

***Introduction To Database Systems***

**Name: Rabail Fiaz**

**Reg. #: L1F22BSCS1096**

**Section: D-16**

**Project:Bricks Company Management System**

**Submitted To: Sir Usama Pervaiz**

***Description:***

The Brick Inventory Management System is a database project designed for a brick manufacturing business. It includes tables for managing inventory, orders, customers, sales, employees, transportation, production, suppliers, and equipment. The system allows for efficient tracking of brick sizes, quantities, and prices, as well as customer details and order statuses. It supports data manipulation and complex queries for reporting purposes. This comprehensive solution enhances operational efficiency and facilitates informed decision-making within the business. Overall, it streamlines key processes in brick manufacturing.

***Entities***

1. **Bricks Inventory**

|  |  |  |
| --- | --- | --- |
| Attributes | Data type |  |
| Size | int | Primary key |
| Quantity | int |  |
| Unit Price | float |  |
| Total Cost | float |  |
| Manufacturer | varchar |  |
| P\_id | int |  |
| S\_id | int |  |

1. **Orders**

|  |  |  |
| --- | --- | --- |
| Attributes | Data type |  |
| **Order Date** | date |  |
| Delivery Date | date |  |
| Quantity | int |  |
| Total price | float |  |
| status | char |  |
| B\_id | int | Primary key |
| C\_id | int |  |
| E-id | int |  |

1. **Customers**

|  |  |  |
| --- | --- | --- |
| Attributes | Data type |  |
| **C\_name** | char |  |
| Email | varchar |  |
| Address | varchar |  |
| Credit Limit | int |  |
| Payment History | varchar |  |
| Payment Date | date |  |
| C\_id | int | Primary key |

1. **Sales**

|  |  |  |
| --- | --- | --- |
| Attributes | Data type |  |
| **Sale Date** | date | Primary key |
| Quantity Sold | int |  |
| Payment Status | char |  |

1. **Employees**

|  |  |  |
| --- | --- | --- |
| Attributes | Data type |  |
| **Ename** | char |  |
| Position | char |  |
| Contact Info | varchar |  |
| salary | float |  |
| Hire Date | date |  |
| P\_id | int | Primary key |

1. **Transportation**

|  |  |  |
| --- | --- | --- |
| Attributes | Data type |  |
| **Driver** | char | Primary key |
| Route | char |  |
| Departure Time | varchar |  |
| Arrival Time | varchar |  |
| Mileage | char |  |
| Fuel Consumption | int |  |
| E\_id | int |  |

1. **Production**

|  |  |  |
| --- | --- | --- |
| Attributes | Data type |  |
| **Quantity Produced** | int |  |
| Production Date | date |  |
| Manufacturing Plant | char |  |
| Production Cost | float |  |
| E\_id | int |  |
| S-id | int | Primary key |

1. **Suppliers**

|  |  |  |
| --- | --- | --- |
| Attributes | Data type |  |
| **Sup\_name** | char | Primary key |
| Contact Info | varchar |  |
| Address | varchar |  |
| Payment Terms | char |  |
| Materials | char |  |

1. **Equipment**

|  |  |  |
| --- | --- | --- |
| Attributes | Data type |  |
| **Etype** | char | Primary key |
| Manufacturer | char |  |
| Purchase date | date |  |
| Maintenance History | char |  |

***Relationships***

**1. BrickInventory**

- P\_id (foreign key) references Production: Each brick inventory entry can be linked to a production record.

- S\_id (foreign key) references Suppliers: Each brick inventory entry is sourced from a supplier.

**2. Orders**

- B\_id (foreign key) references BrickInventory: Each order is associated with a specific brick inventory item.

- C\_id (foreign key) references Customers: Each order is placed by a customer.

- E\_id (foreign key) references Employees: Each order is processed by a specific employee.

**3. Customers**

- C\_id (primary key): Unique identifier for each customer. This serves as a reference in the Orders table.

**4. Sales**

- Not directly linked to other tables in the provided SQL, but could be linked to Orders through sales records that reference specific orders.

**5. Employees**

- E\_id (foreign key) in both Transportation and Orders: Each employee can be responsible for processing orders and driving transportation.

**6. Transportation**

- E\_id (foreign key) references Employees: Each transportation record is managed by a specific employee.

