



Mini Project – Garment Factory

A report submitted to the

Department of Electrical and Information
Engineering Faculty of Engineering
University of Ruhuna
Sri Lanka

On 11TH of September 2023

In completing an assignment for the module
EE4350 Database Systems

By group No: 52:
SURAWEEERA S.A.Y.A. – EG/2020/4225
THANANCHAYAN T. – EG/2020/4227
THANAPATHI T.M.I.U.B. – EG/2020/4228

CONTENTS

1.	Chapter I – Requirement Analysis	2
1.1	Introduction.....	2
1.2	Data Requirement	2
2.	Chapter II – Conceptual Design (ER Diagram)	5
3.	Chapter IV – Implementation	7
3.1	Creating Database.....	7
3.2	Table Creation	7
3.3	Inserting Tuples	15
3.4	Updating and Deleting from Tables	23
4.	Chapter IV – Transactions	27
4.1	Simple Queries.....	27
4.2	Complex Queries	31
5.	Chapter V – Tuning.....	38

1. CHAPTER I – REQUIREMENT ANALYSIS

1.1 INTRODUCTION

In today's modern world, databases and database management systems are essential for day-to-day life. It involves nearly all day-to-day activities. For example, financial management systems in banks, hotel management systems, ticket booking systems like movie or for the travelling, airports, universities etc. small level businesses to huge level companies all are maintaining databases to store the data and retrieve easily. Using proper database management system, easily maintain a huge collection of data. When planning to create a database, need to consider so many things like purpose and requirements, performance, scalability, data security and integrity, data modelling, normalization techniques etc.

In this project, our group developed garment factory management system database. To develop this project, first we analyze the garment factory management system like how it operates, how processes are going on and as well as collect all required information to implement the database. This system aims to manage and organize factory data effectively as well as improve the garment factory performance. By using this system, the factory can manage huge amounts of data easily. This system fulfills all the functional and data requirements.

To complete this project, our project team mainly focused on creating a database system to manage whole garment factory works and function. First, we started to closely study the factory works and understand the factory operations and processes. Using this system factory can manage large amount of data easily as well as it full-fulfill all the factory's needs and helps factory to running without any issues.

1.2 DATA REQUIREMENT

Data requirements are the data which provide a detailed description of the data model, and the system must be used to fulfill its functional requirements. In the ER diagram, there are many entities and each entity has some attributes. These things are needed to create a garment factory management system database.

List of Entities

1. Employee
2. Order
3. Worker
4. Supplier
5. Product
6. Machine or Equipment
7. Raw Material
8. Seller
9. Department

Attributes List

Entity	Attributes (data requirements)
Employee	Employee ID
	Department
	Full Name
	Date of birth
	Contact
Worker	Worker ID
	Full Name
	Department
	Date of birth
	Contact
Product	Product ID
	Product Name
	Product type
	Department
	Stock Quality
Raw material	Raw material ID
	Row material Name
	Department
	Supplier

Department	Department ID
	Department Name
	Department Head
	No of workers
	No of Employees
Order	Order ID
	Seller
	Product
	Quantity
Supplier	Supplier Name
	Raw material
	Contact
Machine or Equipment	Machine ID
	Machine Name
	Department
	Function
Seller	Seller Name
	Order
	Contact

2. CHAPTER II – CONCEPTUAL DESIGN (ER DIAGRAM)

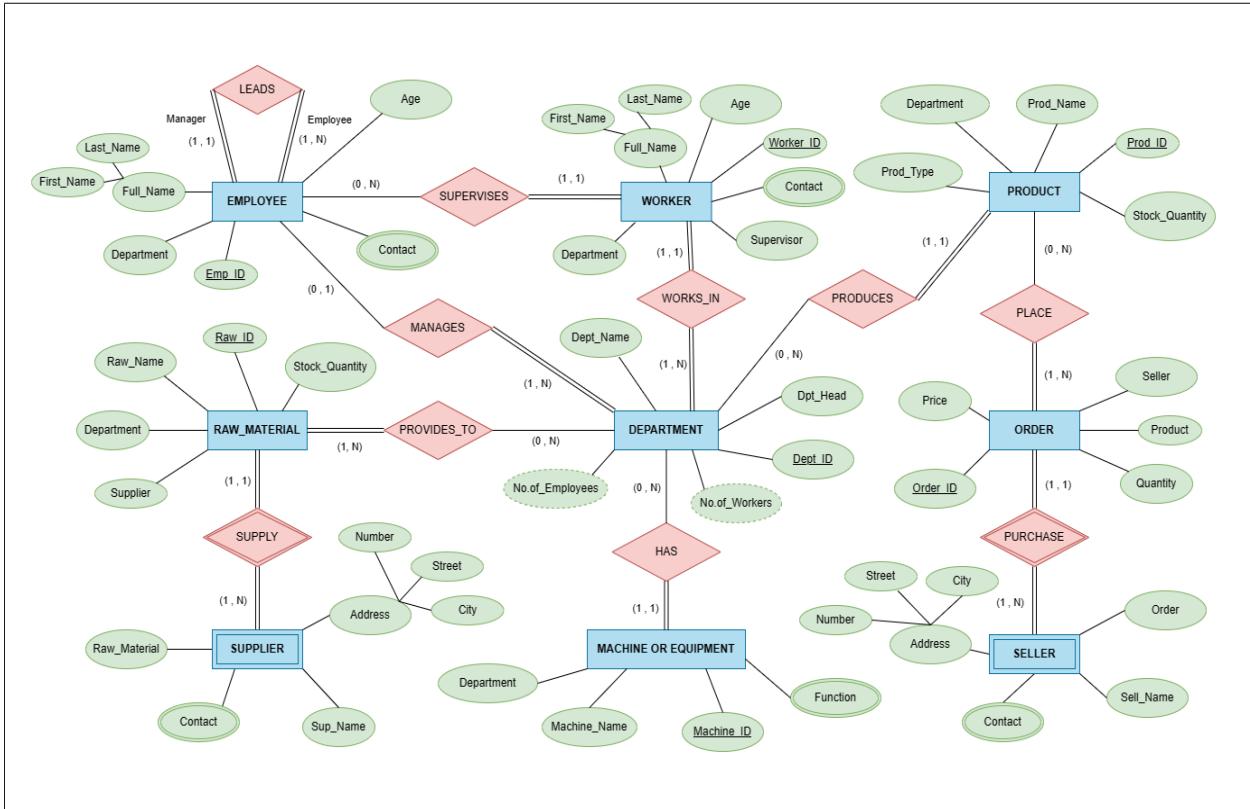


Figure 1: ER diagram of the garment factory database system

In the ER diagram, relationship is association among two or more entities. In every relationship, no of entities that participate in that relationship is called degree of relationship. In this garment factory management system database, we consider all the scenarios which are involved in the garment to get the relationships.

List of Relationships

- A manager can supervise multiple employees. The cardinality of this relationship is (1, N), which means that a manager can supervise one or more employees, but each employee can only be supervised by one manager.
- One employee can manage one department and department can have more than one employee. One of them is the head of the particular department.
- One raw material can supply by one supplier and one supplier can supply more than one raw material.
- One product can get minimum zero order and maximum many orders.

- All the departments have employees, but all the employees don't need to have departments.
- Workers and departments are in total participation in WORKS_IN relationship. It means all the workers have departments and all the departments have workers.
- All the machine/equipment has departments, and all the departments don't need to have machine/equipment.

These are some of the relationships in our garment factory management system database.

