Article:

Material Procurement Management for Construction Projects

-Ohm Sai Bhargav Aluri (000942201)

https://github.com/OHMALURI/DATASCIENCE

INDEX:

A Case Study on Material Procurement Management for Construction Projects

Abstract:

Efficient procurement management is Key for maintaining a seamless supply chain in the construction industry. This article presents an optimized database design specifically made for a Hypothetical material procurement company handling construction projects. The system is designed to enhance decision-making capabilities, ensure data accuracy, and effectively manage resource allocation. The proposed database incorporates key entities, attributes, and relationships to streamline the procurement of materials, optimize inventory control, and track the entire procurement cycle from supplier selection to material delivery. By establishing a scalable and error-resistant foundation, this system aims to meet the evolving needs of the construction industry and support the company's strategic objectives in project efficiency.

Keywords:

Procurement Management, Database Design, Inventory Management, Delivery Tracking, Decision Making, Data accuracy.

INTRODUCTION:

Material procurement is Important for construction projects, ensuring clients have the supplies they need to be successful. For a material procurement company, efficient processes are important because they directly impact timelines, costs, and quality. Lot of problems can occur like managing multiple suppliers and contracts, tracking inventory and material costs, timely deliveries.

These issues can lead to inefficient managing, making it hard to stay on schedule. To address these issues, creating an optimized database is crucial. A well-designed database can simplify procurement processes, improve data accuracy, and help with better decision-making. This article States the importance of developing a strong database for material procurement companies and how it can enhance efficiency and support project success.

Methodology:

This case study uses a structured approach to develop a procurement management database for a material procurement company. The steps are as follows:

1. **Understanding Organization Process:** Gaining a clear understanding of the company's procurement processes.
2. **Defining the Purpose:** list the goals of the database system, focusing on the information to be stored, such as suppliers, materials, and purchase orders.
3. **Identifying Entity:** Identifying key entities for the database, including Suppliers, Materials, Purchase Orders, and Inventory.
4. **Attribute Specification:** Specify the necessary attributes for each entity. For example, the Suppliers entity may include SupplierID, Name, and ContactInfo.
5. **Entity Relationships:** Define how entities interact. For instance, Suppliers provide Materials, creating a one-to-many relationship.
6. **Primary and Foreign Keys:** Select primary keys to uniquely identify records, like SupplierID for Suppliers and MaterialID for Materials. Identifying foreign keys to represent relationships, such as linking Purchase Orders to Suppliers.
7. **Data Integrity Constraints:** Establish rules to maintain data accuracy and consistency, including unique values and referential integrity.
8. **Developing Database:** Building the database using MYSQL based on the defined schema.
9. **Data Population:** Populate the database with relevant data, ensuring accuracy through validation checks.
10. **Testing:** Conduct testing with queries to ensure the database functions correctly.

Mission Statement:

The mission of this Procurement Management System is to enhance decision-making, ensure accuracy, and optimize resource allocation for all construction projects.

Objectives:

- **Building an Efficient Database Design:** Developing a robust database that reduces data redundancy and enhances data quality.
- **Optimize Inventory Management:** Making sure that inventory levels are managed efficiently to avoid Inconsistencies
- **Improving Procurement Tracking:** Providing proper visibility in the procurement cycle to optimize supplier performance and delivery timelines.
- **Enhanced Reporting:** Generating insightful reports on procurement to support data-driven decision-making and improve overall procurement strategies.

Designing Of Database:

Identifying Entities:

In the initial stage of developing the procurement management database for a material procurement company, a preliminary list of tables were identified to understand the essential entities involved in the procurement process. As the design evolved, some tables were refined or eliminated to create a more efficient and robust database structure.

