold stocks of an item above a particular level.

5. For each customer order, show the items ordered along with description, store id and city name and the stores that hold the items.

6. Find the city and the state in which a given customer lives.

7. Find the stock level of a particular item in all stores in a particular city.

8. Find the items, quantity ordered, customer, store and city of an order.

9. Find the walk-in customers, mail order customers and dual customers (both walk-in and mail order).

**Prepare report as follows:**

1. Introduction-objective and scope of the project

2. Business requirement application specification of the data warehousing for the users.

3. Functional specification input and output specification of the data warehousing

4. Data Warehousing Design - stepwise procedure methodology of designing the data warehousing including star schema.

5. Data cube implementation computer automation of implementing the data warehousing loading data into data cubes.

6. Observations:

a. Online analytical processing reports - invoke commands or panels to generate OLAP reports.

b. Data verification verify the OLAP reports source relational tables' data.

7. Conclusion.

**1. Introduction**

**Objective:**  
The aim of this project is to design and implement a data warehouse for an enterprise customer order processing system. The data warehouse will consolidate data from existing operational databases (Headquarter and Sales databases) that cover customer, order, store, item, and inventory information. Using MySQL as the database platform, the system will support online analytical processing (OLAP) features—including roll-up, drill-down, slice, and dice—to support ad hoc queries and comprehensive business reporting.

**Scope:**

* **Enterprise Overview:** The enterprise consists of multiple stores in various cities and states. Each store holds a variety of items in specified quantities. The customer base includes both walk-in customers (assisted by a tourism guide) and mail-order customers (identified by post addresses).
* **Data Sources:** The design will integrate data from the operational Headquarter and Sales databases.
* **Business Processes:** The system will enable the enterprise to match customer orders with available local stock and, if necessary, source items from other cities.
* **Reporting Needs:** The system will answer complex queries regarding inventory, orders, and customer details, and will generate OLAP reports to support decision-making.

**2. Business Requirement Application Specification**

The data warehouse is designed to meet the following business requirements:

* **Customer-Centric Fulfillment:**
  + Match customer orders with store inventories based on the customer’s city.
  + If the local store does not hold the item, search for stock in other cities.
* **Inventory Management:**
  + Track the total quantity of each item held across all stores in a city.
  + Update inventory levels in real time after every transaction.
* **OLAP Reporting:**
  + Provide multi-dimensional analysis using roll-up (aggregating data by city, state, etc.), drill-down (examining detailed data at store or order level), slice (filtering on one dimension such as item type), and dice (selecting multiple dimensions simultaneously).
* **Query Requirements:**  
  The data warehouse must be able to answer the following queries:
  + **Item Stock Query:** Find all stores along with city, state, phone, item description, size, weight, and unit price that hold a particular item.
  + **Store Order Fulfillment Query:** List orders along with customer names and order dates that can be fulfilled by a given store.
  + **Customer Ordered Items Query:** Identify all stores (with city name and phone) that hold items ordered by a given customer.
  + **High Stock Level Query:** List headquarter addresses along with city and state for all stores that hold stocks of an item above a given threshold.
  + **Detailed Order Query:** For each customer order, show the items ordered (with description), the store id, and the city name of the store holding the items.
  + **Customer Location Query:** Determine the city and state in which a given customer lives.
  + **City Stock Level Query:** Find the stock level of a particular item in all stores within a specified city.
  + **Order Details Query:** Identify the items, quantity ordered, customer, store, and city for a specific order.
  + **Customer Type Query:** Classify customers as walk-in, mail order, or dual customers (those who have placed both types of orders).

**3. Functional Specification**

This section outlines the inputs, processing, and outputs of the data warehousing system.