**7. Production**

- E\_id (foreign key) references Employees: Each production entry is managed by an employee.

- S\_id (foreign key) references Suppliers: Each production entry can involve materials sourced from suppliers.

**8. Suppliers**

- S\_id (primary key): Unique identifier for each supplier. This serves as a reference in the BrickInventory and Production tables.

**9. Equipment**

- Not directly linked to other tables in the provided SQL, but could be tied to Production through equipment used in manufacturing.

***Summary of Relationships:***

- BrickInventory ↔ Production (1:M)

- BrickInventory ↔ Suppliers (M:1)

- Orders ↔ BrickInventory (M:1)

- Orders ↔ Customers (M:1)

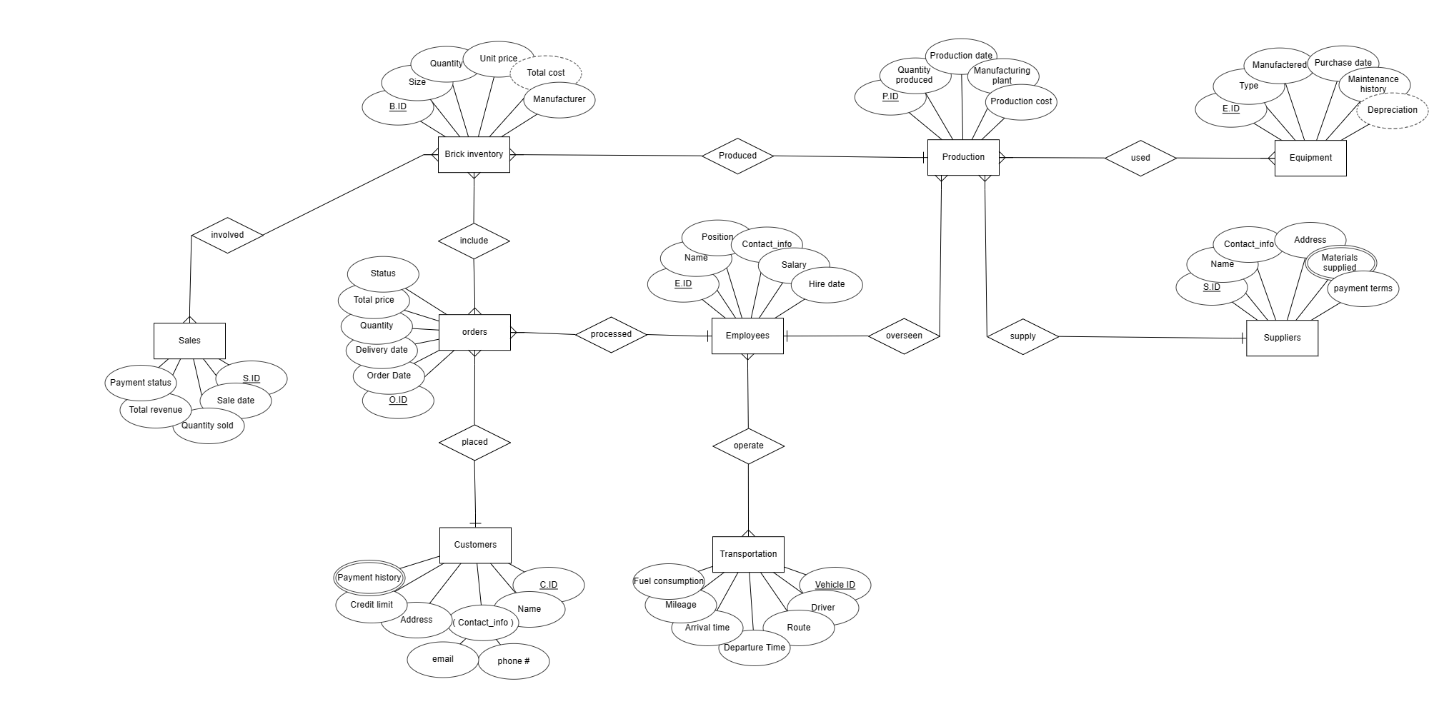
- Orders ↔ Employees (M:1)

- Transportation ↔ Employees (M:1)