Preliminary Tables Identified:

Project, Client, Material, Supplier, Warehouse, Employee, Department, Inventory, Procurement, Budget alert, Finance

These tables were designed to understand the basic functions required for managing projects, clients, materials, suppliers, and inventory. However, through further analysis and refinement, it became clear that some tables needed some changes .After iterative discussions and design reviews, the following changes were made to optimize the database structure:

- **Procurement**: Procurement identifies as a vast process that can be easily represented through the **Order** table. To simplify the database, the **Order** table is included to handle all procurement-related activities, while the **Delivery** table is included to track the delivery of orders.
- **Budget Alert**: The **Budget Alert** table did not align with the objectives of the inventory management system. As a result, it has been removed .
- **Finance:** The financial aspect of the procurement process is not a primary focus of this case study. the **Finance** table has been removed from the database design.

Final Tables Adopted

After carefully analyzing the initial design and the identified needs of the procurement process, the following tables were finalized for the database:

1. **Project:** tells about project details
2. **Client:** Stores all the client information,
3. **Material:** Represents materials available for procurement, detailing pricing and unit measurements.
4. **Supplier:** stores supplier info, including contact details and email .
5. SupplyMaterial : establishes many to many relationships between the supplier and material table.
6. **Warehouse:** stores warehouse details, including capacity and location for effective inventory management.
7. **Employee:** Stores employee information
8. **Department:** Stores various departments info within the organization, allowing for organized employee management.
9. **Inventory:** Manages Inventory levels and provides detailed tracking of materials available in the warehouse.

10. **Order:** stores purchase orders, connecting to the Supplier, Material, and Project tables for comprehensive order records.
11. **Delivery:** stores delivery details for orders, ensuring accurate tracking of order fulfillment.

Fields and Datatype:

These are the fields and data types identified in this system, defining attributes for each entity. The data types represent the nature of data stored in each field, while constraints such as primary keys, foreign keys. **Constraints for these attributes can be found in Appendix 1 of this article.**

1. Client Table(Dimension)

ClientID (Primary Key, INT), ClientName (VARCHAR), ContactPerson (VARCHAR), ContactNumber (BIGINT), Email (VARCHAR), Address (VARCHAR)

2. Material Table(Dimension)

MaterialID (Primary Key, INT), MaterialName (VARCHAR), UnitPrice (DECIMAL), UnitOfMeasure (VARCHAR)

3. Supplier Table(Dimension)

SupplierID (Primary Key, INT), SupplierName (VARCHAR), ContactPerson (VARCHAR), ContactNumber (BIGINT), Email (VARCHAR), Address (VARCHAR)

4. Department Table(Dimension)

DepartmentID (Primary Key, INT), DepartmentName (VARCHAR)

5. Employee Table(Dimension)

EmployeeID (Primary Key, INT), FirstName (VARCHAR), LastName (VARCHAR), Position (VARCHAR), ContactNo (BIGINT), Email (VARCHAR), DepartmentID (INT, Foreign Key)

6. Warehouse Table(Dimension)

- **WarehouseID (Primary Key), WarehouseName (VARCHAR), Location (VARCHAR), Capacity (DECIMAL)**

7. SupplierMaterial Table(Fact)

- **SupplierMaterialID (Pk), SupplierID (Foreign key), MaterialID (Foreign key)**

8. Project Table(Dimension)

ProjectID (Primary Key, INT), ClientID (Foreign key), ProjectName (VARCHAR), StartDate (DATE), EndDate (DATE) Location (VARCHAR), Budget (DECIMAL)

9. Inventory Table(Fact)

InventoryID (Pk, INT), SupplierMaterialID (Foreign Key) , ProjectID (Foreign Key), WarehouseID (Foreign key), QuantityOnHand (DECIMAL), LastUpdated (DATE), ReorderLevel (DECIMAL)

10. Order Table(Fact)

OrderID (Primary Key), SupplierMaterialID (Foreign Key), ProjectID (Foreign Key),OrderDate (DATE) QuantityOrdered (DECIMAL), TotalCost (DECIMAL), EmployeeID (Foreign Key)

11. Delivery Table(Fact)

DeliveryID (Primary Key, INT), OrderID (Foreign Key), EmployeeID (Foreign key), DeliveryDate (DATE), QuantityDelivered (DECIMAL), DeliveryNote (VARCHAR)