3. CHAPTER IV – IMPLEMENTATION

3.1 CREATING DATABASE

The screenshot shows the MySQL Workbench interface. In the central query editor window, the following SQL script is displayed:

```
1 ## CREATING DATABASE ##
2
3 4 • CREATE database factory;
5 • USE factory;
6
7 ## CREATING TABLES ##
8
9 • CREATE table department(
10     Dept_ID varchar(10) not null,
11     Dept_name varchar(50),
12     Dept_head varchar(10),
13     No_of_employee int,
14     No_of_workers int,
15     PRIMARY KEY(Dept_ID)
16 );
17
18 • CREATE table employee(
19     Emp_ID varchar(5) not null,
```

The 'Output' pane at the bottom shows the results of the execution:

#	Time	Action	Message	Duration / Fetch
1218	10:44:36	CREATE database factory	1 row(s) affected	0.046 sec
1219	10:44:36	USE factory	0 row(s) affected	0.000 sec

3.2 TABLE CREATION

Department Table

The screenshot shows the MySQL Workbench interface. In the central query editor window, the following SQL script is displayed:

```
1 ## CREATING DATABASE ##
2
3 4 • CREATE database factory;
5 • USE factory;
6
7 ## CREATING TABLES ##
8
9 • CREATE table department(
10     Dept_ID varchar(10) not null,
11     Dept_name varchar(50),
12     Dept_head varchar(10),
13     No_of_employee int,
14     No_of_workers int,
15     PRIMARY KEY(Dept_ID)
16 );
17
```

The 'Output' pane at the bottom shows the results of the execution:

#	Time	Action	Message	Duration / Fetch
1219	10:44:36	USE factory	0 row(s) affected	0.000 sec
1220	10:44:59	CREATE table department(Dept_ID varchar(10) not null, Dept_name varchar(50), Dept_head varchar(10)...)	0 row(s) affected	0.109 sec

Employee Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains the following SQL script:

```
17
18 • CREATE table employee(
19     Emp_ID varchar(5) not null,
20     First_name varchar(50),
21     Last_name varchar(50) not null,
22     Age int,
23     Department varchar(5),
24     Manager_ID varchar(10),
25     PRIMARY KEY(Emp_ID),
26     CONSTRAINT FK_Employee_DeptID foreign key(Department) references department(Dept_ID)
27     on delete set null on update cascade,
28     CONSTRAINT FK_Manager foreign key(Manager_ID) references employee(Emp_ID)
29     on delete set null on update cascade
30 );
31
32 • CREATE table worker(
33     Worker_ID varchar(5) not null,
```

The output pane shows two successful CREATE table statements:

#	Time	Action	Message	Duration / Fetch
1	1220 10:44:59	CREATE table department(Dept_ID varchar(10) not null, Dept_name varchar(50), Dept_head varchar(10)...)	0 row(s) affected	0.109 sec
2	1221 10:45:57	CREATE table employee(Emp_ID varchar(5) not null, First_name varchar(50), Last_name varchar(50) not...)	0 row(s) affected	0.188 sec

Worker Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains the following SQL script:

```
31
32 • CREATE table worker(
33     Worker_ID varchar(5) not null,
34     First_name varchar(50),
35     Last_name varchar(50) not null,
36     Age int,
37     Department varchar(5),
38     Supervisor varchar(5),
39     PRIMARY KEY(Worker_ID),
40     CONSTRAINT FK_Worker_DeptID foreign key(Department) references department(Dept_ID)
41     on delete set null on update cascade,
42     CONSTRAINT FK_Supervisor foreign key(Supervisor) references employee(Emp_ID)
43     on delete set null on update cascade
44 );
45
46 • CREATE table product(
47     Prod_ID varchar(10) not null,
```

The output pane shows two successful CREATE table statements:

#	Time	Action	Message	Duration / Fetch
1	1221 10:45:57	CREATE table employee(Emp_ID varchar(5) not null, First_name varchar(50), Last_name varchar(50) not...)	0 row(s) affected	0.188 sec
2	1222 10:46:33	CREATE table worker(Worker_ID varchar(5) not null, First_name varchar(50), Last_name varchar(50) not...)	0 row(s) affected	0.422 sec

Product Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the left sidebar, under the 'factory' schema, the 'Tables' section contains 'department', 'employee', and 'worker'. The 'Information' panel at the bottom left shows 'No object selected'. The main query editor area contains the following SQL code:

```
45
46 • CREATE table product(
47     Prod_ID varchar(10) not null,
48     Prod_name varchar(50) not null,
49     Prod_type varchar(50),
50     Department varchar(10),
51     Stock_quantity int,
52     PRIMARY KEY(Prod_ID),
53     CONSTRAINT FK_Product_DeptID foreign key(Department) references department(Dept_ID)
54         on delete cascade
55 );
56
57
58 • CREATE table orders(
59     Order_ID varchar(10) not null,
60     Quantity integer,
61     Price int,
62     PRIMARY KEY(Order_ID)
63 );
64
65 • CREATE table seller(
66     Sell_name varchar(50) not null,
67     Order_ID varchar(10) not null,
68     Address_No int
69 );
```

The 'Output' panel at the bottom shows the results of the execution:

#	Time	Action	Message	Duration / Fetch
1	1225 10:47:39	DROP TABLE Factory`.`product`	0 row(s) affected	0.047 sec
2	1226 10:47:47	CREATE table product(Prod_ID varchar(10) not null, Prod_name varchar(50) not null, Prod_type varchar(... 0 row(s) affected		0.141 sec

Orders Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the left sidebar, under the 'factory' schema, the 'Tables' section contains 'department', 'employee', and 'worker'. The 'Information' panel at the bottom left shows 'No object selected'. The main query editor area contains the following SQL code:

```
52
53     PRIMARY KEY(Prod_ID),
54     CONSTRAINT FK_Product_DeptID foreign key(Department) references department(Dept_ID)
55         on delete cascade
56
57
58 • CREATE table orders(
59     Order_ID varchar(10) not null,
60     Quantity integer,
61     Price int,
62     PRIMARY KEY(Order_ID)
63 );
64
65 • CREATE table seller(
66     Sell_name varchar(50) not null,
67     Order_ID varchar(10) not null,
68     Address_No int
69 );
```

The 'Output' panel at the bottom shows the results of the execution:

#	Time	Action	Message	Duration / Fetch
1	1226 10:47:47	CREATE table product(Prod_ID varchar(10) not null, Prod_name varchar(50) not null, Prod_type varchar(... 0 row(s) affected		0.141 sec
2	1227 10:48:29	CREATE table orders(Order_ID varchar(10) not null, Quantity integer, Price int, PRIMARY KEY(Order_ID)) 0 row(s) affected		0.110 sec

Seller Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the left sidebar, under the 'factory' schema, the 'Tables' section contains 'department', 'employee', and 'worker'. The 'Information' section shows 'No object selected'. The main pane displays the SQL code for creating the 'seller' table:

```
64 • CREATE table seller(
65     Sell_name varchar(50) not null,
66     Order_ID varchar(10) not null,
67     Address_No int,
68     Street varchar(50),
69     City varchar(50),
70     PRIMARY KEY(Order_ID, Sell_name),
71     CONSTRAINT FK_Seller_OrderID foreign key(Order_ID) references orders(Order_ID)
72         on delete cascade
73 );
74
75
76
77 • CREATE table raw_material(
78     Raw_ID varchar(10) not null,
79     Raw_name varchar(50) not null,
80     Stock_quantity int
```

The 'Output' pane at the bottom shows two log entries:

#	Time	Action	Message	Duration / Fetch
1227	10:48:29	CREATE table orders(Order_ID varchar(10) not null, Quantity integer, Price int, PRIMARY KEY(Order_ID))	0 row(s) affected	0.110 sec
1228	10:49:19	CREATE table seller(Sell_name varchar(50) not null, Order_ID varchar(10) not null, Address_No int, Street v...)	0 row(s) affected	0.125 sec

Raw Material

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the left sidebar, under the 'factory' schema, the 'Tables' section contains 'department', 'employee', and 'worker'. The 'Information' section shows 'No object selected'. The main pane displays the SQL code for creating the 'raw_material' table:

```
71 PRIMARY KEY(Order_ID, Sell_name),
72 CONSTRAINT FK_Seller_OrderID foreign key(Order_ID) references orders(Order_ID)
73 on delete cascade
74 );
75
76
77 • CREATE table raw_material(
78     Raw_ID varchar(10) not null,
79     Raw_name varchar(50) not null,
80     Stock_quantity int,
81     PRIMARY KEY(Raw_ID)
82 );
83
84 • CREATE table supplier(
85     Sup_name varchar(50) not null,
86     Raw_material varchar(10) not null,
87     Address_No int
```

The 'Output' pane at the bottom shows two log entries:

#	Time	Action	Message	Duration / Fetch
1228	10:49:19	CREATE table seller(Sell_name varchar(50) not null, Order_ID varchar(10) not null, Address_No int, Street v...)	0 row(s) affected	0.125 sec
1229	10:50:45	CREATE table raw_material(Raw_ID varchar(10) not null, Raw_name varchar(50) not null, Stock_quant...)	0 row(s) affected	0.297 sec

Supplier Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the SQL pane, the code for creating the 'supplier' table is visible:

```
84 • CREATE table supplier(
85     Sup_name varchar(50) not null,
86     Raw_material varchar(10) not null,
87     Address_No int,
88     Street varchar(50),
89     City varchar(50),
90     PRIMARY KEY(Raw_material, Sup_name),
91     CONSTRAINT FK_Supplier_DeptID foreign key(Raw_material) references raw_material(Raw_ID)
92     on delete cascade
93 );
94
95 • CREATE table machine(
96     Machine_ID varchar(10) not null,
97     Machine_name varchar(50) not null,
98     Machine_function varchar(100) not null,
99     Department varchar(10)
```

The 'Output' pane shows two log entries indicating successful table creation:

#	Time	Action	Message	Duration / Fetch
1229	10:50:45	CREATE table raw_material(Raw_ID varchar(10) not null, Raw_name varchar(50) not null, Stock_quantity int, Unit_Price decimal(10,2), Department varchar(10))	0 row(s) affected	0.297 sec
1230	10:51:24	CREATE table supplier(Sup_name varchar(50) not null, Raw_material varchar(10) not null, Address_No int, Street varchar(50), City varchar(50), PRIMARY KEY(Raw_material, Sup_name), CONSTRAINT FK_Supplier_DeptID foreign key(Raw_material) references raw_material(Raw_ID) on delete cascade)	0 row(s) affected	0.110 sec

Machine Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the SQL pane, the code for creating the 'machine' table is visible:

```
90 PRIMARY KEY(Raw_material, Sup_name),
91 CONSTRAINT FK_Supplier_DeptID foreign key(Raw_material) references raw_material(Raw_ID)
92 on delete cascade
93 );
94
95 • CREATE table machine(
96     Machine_ID varchar(10) not null,
97     Machine_name varchar(50) not null,
98     Machine_function varchar(100) not null,
99     Department varchar(10),
100 PRIMARY KEY(Machine_ID),
101 CONSTRAINT FK_DeptID foreign key(Department) references department(Dept_ID)
102 on delete cascade on update cascade
103 );
104
105
106 • CREATE table new_department(/
```

The 'Output' pane shows two log entries indicating successful table creation:

#	Time	Action	Message	Duration / Fetch
1230	10:51:24	CREATE table supplier(Sup_name varchar(50) not null, Raw_material varchar(10) not null, Address_No int, Street varchar(50), City varchar(50), PRIMARY KEY(Raw_material, Sup_name), CONSTRAINT FK_Supplier_DeptID foreign key(Raw_material) references raw_material(Raw_ID) on delete cascade)	0 row(s) affected	0.110 sec
1231	10:51:48	CREATE table machine(Machine_ID varchar(10) not null, Machine_name varchar(50) not null, Machine_function varchar(100) not null, Department varchar(10), PRIMARY KEY(Machine_ID), CONSTRAINT FK_DeptID foreign key(Department) references department(Dept_ID) on delete cascade on update cascade)	0 row(s) affected	0.156 sec

Raw-Department Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the code editor, lines 106 to 120 define the 'raw_department' table:

```
106 • CREATE table raw_department(
107     Raw_ID varchar(10) not null,
108     Dept_ID varchar(10) not null,
109     PRIMARY KEY(Raw_ID, Dept_ID),
110     CONSTRAINT fk_Raw_Dept FOREIGN KEY(Raw_ID) references raw_material(Raw_ID)
111     on delete cascade,
112     CONSTRAINT fk_Dept_Raw FOREIGN KEY(Dept_ID) references department(Dept_ID)
113     on delete cascade
114 );
115
116 • CREATE table product_order(
117     Prod_ID varchar(10) not null,
118     Order_ID varchar(10) not null,
119     PRIMARY KEY(Prod_ID, Order_ID),
120     CONSTRAINT fk_Prod_Order FOREIGN KEY(Prod_ID) references product(Prod_ID)
121     on delete cascade,
122     CONSTRAINT fk_Order_Prod FOREIGN KEY(Order_ID) references orders(Order_ID)
123     on delete cascade
124 );
125
126 • CREATE table employee_contact(
127     Emp_ID varchar(10) not null
```

The 'Output' pane shows two successful queries:

#	Time	Action	Message	Duration / Fetch
1231	10:51:48	CREATE table machine(Machine_ID varchar(10) not null, Machine_name varchar(50) not null, Machine_desc text, Machine_Status enum('Active', 'Inactive'))	0 row(s) affected	0.156 sec
1232	10:52:17	CREATE table raw_department(Raw_ID varchar(10) not null, Dept_ID varchar(10) not null, PRIMARY KEY(Raw_ID, Dept_ID), CONSTRAINT fk_Raw_Dept FOREIGN KEY(Raw_ID) references raw_material(Raw_ID) on delete cascade, CONSTRAINT fk_Dept_Raw FOREIGN KEY(Dept_ID) references department(Dept_ID) on delete cascade)	0 row(s) affected	0.125 sec

Product-Order Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the code editor, lines 116 to 126 define the 'product_order' table:

```
111
112
113
114 );
115
116 • CREATE table product_order(
117     Prod_ID varchar(10) not null,
118     Order_ID varchar(10) not null,
119     PRIMARY KEY(Prod_ID, Order_ID),
120     CONSTRAINT fk_Prod_Order FOREIGN KEY(Prod_ID) references product(Prod_ID)
121     on delete cascade,
122     CONSTRAINT fk_Order_Prod FOREIGN KEY(Order_ID) references orders(Order_ID)
123     on delete cascade
124 );
125
126 • CREATE table employee_contact(
127     Emp_ID varchar(10) not null
```

The 'Output' pane shows two successful queries:

#	Time	Action	Message	Duration / Fetch
1232	10:52:17	CREATE table raw_department(Raw_ID varchar(10) not null, Dept_ID varchar(10) not null, PRIMARY KEY(Raw_ID, Dept_ID), CONSTRAINT fk_Raw_Dept FOREIGN KEY(Raw_ID) references raw_material(Raw_ID) on delete cascade, CONSTRAINT fk_Dept_Raw FOREIGN KEY(Dept_ID) references department(Dept_ID) on delete cascade)	0 row(s) affected	0.125 sec
1233	10:52:41	CREATE table product_order(Prod_ID varchar(10) not null, Order_ID varchar(10) not null, PRIMARY KEY(Prod_ID, Order_ID), CONSTRAINT fk_Prod_Order FOREIGN KEY(Prod_ID) references product(Prod_ID) on delete cascade, CONSTRAINT fk_Order_Prod FOREIGN KEY(Order_ID) references orders(Order_ID) on delete cascade)	0 row(s) affected	0.140 sec

Employee-Contact Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the code editor, lines 126 through 140 define the 'employee_contact' table:

```
126 • CREATE table employee_contact(
127     Emp_ID varchar(10) not null,
128     Contact varchar(10) not null,
129     PRIMARY KEY(Emp_ID, Contact),
130     CONSTRAINT fk_EmpID_Contact FOREIGN KEY(Emp_ID) references employee(Emp_ID)
131     on delete cascade
132 );
133
134 • CREATE table worker_contact(
135     Worker_ID varchar(10) not null,
136     Contact varchar(10) not null,
137     PRIMARY KEY(Worker_ID, Contact),
138     CONSTRAINT fk_WorkerID_Contact FOREIGN KEY(Worker_ID) references worker(Worker_ID)
139     on delete cascade
140 );
```

The 'Output' pane shows two successful CREATE table statements with 0 rows affected each. The status bar at the bottom right indicates the date and time as 9/11/2023 10:53 AM.

Worker-Contact Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the code editor, lines 134 through 147 define the 'worker_contact' and 'supplier_contact' tables:

```
132 );
133
134 • CREATE table worker_contact(
135     Worker_ID varchar(10) not null,
136     Contact varchar(10) not null,
137     PRIMARY KEY(Worker_ID, Contact),
138     CONSTRAINT fk_WorkerID_Contact FOREIGN KEY(Worker_ID) references worker(Worker_ID)
139     on delete cascade
140 );
141
142 • CREATE table supplier_contact(
143     Sup_name varchar(50) not null,
144     Raw_material varchar(10) not null,
145     Contact varchar(10) not null,
146     PRIMARY KEY(Raw_material,Contact),
147     CONSTRAINT fk_RawMaterial_Contact FOREIGN KEY(Raw_material) references supplier(Raw_material)
148     on delete cascade
149 );
```

The 'Output' pane shows two successful CREATE table statements with 0 rows affected each. The status bar at the bottom right indicates the date and time as 9/11/2023 10:53 AM.

Supplier-Contact Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the code editor, lines 142 through 154 define the 'supplier_contact' table:

```
139     on delete cascade
140 );
141
142 • CREATE table supplier_contact(
143     Sup_name varchar(50) not null,
144     Raw_material varchar(10) not null,
145     Contact varchar(10) not null,
146     PRIMARY KEY(Raw_material,Contact),
147     CONSTRAINT fk_RawMaterial_Contact FOREIGN KEY(Raw_material) references supplier(Raw_material)
148     on delete cascade
149 );
150
151 • CREATE table seller_contact(
152     Sell_name varchar(50) not null,
153     Order_ID varchar(10) not null,
154     Contact varchar(10) not null,
155     CONSTRAINT fk_OrderID_Contact FOREIGN KEY(Order_ID) references contact(Contact)
```

The 'Output' pane shows two successful CREATE table statements with their execution times and message details.

#	Time	Action	Message	Duration / Fetch
1235	10:53:33	CREATE table worker_contact(Worker_ID varchar(10) not null, Contact varchar(10) not null, PRIMARY KE... 0 row(s) affected		0.125 sec
1236	10:54:04	CREATE table supplier_contact(Sup_name varchar(50) not null, Raw_material varchar(10) not null, Contact... 0 row(s) affected		0.110 sec

Seller-Contact Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. In the code editor, lines 142 through 154 define the 'seller_contact' table:

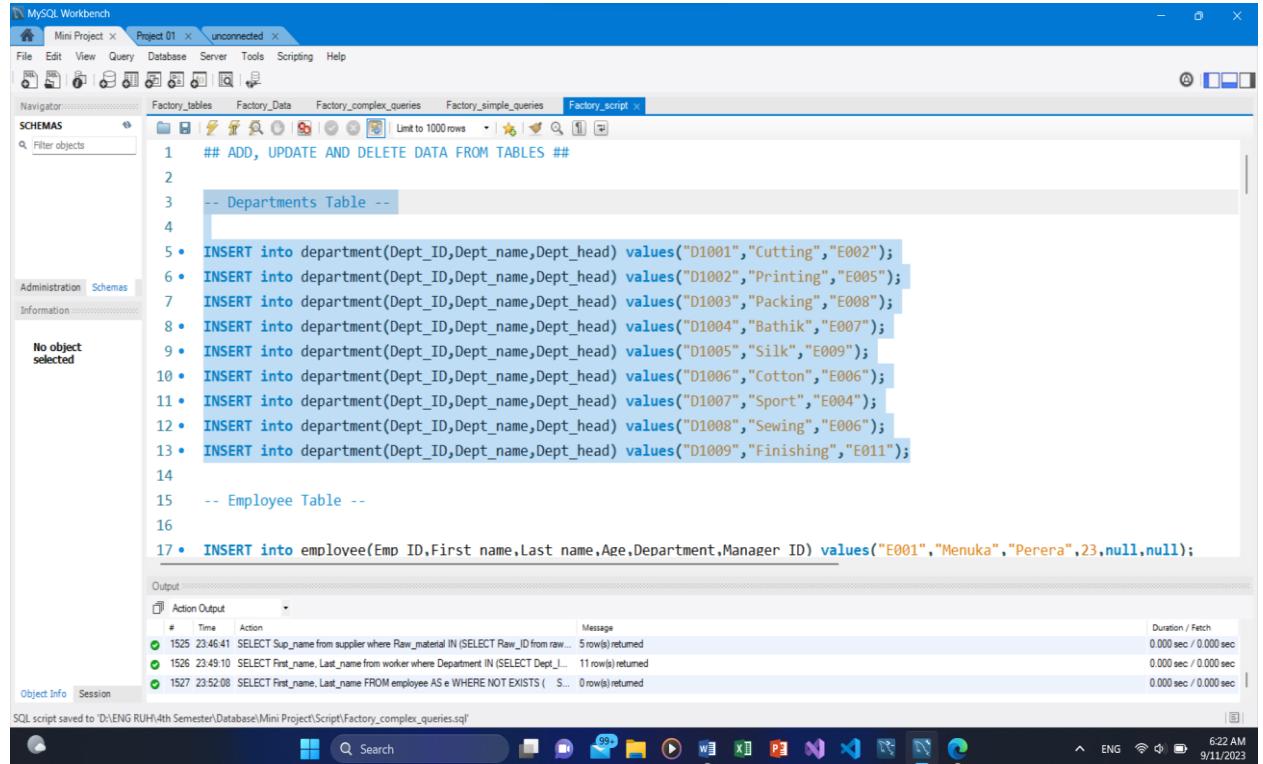
```
139     on delete cascade
140 );
141
142 • CREATE table supplier_contact(
143     Sup_name varchar(50) not null,
144     Raw_material varchar(10) not null,
145     Contact varchar(10) not null,
146     PRIMARY KEY(Raw_material,Contact),
147     CONSTRAINT fk_RawMaterial_Contact FOREIGN KEY(Raw_material) references supplier(Raw_material)
148     on delete cascade
149 );
150
151 • CREATE table seller_contact(
152     Sell_name varchar(50) not null,
153     Order_ID varchar(10) not null,
154     Contact varchar(10) not null,
155     CONSTRAINT fk_OrderID_Contact FOREIGN KEY(Order_ID) references contact(Contact)
```

The 'Output' pane shows two successful CREATE table statements with their execution times and message details.