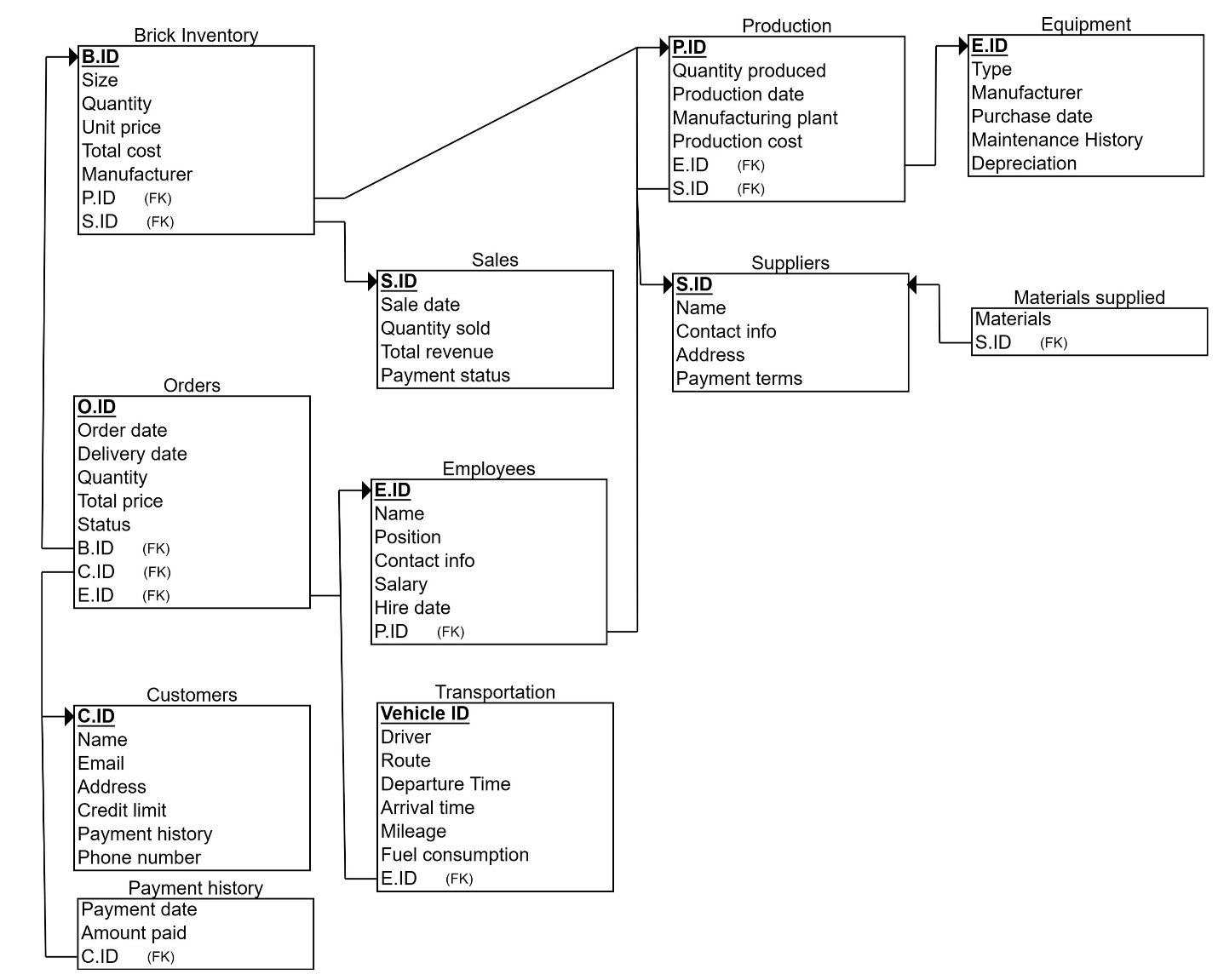
- Production ↔ Employees (M:1)

- Production ↔ Suppliers (M:1)

***“ERD”***



***“Relational schema”***



***“DDL”***

1. **Creating Tables:**

create database project;

use project;

create table BrickInventory(

size int(50),

quantity int(150),

unit\_price float(50),

total\_cost float(50),

manufacturer varchar(30),

P\_id int(5),

S\_id int(5)

);

select\*from BrickInventory;

create table Orders(

order\_date date,

delivery\_date date,

quantity int(150),

total\_price float(50),

status char(10),

B\_id int(5),

C\_id int(5),

E\_id int(5)