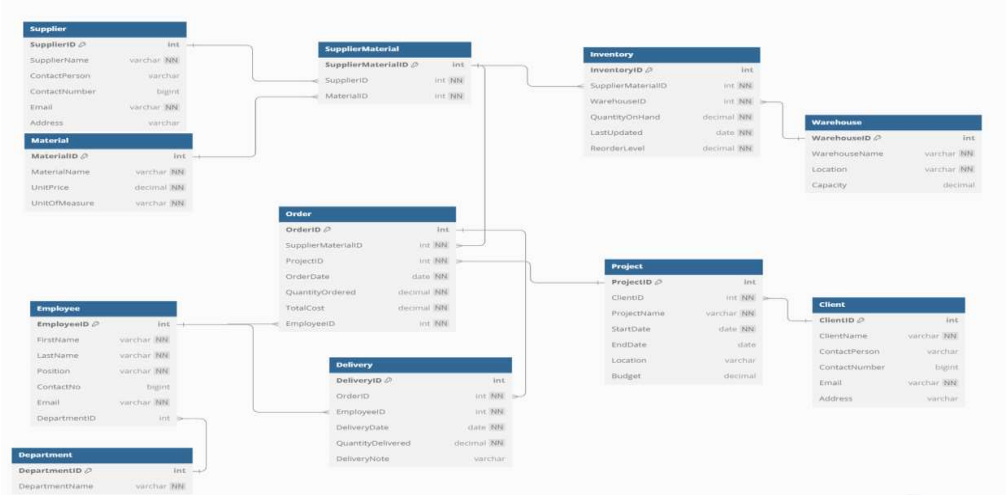
Relationships:

These are the relationships identified b/w the entities . These relationships maintain efficient management of procurement, inventory, orders, and deliveries, supporting seamless coordination within the organization.

- Client ↔ Project(one to many): One client can have multiple projects.
- Supplier ↔ Material(many to many) :One supplier can provide multiple materials and vise versa
- SupplierMaterial ↔ Inventory(one to many): One supplier-material combination can be found in many inventories.
- Warehouse ↔ Inventory(one to many): One warehouse can store multiple inventories.
- Project ↔ Order(one to many): Each project can have multiple orders.
- SupplierMaterial ↔ Order(one to many): Each order contains one supplier-material pair.
- Employee ↔ order(one to many): One employee handles multiple orders.
- Order ↔ Delivery(one to many): Multiple deliveries can be associated with a single order.
- Employee ↔ Delivery(one to many): One employee may handle multiple deliveries.
- Department ↔ Employee(one to many): Each department can have multiple employee

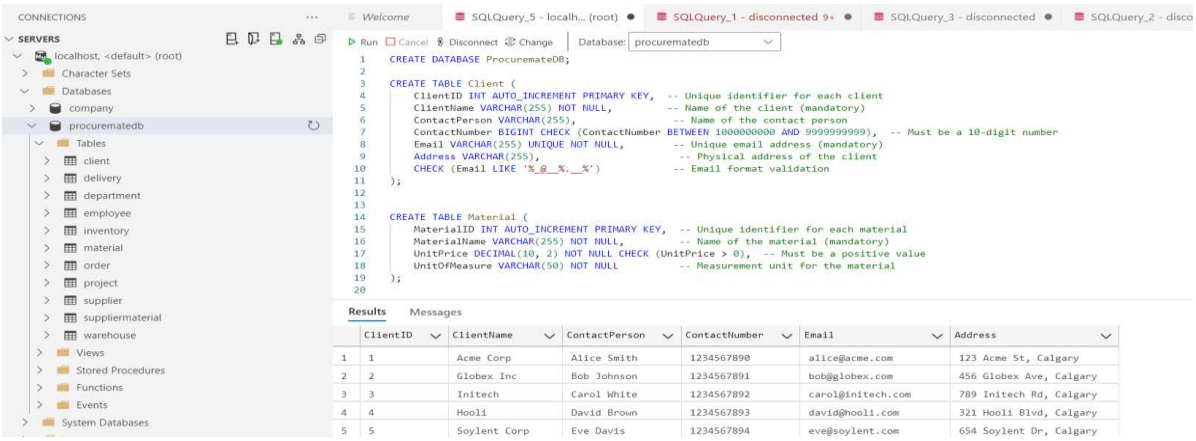
Final Database Design:

This is the final database design for the procurement and inventory management system, It how entities like clients, projects, suppliers, materials, and employees relate to each other.It shows how the process is happening . This design helps manage procurement, inventory, orders, and deliveries effectively while maintaining data integrity and consistency.



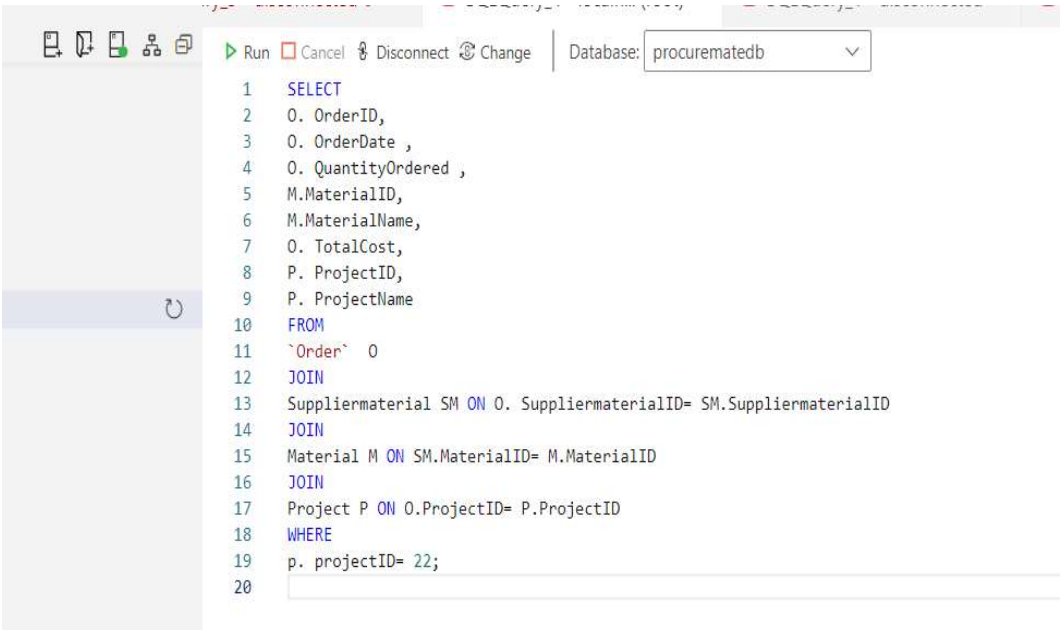
Creation Of Database Using MySQL :

A database named **'Procurematedb'** has been created using all the mentioned tables, attributes, and constraints, and it has been populated with sample data.



Testing Database with Queries:

The provided SQL query is used to retrieve Info about orders for a Particular project with ProjectID = 22. It joins multiple tables (Order, SupplierMaterial, Material, and Project) to extract relevant data.By combining these tables, the query provides a comprehensive view of orders associated with the specified project, including material details and supplier relationships.



Results Messages

	OrderID	OrderDate	QuantityOrdered	MaterialID	MaterialName	TotalCost	ProjectID	ProjectName
1	35	2024-01-05	20.00	5	Paint	900.00	22	Project Beta
2	36	2024-01-06	30.00	4	Glass	6000.00	22	Project Beta
3	53	2024-01-23	8.00	3	Wood	640.00	22	Project Beta

The query joins multiple tables and retrieves the required data, indicating that the relationships between the Order, SupplierMaterial, Material, and Project tables are properly maintained and operational. This demonstrates that the database is working as intended for querying and retrieving information.