#	Time	Action	Message	Duration / Fetch
1235	10:53:33	CREATE table worker_contact(Worker_ID varchar(10) not null, Contact varchar(10) not null, PRIMARY KE... 0 row(s) affected		0.125 sec
1236	10:54:04	CREATE table supplier_contact(Sup_name varchar(50) not null, Raw_material varchar(10) not null, Contact... 0 row(s) affected		0.110 sec

3.3 INSERTING TUPLES

Department Table



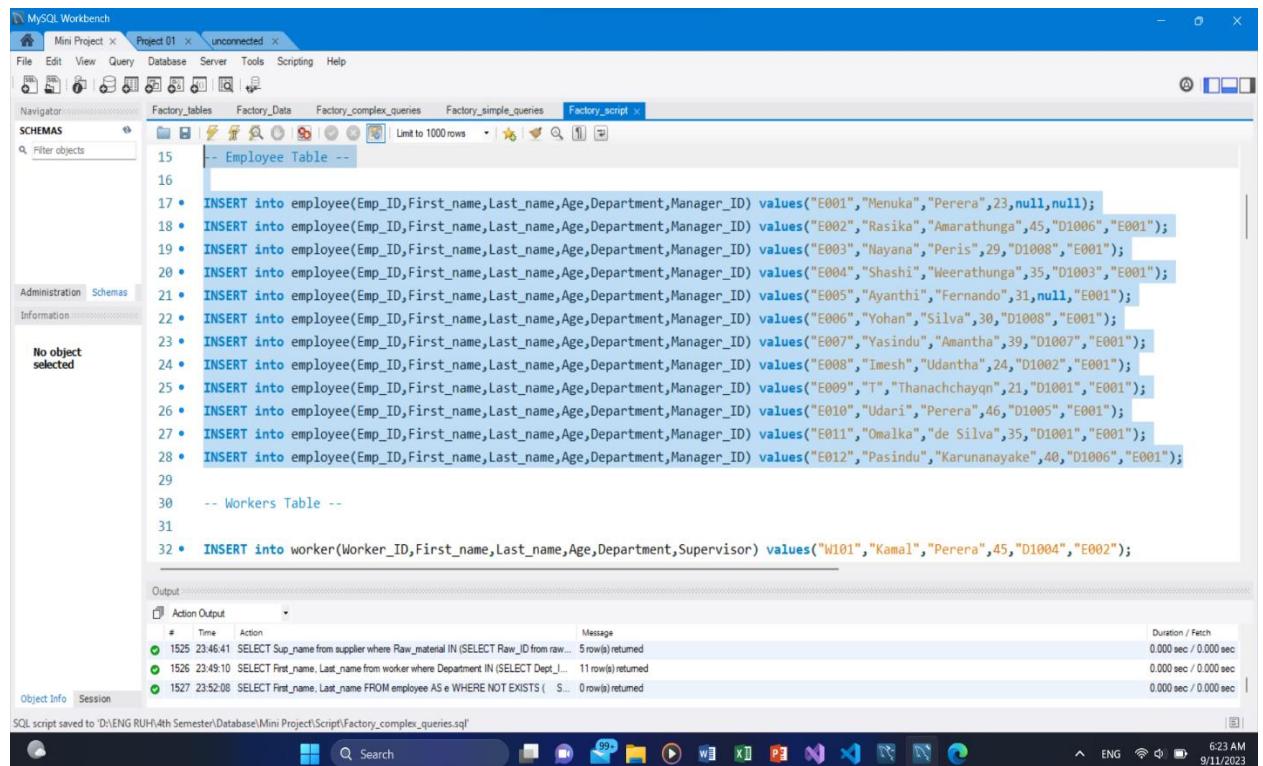
The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code area contains a script for inserting data into the 'department' table:

```
1 ## ADD, UPDATE AND DELETE DATA FROM TABLES #
2
3 -- Departments Table --
4
5 • INSERT into department(Dept_ID,Dept_name,Dept_head) values("D1001","Cutting","E002");
6 • INSERT into department(Dept_ID,Dept_name,Dept_head) values("D1002","Printing","E005");
7 INSERT into department(Dept_ID,Dept_name,Dept_head) values("D1003","Packing","E008");
8 • INSERT into department(Dept_ID,Dept_name,Dept_head) values("D1004","Bathik","E007");
9 • INSERT into department(Dept_ID,Dept_name,Dept_head) values("D1005","Silk","E009");
10 • INSERT into department(Dept_ID,Dept_name,Dept_head) values("D1006","Cotton","E006");
11 • INSERT into department(Dept_ID,Dept_name,Dept_head) values("D1007","Sport","E004");
12 • INSERT into department(Dept_ID,Dept_name,Dept_head) values("D1008","Sewing","E006");
13 • INSERT into department(Dept_ID,Dept_name,Dept_head) values("D1009","Finishing","E011");
14
15 -- Employee Table --
16
17 • INSERT into employee(Emp_ID,First name,Last name,Age,Department,Manager ID) values("E001","Menuka","Perera",23,null,null);
```

The output pane shows the execution results for the first three queries:

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...)	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_ID...)	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Employee Table



The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code area contains a script for inserting data into the 'employee' table:

```
15 -- Employee Table --
16
17 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E001","Menuka","Perera",23,null,null);
18 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E002","Rasika","Amarathunga",45,"D1006","E001");
19 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E003","Nayana","Peris",29,"D1008","E001");
20 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E004","Shashi","Weerathunga",35,"D1003","E001");
21 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E005","Ayanthi","Fernando",31,null,"E001");
22 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E006","Yohan","Silva",30,"D1008","E001");
23 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E007","Yasindu","Amantha",39,"D1007","E001");
24 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E008","Imesh","Udantha",24,"D1002","E001");
25 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E009","T","Thanachchayqn",21,"D1001","E001");
26 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E010","Udari","Perera",46,"D1005","E001");
27 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E011","Omalka","de Silva",35,"D1001","E001");
28 • INSERT into employee(Emp_ID,First_name,Last_name,Age,Department,Manager_ID) values("E012","Pasindu","Karunanayake",40,"D1006","E001");
29
30 -- Workers Table --
31
32 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W101","Kamal","Perera",45,"D1004","E002");
```

The output pane shows the execution results for the first three queries:

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...)	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_ID...)	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Worker Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code area contains SQL insert statements for the 'worker' table, grouped by comments: '-- Workers Table --' and '-- Products Table --'. The output pane shows three log entries related to the script execution.

```

29
30  -- Workers Table --
31
32 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W101","Kamal","Perera",45,"D1004","E002");
33 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W102","Nanda","Amarathunga",20,"D1002","E008");
34 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W103","Siril","Peris",24,"D1005","E004");
35 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W104","Asela","Weerathunga",37,"D1008","E003");
36 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W105","Oshan","Ferna",29,"D1003","E004");
37 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W106","Udari","Silva",34,"D1006","E005");
38 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W107","Umatha","Yehan",35,"D1006","E005");
39 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W108","Sandaru","Silva",30,"D1002","E006");
40 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W109","Piyum","Niroshan",25,"D1001","E005");
41 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W110","Udari","Maduwanthika",41,"D1006","E008");
42 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W111","Lahiru","Kamal",56,"D1007","E006");
43 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W112","Supun","Perera",27,"D1007","E008");
44
45  -- Products Table --
46