);

select\*from Orders;

create table customers(

C\_Name char(20),

email varchar(30),

Address varchar(30),

credit\_limit int(15),

payment\_history varchar(20),

phone\_num varchar(50),

payment\_date date,

C\_id int(5)

);

select\*from customers;

create table sales(

sale\_date date,

quantity\_sold int(150),

Payment\_status char(20)

);

select\*from sales;

create table employees(

Ename char(20),

position char(120),

contact\_info varchar(15),

salary int(150),

hire\_date date,

P\_id int(5)

);

select\*from employees;

create table transportation(

Driver char(20),

Route char(20),

Departure\_time varchar(15),

Arrival\_time varchar(15),

mileage char(15),

Fuel\_consumption float(50),

E\_id int(5)

);

select\*from transportation;

create table production(

Quantity\_produced int(150),

production\_date date,

Manufacturing\_plant char(20),

production\_cost float(150,1),

E\_id int(5),

S\_id int(5)

);

select\*from production;

create table suppliers(

Sup\_name char(20),

contactInfo varchar(15),

Address varchar(20),

payment\_terms char(20),

Materials char(30)

);

select\*from suppliers;

create table equipment(

etype char(120),

manufacturer char(120),

purchase\_date date,

maintenance\_history char(120)

);

select\*from equipment;

1. **Data Insertion**

-- BrickInventory

insert into BrickInventory (size, quantity, unit\_price, total\_cost, manufacturer, P\_id, S\_id)

values

(8, 1000, 0.5, 500, 'ABC Bricks', 1, 1),

(6, 800, 0.4, 320, 'XYZ Bricks', 2, 2),

(4, 1200, 0.3, 360, '123 Bricks', 3, 3),

(10, 600, 0.6, 360, '456 Bricks', 4, 4),

(12, 500, 0.7, 350, '789 Bricks', 5, 5);

select\*from BrickInventory;

-- Orders

insert into Orders (order\_date, delivery\_date, quantity, total\_price, status, B\_id, C\_id, E\_id)

values

('2024-05-01', '2024-05-10', 200, 150, 'Pending', 1, 1, 1),

('2024-05-02', '2024-05-12', 300, 200, 'Delivered', 2, 2, 2),

('2024-05-03', '2024-05-15', 400, 250, 'Pending', 3, 3, 3),

('2024-05-04', '2024-05-18', 500, 300, 'Delivered', 4, 4, 4),

('2024-05-05', '2024-05-20', 600, 350, 'Pending', 5, 5, 5);

select\*from Orders;

-- Customers

insert into customers (C\_Name, email, Address, credit\_limit, payment\_history, phone\_num, payment\_date, C\_id)

values

('John Doe', 'john.doe@example.com', '123 Main St', 1000, 'Good', 1234567890, '2024-05-25', 1),

('Jane Smith', 'jane.smith@example.com', '456 Oak St', 1500, 'Excellent', 9876543210, '2024-05-28', 2),

('Bob Johnson', 'bob.johnson@example.com', '789 Elm St', 2000, 'Good', 5555555555, '2024-05-30', 3),

('Alice Williams', 'alice.williams@example.com', '101 Pine St', 2500, 'Excellent', 4444444444, '2024-06-02', 4),

('Michael Brown', 'michael.brown@example.com', '202 Maple St', 3000, 'Good', 7777777777, '2024-06-05', 5);

select\*from customers;

-- Sales

insert into sales (sale\_date, quantity\_sold, Payment\_status)

values

('2024-05-01', 150, 'Paid'),

('2024-05-02', 200, 'Paid'),

('2024-05-03', 250, 'Pending'),

('2024-05-04', 300, 'Paid'),

('2024-05-05', 350, 'Pending');

select\*from sales;

-- Employees

insert into employees (Ename, position, contact\_info, salary, hire\_date, P\_id)

values

('John Smith', 'Manager', 1111111111, 5000, '2020-01-01', 1),

('Jane Doe', 'Sales Repres', 2222222222, 3000, '2021-02-01', 2),

('Bob Brown', 'Accountant', 3333333333, 3500, '2021-03-15', 3),

('Alice Johnson', 'Production Superv', 4444444444, 3200, '2022-05-10', 4),

('Michael Williams', 'Transportation Manager', 5555555555, 3800, '2023-04-20', 5);

select\*from employees;