**Input Specification**

* **Source Systems:**
  + **Headquarter Database:**
    - *Customer* (Customer\_id, Customer\_name, City\_id, First\_order\_date)
    - *Walk-in\_customers* (Customer\_id, tourism\_guide, Time)
    - *Mail\_order\_customers* (Customer\_id, post\_address, Time)
  + **Sales Databases:**
    - *Headquarters* (City\_id, City\_name, Headquarter\_addr, State, Time)
    - *Stores* (Store\_id, City\_id, Phone, Time)
    - *Items* (Item\_id, Description, Size, Weight, Unit\_price, Time)
    - *Stored\_items* (Store\_id, Item\_id, Quantity\_held, Time)
    - *Order* (Order\_no, Order\_date, Customer\_id)
    - *Ordered\_item* (Order\_no, Item\_id, Quantity\_ordered, Ordered\_price, Time)
* **ETL Process:**
  + **Extract:** Data is extracted periodically from the operational databases.
  + **Transform:** Data is cleaned, integrated, and aggregated. This may include:
    - Converting different date formats.
    - Merging customer types into a unified customer dimension.
    - Aggregating inventory data by store and city.
  + **Load:** Transformed data is loaded into the data warehouse tables (fact and dimension tables).

**Output Specification**

* **OLAP Reports:**
  + Multi-dimensional reports that allow end-users to view data aggregated by store, city, state, or product.
  + Interactive queries for ad hoc analysis (using SQL GROUP BY, ROLLUP, CUBE functions, etc.).
  + Standardized output reports such as inventory summaries, order histories, and customer segmentation reports.
* **User Interface:**
  + Command panels or web-based dashboards that enable the user to select dimensions and measures (e.g., item descriptions, quantities, dates) and to generate reports on demand.

**4. Data Warehousing Design**

This section describes the stepwise methodology and star schema design.

**Stepwise Procedure & Methodology**

1. **Requirements Analysis:**
   * Gather and analyze business requirements and user queries.
   * Identify key performance indicators (KPIs) such as order fulfillment rate, inventory levels, and customer order frequency.
2. **Identify Facts and Dimensions:**
   * **Fact Tables:**
     + *Fact\_Orders:* Contains measures such as quantity ordered, ordered price, and serves as a central fact table for orders.
     + *Fact\_Inventory:* Stores aggregated inventory data per store or city.
   * **Dimension Tables:**
     + *Dim\_Customer:* Customer id, name, type (walk-in/mail order), city, first order date.
     + *Dim\_Store:* Store id, phone, and associated city id.
     + *Dim\_Item:* Item id, description, size, weight, unit price.
     + *Dim\_City:* City id, city name, state.
     + *Dim\_Headquarter:* Headquarter details (city id, address, state).
3. **Design the Star Schema:**
   * **Fact\_Orders** (Order\_no, Order\_date, Customer\_id, Store\_id, Item\_id, Quantity\_ordered, Ordered\_price)  
     → Foreign keys link to:
     + Dim\_Customer (Customer\_id)
     + Dim\_Store (Store\_id)
     + Dim\_Item (Item\_id)
     + Dim\_Date (derived from Order\_date, if a date dimension is implemented)
   * **Fact\_Inventory** (Store\_id, Item\_id, Quantity\_held, Last\_updated)  
     → Foreign keys link to:
     + Dim\_Store (Store\_id)
     + Dim\_Item (Item\_id)
   * **Dimension Tables:**
     + *Dim\_Customer, Dim\_Store, Dim\_Item, Dim\_City, Dim\_Headquarter*  
       These dimensions are linked through the fact tables to provide a full context for each measure.
4. **ETL Implementation:**
   * Use MySQL scripts or a dedicated ETL tool to extract data from the source relational databases.
   * Cleanse and transform the data as needed.
   * Load the transformed data into the designed star schema tables in the data warehouse.