```

Output:

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_...	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Product Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code area contains SQL insert statements for the 'product' table, grouped by comments: '-- Products Table --' and '-- Raw Materials Table --'. The output pane shows three log entries related to the script execution.

```

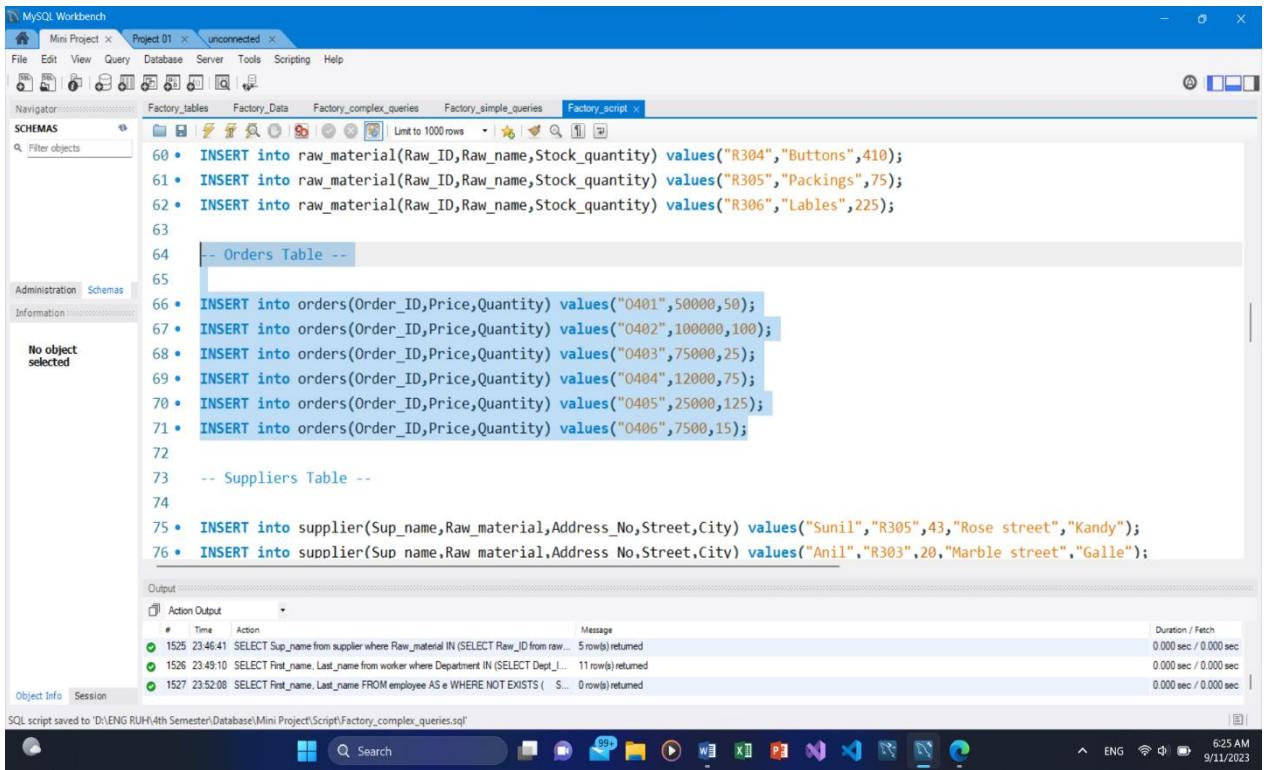
43 • INSERT into worker(Worker_ID,First_name,Last_name,Age,Department,Supervisor) values("W112","Supun","Perera",27,"D1007","E008");
44
45  -- Products Table --
46
47 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P201","Blouse","Ladies","D1007",400);
48 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P202","Socks","Kids","D1004",150);
49 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P203","Geans","Ladies","D1006",275);
50 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P204","Geans","Mens","D1005",175);
51 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P205","Shirt","Mens","D1005",325);
52 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P206","Frock","Ladies","D1004",200);
53 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P207","Blouse","Ladies","D1006",225);
54
55  -- Raw Materials Table --
56
57 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R301","Fabric rolls",200);
58 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R302","Threads",325);
59 • INSERT into raw material(Raw ID,Raw name,Stock quantity) values("R303","Inks",270);

```

Output:

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_...	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Orders Table



The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains SQL scripts for creating tables and inserting data. The 'Orders Table' section starts at line 64:

```

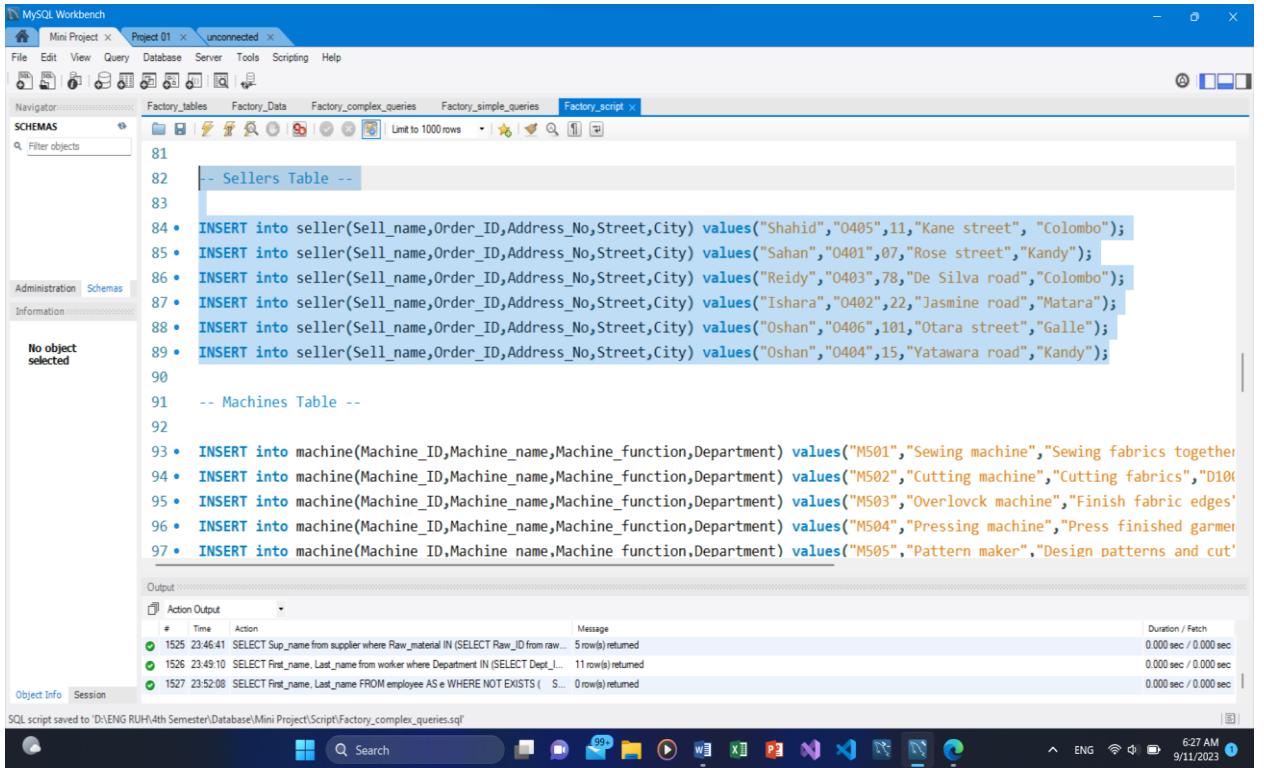
64 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R304","Buttons",410);
65 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R305","Packings",75);
66 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R306","Labels",225);
67
68 -- Orders Table --
69
70 • INSERT into orders(Order_ID,Price,Quantity) values("0401",50000,50);
71 • INSERT into orders(Order_ID,Price,Quantity) values("0402",100000,100);
72 • INSERT into orders(Order_ID,Price,Quantity) values("0403",75000,25);
73 • INSERT into orders(Order_ID,Price,Quantity) values("0404",12000,75);
74 • INSERT into orders(Order_ID,Price,Quantity) values("0405",25000,125);
75 • INSERT into orders(Order_ID,Price,Quantity) values("0406",7500,15);
76
77 -- Suppliers Table --
78
79 • INSERT into supplier(Sup_name,Raw_material,Address_No,Street,City) values("Sunil","R305",43,"Rose street","Kandy");
80 • INSERT into supplier(Sup_name,Raw material,Address No,Street,City) values("Anil","R303",20,"Marble street","Galle");