-- Transportation

insert into transportation (Driver, Route, Departure\_time, Arrival\_time, mileage, Fuel\_consumption, E\_id)

values

('Tom', 'Route 1', '08:00', '10:00', '100 miles', 20.5, 1),

('Emily', 'Route 2', '09:00', '11:30', '150 miles', 25.5, 2),

('David', 'Route 3', '10:30', '13:00', '120 miles', 22.5, 3),

('Sarah', 'Route 4', '11:00', '13:30', '130 miles', 23.5, 4),

('Chris', 'Route 5', '12:00', '14:30', '140 miles', 24.5, 5);

select\*from transportation;

-- Production

insert into production (Quantity\_produced, production\_date, Manufacturing\_plant, production\_cost, E\_id, S\_id)

values

(1000, '2024-05-01', 'Plant 1', 5000, 1, 1),

(1200, '2024-05-02', 'Plant 2', 6000, 2, 2),

(1500, '2024-05-03', 'Plant 3', 7500, 3, 3),

(1800, '2024-05-04', 'Plant 4', 9000, 4, 4),

(2000, '2024-05-05', 'Plant 5', 10000, 5, 5);

select\*from production;

-- Suppliers

insert into suppliers (Sup\_name, contactInfo, Address, payment\_terms, Materials)

values

('Supplier 1', 1111111111, '123 Supplier St', 'Net 30', 'Bricks'),

('Supplier 2', 2222222222, '456 Supplier St', 'Net 45', 'Concrete'),

('Supplier 3', 3333333333, '789 Supplier St', 'Net 60', 'Steel'),

('Supplier 4', 4444444444, '101 Supplier St', 'Net 30', 'Wood'),

('Supplier 5', 5555555555, '202 Supplier St', 'Net 45', 'Glass');

select\*from suppliers;

-- Equipment

insert into equipment (etype, manufacturer, purchase\_date, maintenance\_history)

values

('Crane', 'ABC Equipment', '2023-01-01', 'Regular maintenance every 3 months'),

('Forklift', 'XYZ Machinery', '2023-02-01', 'Maintenance performed as needed'),

('Excavator', '123 Inc.', '2023-03-01', 'Regular maintenance every 6 months'),

('Truck', '456 Vehicles', '2023-04-01', 'Regular maintenance every 4 months'),

('Loader', '789 Tools', '2023-05-01', 'Maintenance performed as needed');

select\*from equipment;

***“DML”***

1. **Updating Data:**

-- BrickInventory

update brickinventory

set size = 10, quantity = 1200, unit\_price = 0.4, total\_cost = 480, manufacturer = 'abc bricks', P\_id = 2, S\_id = 2

where size = 8;

-- Orders

update orders

set order\_date = '2024-05-02', delivery\_date = '2024-05-12', quantity = 350, total\_price = 220, status = 'delivered', B\_id = 3, C\_id = 3, E\_id = 3

where order\_date = '2024-05-01';

-- Customers

update customers

set C\_Name = 'john smith', email = 'john.smith@example.com', Address = '456 oak st', credit\_limit = 1200, payment\_history = 'good', phone\_num = 9876543210, payment\_date = '2024-05-26'

where C\_Name = 'John Doe';

-- Sales

update sales

set sale\_date = '2024-05-03', quantity\_sold = 300, Payment\_status = 'paid'

where sale\_date = '2024-05-01';

-- Employees

update employees

set Ename = 'jane doe', position = 'sales representative', contact\_info = 2222222222, salary = 3200, hire\_date = '2021-02-01', P\_id = 3

where Ename = 'John Smith';

-- Transportation

UPDATE transportation

SET Driver = 'tommy', Route = 'route 1', Departure\_time = '08:30', Arrival\_time = '10:30', mileage = '110 miles', Fuel\_consumption = 21.5, E\_id = 2

WHERE Driver = 'Tom';

-- Production

update production

set Quantity\_produced = 1300, production\_date = '2024-05-02', Manufacturing\_plant = 'plant 2', production\_cost = 6500, E\_id = 3, S\_id = 3

where Quantity\_produced = 1000;

-- Suppliers

update suppliers

set Sup\_name = 'supplier 1', contactInfo = 2222222222, Address = '456 supplier st', payment\_terms = 'net 45', Materials = 'concrete'

where Sup\_name = 'Supplier 1';