Conclusion :

In conclusion, the proposed database design for the hypothetical material procurement company gives a robust solution for managing the procurement for the procurement comany. By ordering the procurement process, optimizing inventory control, and enhancing tracking , the system makes sure that it has more efficient use of resources and supports better decision-making. The incorporation of key entities, attributes, and relationships establishes a reliable foundation that not only meets current operational needs but is also adaptable to future demands. Ultimately, this optimized database design provides to a seamless supply chain, fostering project efficiency and aligning with the company's strategic objectives.

Learnings:

From this project, I learned that efficient procurement management is Important for maintaining a smooth supply chain in the construction industry. An optimized database design plays a key role in enhancing decision-making, ensuring data accuracy, and improving inventory management. Establishing clear relationships between key entities, such as suppliers, materials, orders, and projects, significantly ordered the procurement process and allowed for better resource allocation. Using normalization process helped to avoid data redundancy. well-structured schema gave a strong foundation for efficient procurement management and resource allocation.

Future Scope:

The future scope of this case study includes several improvements. We can expand the database to include more advanced features, such as automatic supplier evaluations and real-time inventory tracking. Adding predictive analytics would help forecast material needs and spot potential supply chain problems. We could also enhance financial tracking to help with budgeting and cost management.

Appendix

Appendix 1:

Tables and Constraints :

1. Client Table (Dimension)

Field	Data Type	Constraint
ClientID	INT	Primary Key, Auto Increment
ClientName	VARCHAR	Not Null
ContactPerson	VARCHAR	NOT NULL
ContactNumber	BIGINT	Check Constraint: CHECK (ContactNumber BETWEEN 1000000000 AND 9999999999)
Email	VARCHAR	Unique, Not Null, Check Constraint: CHECK (Email LIKE '%_@%.%')
Address	VARCHAR	NOT NUll

2. Material Table (Dimension)

Field	Data Type	Constraint
MaterialID	INT	Primary Key, Auto Increment
MaterialName	VARCHAR	Not Null
UnitPrice	DECIMAL	Not Null, Check Constraint: CHECK (UnitPrice > 0)
UnitOfMeasure	VARCHAR	Not Null

3. Supplier Table (Dimension)

Field	Data Type	Constraint
SupplierID	INT	Primary Key, Auto Increment
SupplierName	VARCHAR	Not Null
ContactPerson	VARCHAR	NOT NULL
ContactNumber	BIGINT	Check Constraint: CHECK (ContactNumber BETWEEN 1000000000 AND 9999999999)
Email	VARCHAR	Unique, Not Null, Check Constraint: CHECK (Email LIKE '%_@%.%')
Address	VARCHAR	NOT NULL

4. Department Table (Dimension)

Field	Data Type	Constraint

DepartmentID	INT	Primary Key, Auto Increment
DepartmentName	VARCHAR	Not Null

5. Employee Table (Dimension)

Field	Data Type	Constraint
EmployeeID	INT	Primary Key, Auto Increment
FirstName	VARCHAR	Not Null
LastName	VARCHAR	Not Null
Position	VARCHAR	Not Null
ContactNo	BIGINT	Check Constraint: CHECK (ContactNo BETWEEN 1000000000 AND 9999999999)
Email	VARCHAR	Unique, Not Null, Check Constraint: CHECK (Email LIKE '%_@%.%')
DepartmentID	INT	Foreign Key: REFERENCES Department(DepartmentID)

6. Warehouse Table (Dimension)

Field	Data Type	Constraint
WarehouseID	INT	Primary Key, Auto Increment
WarehouseName	VARCHAR	Not Null
Location	VARCHAR	Not Null
Capacity	DECIMAL	Check Constraint: CHECK (Capacity > 0)

7. SupplierMaterial Table (Fact)

Field	Data Type	Constraint
SupplierMaterialID	INT	Primary Key, Auto Increment
SupplierID	INT	Not Null, Foreign Key: REFERENCES Supplier(SupplierID)
MaterialID	INT	Not Null, Foreign Key: REFERENCES Material(MaterialID)
		Unique Constraint: UNIQUE (SupplierID, MaterialID)