```

The 'Output' pane shows the execution log with three successful SELECT statements:

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_I...	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Sellers Table



The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains SQL scripts for creating tables and inserting data. The 'Sellers Table' section starts at line 81:

```

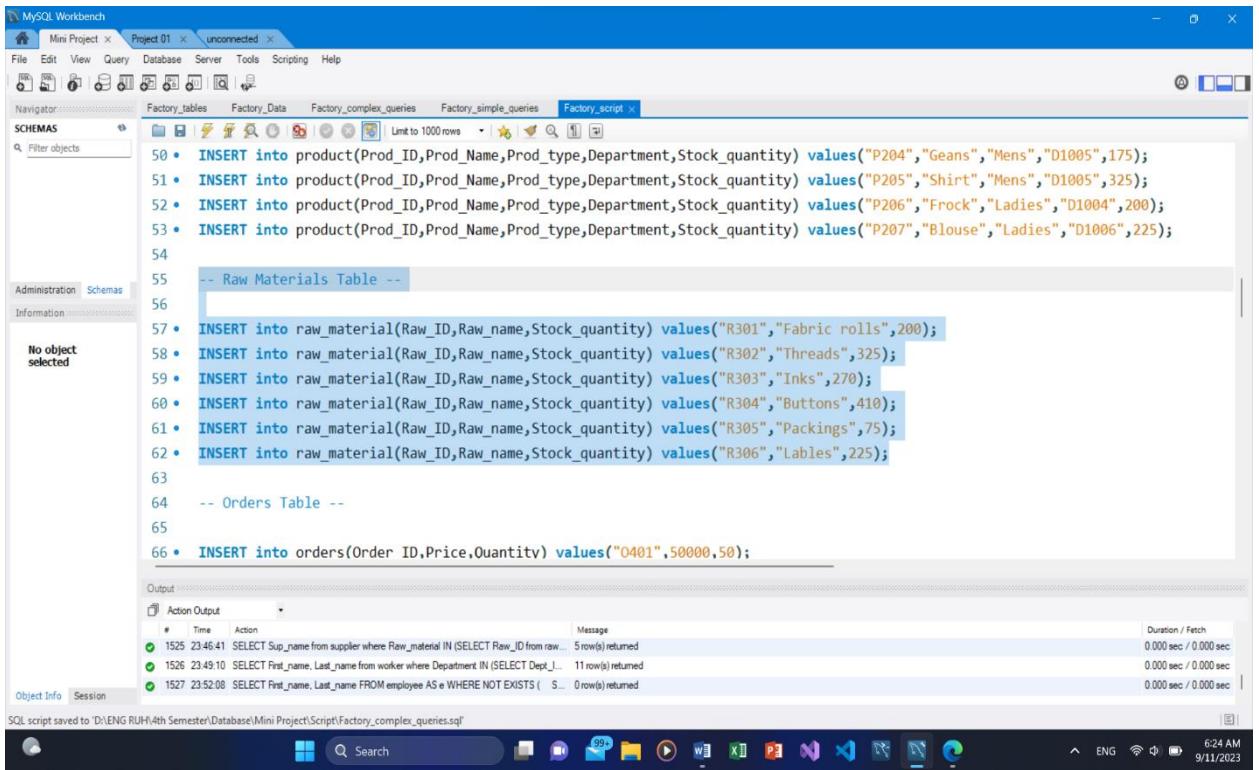
81
82 -- Sellers Table --
83
84 • INSERT into seller(Sell_name,Order_ID,Address_No,Street,City) values("Shahid","0405",11,"Kane street", "Colombo");
85 • INSERT into seller(Sell_name,Order_ID,Address_No,Street,City) values("Sahan","0401",07,"Rose street", "Kandy");
86 • INSERT into seller(Sell_name,Order_ID,Address_No,Street,City) values("Reidy","0403",78,"De Silva road", "Colombo");
87 • INSERT into seller(Sell_name,Order_ID,Address_No,Street,City) values("Ishara","0402",22,"Jasmine road", "Matara");
88 • INSERT into seller(Sell_name,Order_ID,Address_No,Street,City) values("Oshan","0406",101,"Otara street", "Galle");
89 • INSERT into seller(Sell_name,Order_ID,Address_No,Street,City) values("Oshan","0404",15,"Yatawara road", "Kandy");
90
91 -- Machines Table --
92
93 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M501","Sewing machine","Sewing fabrics together");
94 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M502","Cutting machine","Cutting fabrics","D10");
95 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M503","Overlovck machine","Finish fabric edges");
96 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M504","Pressing machine","Press finished garment");
97 • INSERT into machine(Machine ID,Machine name,Machine function,Department) values("M505","Pattern maker","Design patterns and cut"

```

The 'Output' pane shows the execution log with three successful SELECT statements:

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_I...	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Raw Materials



The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code area contains SQL queries for inserting data into the 'product' and 'raw_material' tables. The 'Output' pane shows three log entries from the session.

```

50 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P204","Gears","Mens","D1005",175);
51 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P205","Shirt","Mens","D1005",325);
52 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P206","Frock","Ladies","D1004",200);
53 • INSERT into product(Prod_ID,Prod_Name,Prod_type,Department,Stock_quantity) values("P207","Blouse","Ladies","D1006",225);
54
55 -- Raw Materials Table --
56
57 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R301","Fabric rolls",200);
58 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R302","Threads",325);
59 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R303","Inks",270);
60 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R304","Buttons",410);
61 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R305","Packings",75);
62 • INSERT into raw_material(Raw_ID,Raw_name,Stock_quantity) values("R306","Labels",225);
63
64 -- Orders Table --
65
66 • INSERT into orders(Order_ID,Price,Quantity) values("O401",50000,50);