-- Equipment

update equipment

set etype = 'crane', manufacturer = 'abc equipment', purchase\_date = '2023-01-01', maintenance\_history = 'regular maintenance every 3 months'

where etype = 'Crane';

select \* from BrickInventory

where size < 10;

select \* from Orders

where status = 'Pending';

select Address from customers

where C\_Name like 'J%';

select quantity\_sold from sales

where Payment\_status = 'Paid';

select Ename from employees

where YEAR(hire\_date) = 2021;

select \* from transportation

where mileage > '100 miles' and mileage < '150 miles';

select production\_cost from production

where S\_id > 3;

1. **Data Deletion:**

-- Delete two rows from BrickInventory

delete from BrickInventory

limit 2;

select\*from BrickInventory;

-- Delete two rows from Orders

delete from Orders

limit 2;

select\*from Orders;

-- Delete two rows from customers

delete from customers

limit 2;

select\*from customers;

-- Delete two rows from sales

delete from sales

limit 2;

select\*from sales;

-- Delete two rows from employees

delete from employees

limit 2;

select\*from employees;

-- Delete two rows from transportation

delete from transportation

limit 2;

select\*from transportation;

-- Delete two rows from production

delete from production

limit 2;

select\*from production;

-- Delete two rows from suppliers

delete from suppliers

limit 2;

select\*from suppliers;

-- Delete two rows from equipment

delete from equipment

limit 2;

select\*from equipment;

***“JOINS”***

**1.Cross Joins:**

**Queries:**

SELECT \* FROM BrickInventory CROSS JOIN Orders;

SELECT \* FROM BrickInventory CROSS JOIN customers

SELECT \* FROM BrickInventory CROSS JOIN sales;

SELECT \* FROM BrickInventory CROSS JOIN employees;

SELECT \* FROM BrickInventory CROSS JOIN transportation;

1. **Self Join:**

**Queries:**

SELECT \* FROM employees e1 INNER JOIN employees e2 ON e1.Ename = e2.Ename;

SELECT \* FROM employees e1 INNER JOIN employees e2 ON e1.salary = e2.salary;

SELECT \* FROM employees e1 INNER JOIN employees e2 ON e1.position = e2.position;

SELECT \* FROM employees e1 INNER JOIN employees e2 ON e1.hire\_date = e2.hire\_date;

SELECT \* FROM employees e1 INNER JOIN employees e2 ON e1.contact\_info = e2.contact\_info;

**3. Left Outer Joins:**

**Queries:**

SELECT \* FROM BrickInventory LEFT OUTER JOIN Orders ON BrickInventory.P\_id = Orders.P\_id;

SELECT \* FROM Orders LEFT OUTER JOIN customers ON Orders.C\_id = customers.C\_id;

SELECT \* FROM customers LEFT OUTER JOIN sales ON customers.phone\_num = sales.phone\_num;

SELECT \* FROM sales LEFT OUTER JOIN employees ON sales.E\_id = employees.E\_id;

SELECT \* FROM employees LEFT OUTER JOIN transportation ON employees.E\_id = transportation.E\_id;

1. **Right outer Join**:

**Queries:**

SELECT \* FROM BrickInventory RIGHT OUTER JOIN Orders ON BrickInventory.P\_id = Orders.P\_id;

SELECT \* FROM Orders RIGHT OUTER JOIN customers ON Orders.C\_id = customers.C\_id;

SELECT \* FROM customers RIGHT OUTER JOIN sales ON customers.phone\_num = sales.phone\_num;

SELECT \* FROM sales RIGHT OUTER JOIN employees ON sales.E\_id = employees.E\_id;

SELECT \* FROM employees RIGHT OUTER JOIN transportation ON employees.E\_id = transportation.E\_id;

1. **Full Join:**

**Queries:**

SELECT \* FROM BrickInventory LEFT OUTER JOIN Orders ON BrickInventory.P\_id = Orders.P\_id

UNION

SELECT \* FROM BrickInventory RIGHT OUTER JOIN Orders ON BrickInventory.P\_id = Orders.P\_id;

SELECT \* FROM Orders LEFT OUTER JOIN customers ON Orders.C\_id = customers.C\_id UNION