8. Project Table (Dimension)

Field	Data Type	Constraint
ProjectID	INT	Primary Key, Auto Increment
ClientID	INT	Not Null, Foreign Key: REFERENCES Client(ClientID)
ProjectName	VARCHAR	Not Null
StartDate	DATE	Not Null
EndDate	DATE	NOT NULL
Location	VARCHAR	NOT NULL
Budget	DECIMAL	Check Constraint: CHECK (Budget >= 0)

9. Inventory Table (Fact)

Field	Data Type	Constraint
InventoryID	INT	Primary Key, Auto Increment
SupplierMaterialID	INT	Not Null, Foreign Key: REFERENCES SupplierMaterial(SupplierMaterialID)
ProjectID	INT	Not Null, Foreign Key: REFERENCES Project(ProjectID)
WarehouseID	INT	Not Null, Foreign Key: REFERENCES Warehouse(WarehouseID)
QuantityOnHand	DECIMAL	Check Constraint: CHECK (QuantityOnHand >= 0)
LastUpdated	DATE	Not Null
ReorderLevel	DECIMAL	Check Constraint: CHECK (ReorderLevel >= 0)

10. Order Table (Fact)

Field	Data Type	Constraint
OrderID	INT	Primary Key, Auto Increment
SupplierMaterialID	INT	Not Null, Foreign Key: REFERENCES SupplierMaterial(SupplierMaterialID)
ProjectID	INT	Not Null, Foreign Key: REFERENCES Project(ProjectID)
OrderDate	DATE	Not Null
QuantityOrdered	DECIMAL	Check Constraint: CHECK (QuantityOrdered > 0)
TotalCost	DECIMAL	Check Constraint: CHECK (TotalCost > 0)
EmployeeID	INT	Not Null, Foreign Key: REFERENCES Employee(EmployeeID)

11. Delivery Table (Fact)

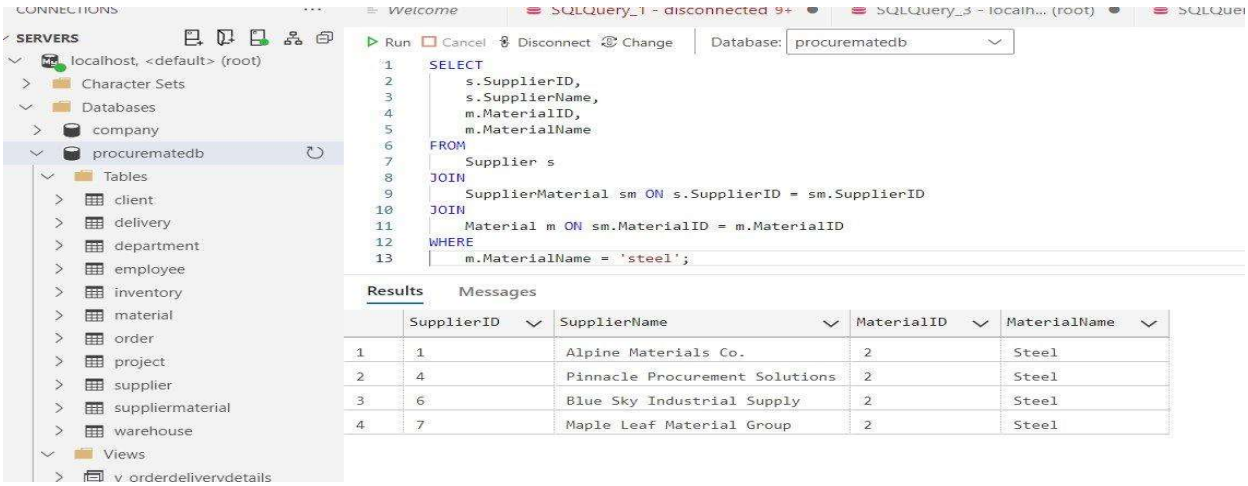
Field	Data Type	Constraint
DeliveryID	INT	Primary Key, Auto Increment
OrderID	INT	Not Null, Foreign Key: REFERENCES Order(OrderID)
EmployeeID	INT	Not Null, Foreign Key: REFERENCES Employee(EmployeeID)
DeliveryDate	DATE	Not Null
QuantityDelivered	DECIMAL	Check Constraint: CHECK (QuantityDelivered > 0)
DeliveryNote	VARCHAR	NOT NULL