**5. Data Cube Implementation and Automation**

**Data Cube Implementation**

* **Definition:**  
  Data cubes are multi-dimensional arrays of values, derived from the fact tables, that allow fast analysis across different dimensions (such as city, store, item, and time).
* **Process:**
  + **Aggregation:**  
    Using MySQL’s grouping and aggregation functions (e.g., GROUP BY, ROLLUP, CUBE), data can be aggregated along various dimensions.
  + **Cube Building:**  
    Create materialized views or summary tables that pre-calculate common aggregations. For example:

CREATE TABLE Inventory\_Cube AS

SELECT s.City\_id, s.Store\_id, i.Item\_id,

SUM(si.Quantity\_held) AS Total\_Quantity

FROM Stored\_items si

JOIN Stores s ON si.Store\_id = s.Store\_id

JOIN Items i ON si.Item\_id = i.Item\_id

GROUP BY s.City\_id, s.Store\_id, i.Item\_id;

* + **Automation:**
    - Schedule ETL jobs and cube refresh operations using MySQL events or an external scheduler.
    - Develop stored procedures to update aggregated data on-demand or at set intervals.

**OLAP Operations**

* **Roll-Up & Drill-Down:**
  + Roll-up: Aggregate data from store level to city level.

SELECT c.City\_name, SUM(Total\_Quantity) AS City\_Total

FROM Inventory\_Cube ic

JOIN Stores s ON ic.Store\_id = s.Store\_id

JOIN Dim\_City c ON s.City\_id = c.City\_id

GROUP BY c.City\_name;

* + Drill-down: Query at the detailed level by adding further filters.
* **Slice and Dice:**
  + Slice: Extract a subset of the cube (e.g., view inventory for a specific item).

SELECT \* FROM Inventory\_Cube

WHERE Item\_id = 'ITEM123';

* + Dice: Select multiple dimensions (e.g., inventory of an item in a specific city and store).

**6. Observations**

**a. OLAP Reports Generation**

* **Interactive Reports:**  
  The system supports dynamic querying via OLAP commands. Users can invoke panels that allow selection of dimensions and measures.
* **Sample Commands:**
  + **Query 1 (Item Stock):**

SELECT s.Store\_id, c.City\_name, c.State, s.Phone,

i.Description, i.Size, i.Weight, i.Unit\_price

FROM Stored\_items si

JOIN Items i ON si.Item\_id = i.Item\_id

JOIN Stores s ON si.Store\_id = s.Store\_id

JOIN Dim\_City c ON s.City\_id = c.City\_id

WHERE i.Item\_id = 'SPECIFIC\_ITEM';

* + **Query 6 (Customer Location):**

SELECT c.City\_name, c.State

FROM Customer cu

JOIN Dim\_City c ON cu.City\_id = c.City\_id

WHERE cu.Customer\_id = 'SPECIFIC\_CUSTOMER';

**b. Data Verification**

* **Verification Process:**
  + Cross-check OLAP report outputs with source data in the operational databases.
  + Implement reconciliation queries to compare aggregated values in the data warehouse with sums derived from the source tables.
  + Use MySQL triggers or scheduled verification scripts to ensure that updates in the operational databases are reflected in the warehouse.
* **Example Verification Query:**

SELECT i.Item\_id, SUM(si.Quantity\_held) AS Warehouse\_Total,

(SELECT SUM(Quantity\_held) FROM Stored\_items WHERE Item\_id = i.Item\_id) AS Source\_Total

FROM Items i

JOIN Inventory\_Cube si ON i.Item\_id = si.Item\_id

GROUP BY i.Item\_id;

This query helps ensure that the cube’s data accurately reflects the source.

**7. Conclusion**

The project demonstrates a comprehensive approach to designing a data warehouse for a multi-store customer order processing system. By integrating data from disparate operational databases and employing a star schema design, the system enables fast and flexible OLAP analysis. The implementation of data cubes and the automated ETL process ensures that management can access real-time, aggregated data for decision-making. The ability to execute complex queries such as inventory checks, order fulfillment validation, and customer segmentation highlights the value of the data warehouse as a strategic asset for the enterprise.