SELECT \* FROM Orders RIGHT OUTER JOIN customers ON Orders.C\_id = customers.C\_id;

SELECT \* FROM customers LEFT OUTER JOIN sales ON customers.phone\_num = sales.phone\_num

UNION

SELECT \* FROM customers RIGHT OUTER JOIN sales ON customers.phone\_num = sales.phone\_num;

SELECT \* FROM sales LEFT OUTER JOIN employees ON sales.E\_id = employees.E\_id

UNION

SELECT \* FROM sales RIGHT OUTER JOIN employees ON sales.E\_id = employees.E\_id;

SELECT \* FROM employees LEFT OUTER JOIN transportation ON employees.E\_id = transportation.E\_id

UNION

SELECT \* FROM employees RIGHT OUTER JOIN transportation ON employees.E\_id = transportation.E\_id;

**6. Inner Join:**

**Queries:**

SELECT \* FROM BrickInventory INNER JOIN Orders ON BrickInventory.P\_id = Orders.P\_id;

SELECT \* FROM Orders INNER JOIN customers ON Orders.C\_id = customers.C\_id;

SELECT \* FROM customers INNER JOIN sales ON customers.phone\_num = sales.phone\_num;

SELECT \* FROM sales INNER JOIN employees ON sales.E\_id = employees.E\_id;

SELECT \* FROM employees INNER JOIN transportation ON employees.E\_id = transportation.E\_id;

**7. Natural join:**

**Queries:**

SELECT \* FROM BrickInventory NATURAL JOIN Orders;

SELECT \* FROM Orders NATURAL JOIN customers;

ELECT \* FROM customers NATURAL JOIN sales;

SELECT \* FROM sales NATURAL JOIN employees;

SELECT \* FROM employees NATURAL JOIN transportation;

***“Nested Queries”***

-- 1. Find Customers with Orders Greater than a Specific Quantity

SELECT C\_Name

FROM customers

WHERE C\_id IN (

SELECT C\_id

FROM Orders

WHERE quantity > 300

);

-- 2. Get Brick Inventory Items with Total Cost Above Average

SELECT size, total\_cost

FROM BrickInventory

WHERE total\_cost > (

SELECT AVG(total\_cost)

FROM BrickInventory

);

-- 3. List Employees Who Have Processed Orders for a Specific Customer

SELECT Ename

FROM employees

WHERE E\_id IN (

SELECT E\_id

FROM Orders

WHERE C\_id = 1

);

-- 4. Retrieve Suppliers with Materials Supplied to Orders Over a Certain Price

SELECT Sup\_name

FROM suppliers

WHERE Sup\_name IN (

SELECT DISTINCT Sup\_name

FROM BrickInventory

WHERE P\_id IN (

SELECT B\_id

FROM Orders

WHERE total\_price > 250

)

);

-- 5. Find Production Entries with Cost Greater than the Average Cost

SELECT Manufacturing\_plant, production\_cost

FROM production

WHERE production\_cost > (

SELECT AVG(production\_cost)

FROM production

);

***“Correlated Queries”***

-- 1. Find Total Sales for Each Customer

SELECT C\_Name,

(SELECT SUM(quantity\_sold)

FROM sales

WHERE sales.E\_id IN (

SELECT E\_id

FROM Orders

WHERE Orders.C\_id = customers.C\_id

)) AS total\_sales

FROM customers;

-- 2. List Employees with Orders They Processed and Their Payment Status

SELECT Ename, status

FROM employees e

WHERE EXISTS (

SELECT 1

FROM Orders o

WHERE o.E\_id = e.E\_id

);

-- 3. Get All Brick Inventory Items Sold in Orders with a Total Price Greater than a Specific Amount

SELECT size, quantity

FROM BrickInventory b

WHERE EXISTS (

SELECT 1

FROM Orders o

WHERE o.B\_id = b.P\_id AND o.total\_price > 200

);

-- 4. Find Employees with Salary Higher than the Average Salary of Their Position

SELECT Ename

FROM employees e1

WHERE salary > (

SELECT AVG(salary)

FROM employees e2

WHERE e1.position = e2.position

);

-- 5. List Customers with Payment History Better than Average

SELECT C\_Name

FROM customers c

WHERE credit\_limit > (

SELECT AVG(credit\_limit)

FROM customers

);

***The End***

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*