Appendix 2:

Queries/ views :

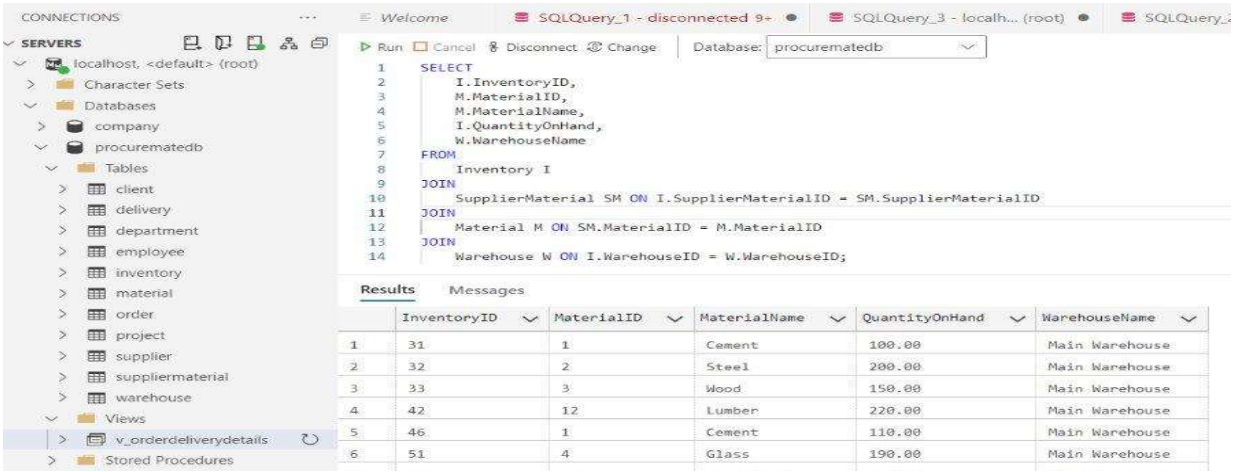
Query 1: Identify Suppliers for Specific Material (ex: steel)

In this query i used the suppliermaterial table and its many to many relationships to identify different suppliers for a single material, in this case it is steel.



Query 2:

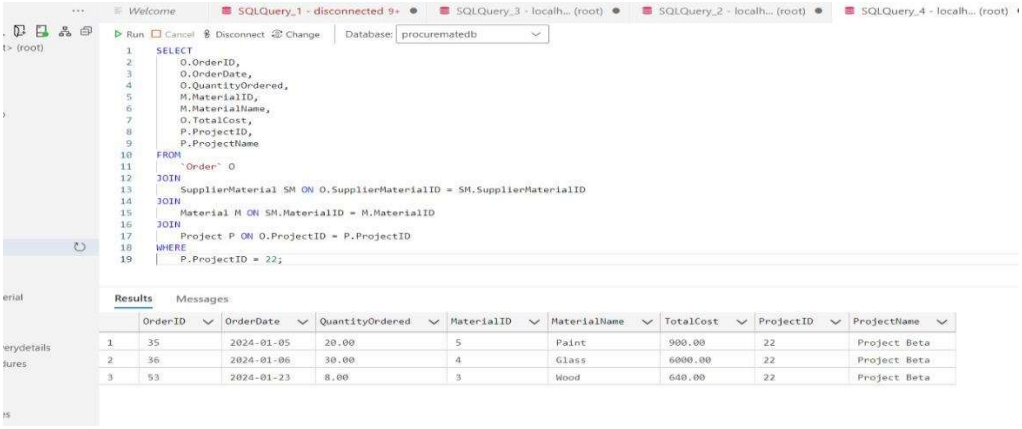
provides valuable insights into the inventory status of various materials across different warehouse locations.In this query inventory table is used to get the overall info about the inventory status with the help of other tables like supplymaterial, warehouse by using JOIN operation.



Query 3:

materials ordered for a specific project (ex projectid:22)

In this query, order table is used, with the help of join operation with material, project table to identify the different orders made by a single project, and their order details including cost. This query also states the one-to-many relationship b/w the projects and orders table. In this case we are using project 22.



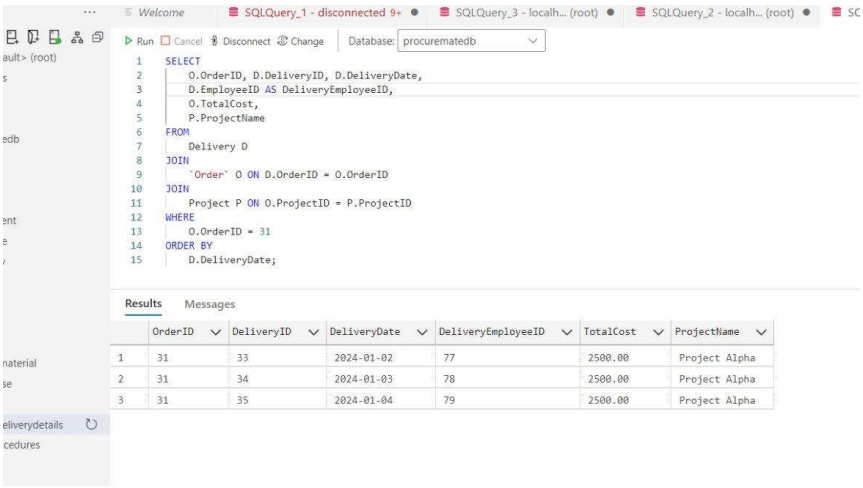
```
SELECT
    O.OrderID,
    O.OrderDate,
    O.QuantityOrdered,
    M.MaterialID,
    M.MaterialName,
    O.TotalCost,
    P.ProjectID,
    P.ProjectName
FROM
    `Order` O
JOIN
    SupplierMaterial SM ON O.SupplierMaterialID = SM.SupplierMaterialID
JOIN
    Material M ON SM.MaterialID = M.MaterialID
JOIN
    Project P ON O.ProjectID = P.ProjectID
WHERE
    P.ProjectID = 22;
```

	OrderID	OrderDate	QuantityOrdered	MaterialID	MaterialName	TotalCost	ProjectID	ProjectName
1	35	2024-01-05	20.00	5	Paint	900.00	22	Project Beta
2	36	2024-01-06	30.00	4	Glass	6000.00	22	Project Beta
3	53	2024-01-23	8.00	3	Wood	640.00	22	Project Beta

Query 4:

tracking all deliveries made for a particular order (ex orderID: 31)

In this query delivery and order tables are used to get the information regarding each delivery and the number of deliveries per order, this query proves the one-to-many relations b/w the order and the delivery table and gives the whole info regarding delivery, including its Delivery agent, etc.



```
SELECT
    O.OrderID, D.DeliveryID, D.DeliveryDate,
    D.EmployeeID AS DeliveryEmployeeID,
    O.TotalCost,
    P.ProjectName
FROM
    Delivery D
JOIN
    `Order` O ON D.OrderID = O.OrderID
JOIN
    Project P ON O.ProjectID = P.ProjectID
WHERE
    O.OrderID = 31
ORDER BY
    D.DeliveryDate;
```

	OrderID	DeliveryID	DeliveryDate	DeliveryEmployeeID	TotalCost	ProjectName
1	31	33	2024-01-02	77	2500.00	Project Alpha
2	31	34	2024-01-03	78	2500.00	Project Alpha
3	31	35	2024-01-04	79	2500.00	Project Alpha