```

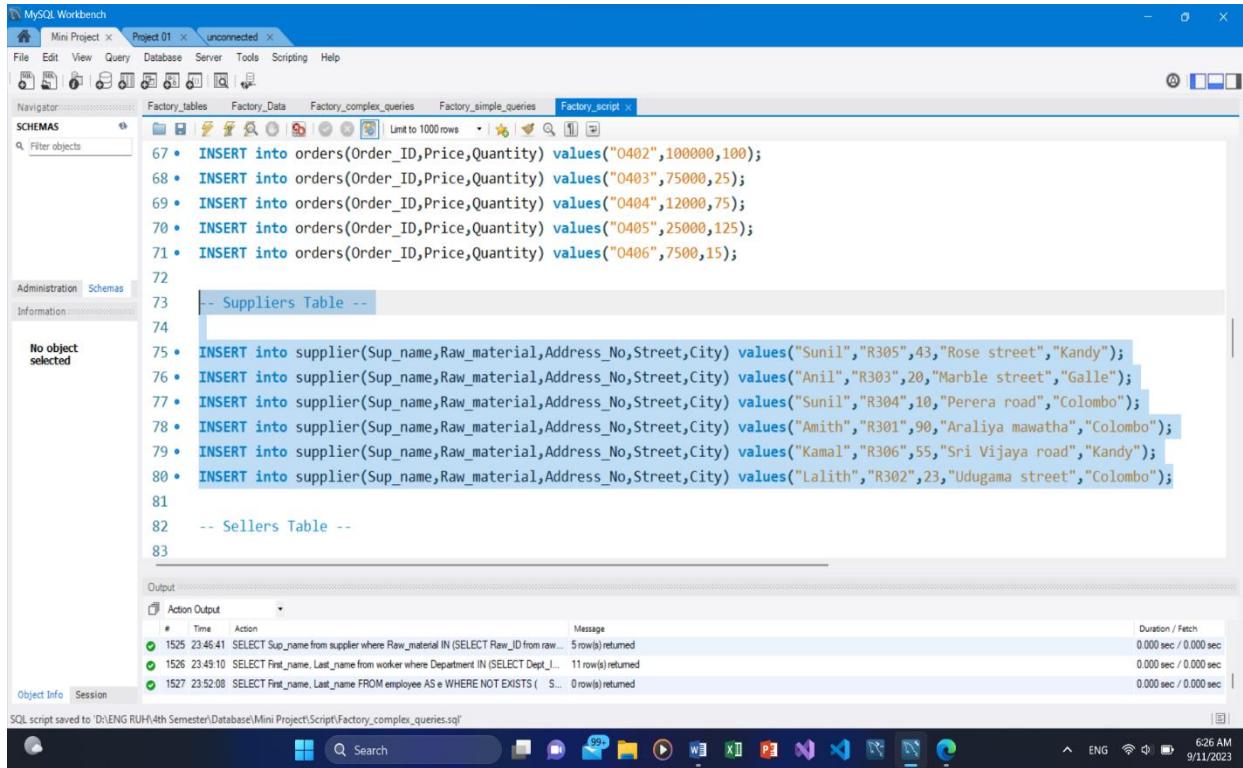
Object Info Session

SQL script saved to 'D:\ENG RUH\4th Semester\Database\Mini Project\Script\Factory_complex_queries.sql'

Output

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...)	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_I...	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Supplier Table



The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code area contains SQL queries for inserting data into the 'orders' and 'supplier' tables. The 'Output' pane shows three log entries from the session.

```

67 • INSERT into orders(Order_ID,Price,Quantity) values("O402",100000,100);
68 • INSERT into orders(Order_ID,Price,Quantity) values("O403",75000,25);
69 • INSERT into orders(Order_ID,Price,Quantity) values("O404",12000,75);
70 • INSERT into orders(Order_ID,Price,Quantity) values("O405",25000,125);
71 • INSERT into orders(Order_ID,Price,Quantity) values("O406",7500,15);
72
73 -- Suppliers Table --
74
75 • INSERT into supplier(Sup_name,Raw_material,Address_No,Street,City) values("Sunil","R305",43,"Rose street","Kandy");
76 • INSERT into supplier(Sup_name,Raw_material,Address_No,Street,City) values("Anil","R303",20,"Marble street","Galle");
77 • INSERT into supplier(Sup_name,Raw_material,Address_No,Street,City) values("Sunil","R304",10,"Perera road","Colombo");
78 • INSERT into supplier(Sup_name,Raw_material,Address_No,Street,City) values("Amith","R301",90,"Araliya mawatha","Colombo");
79 • INSERT into supplier(Sup_name,Raw_material,Address_No,Street,City) values("Kamal","R306",55,"Sri Vijaya road","Kandy");
80 • INSERT into supplier(Sup_name,Raw_material,Address_No,Street,City) values("Lalith","R302",23,"Udugama street","Colombo");
81
82 -- Sellers Table --
83

```

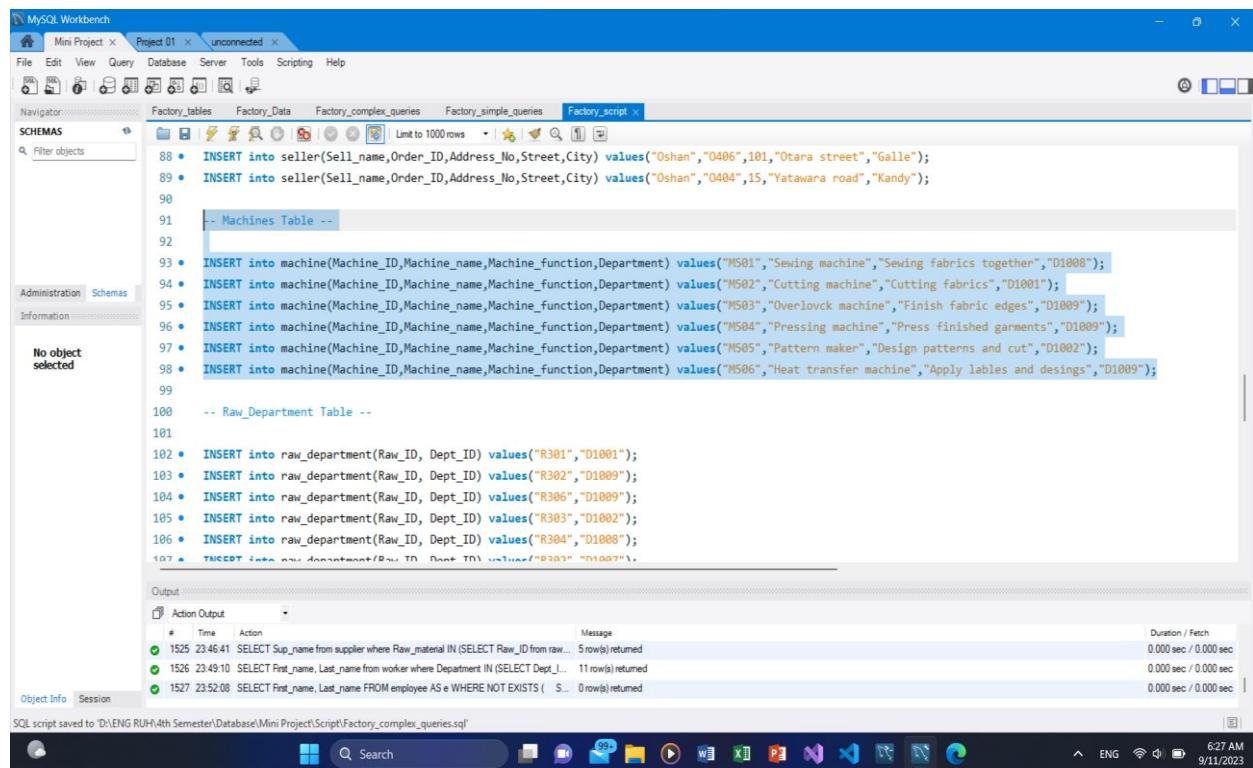
Object Info Session

SQL script saved to 'D:\ENG RUH\4th Semester\Database\Mini Project\Script\Factory_complex_queries.sql'

Output

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...)	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_I...	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Machines Table



The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains SQL scripts for populating the 'Machine' table and the 'Raw_Department' table. The 'Output' pane shows the execution log with three successful SELECT statements and their execution times.

```

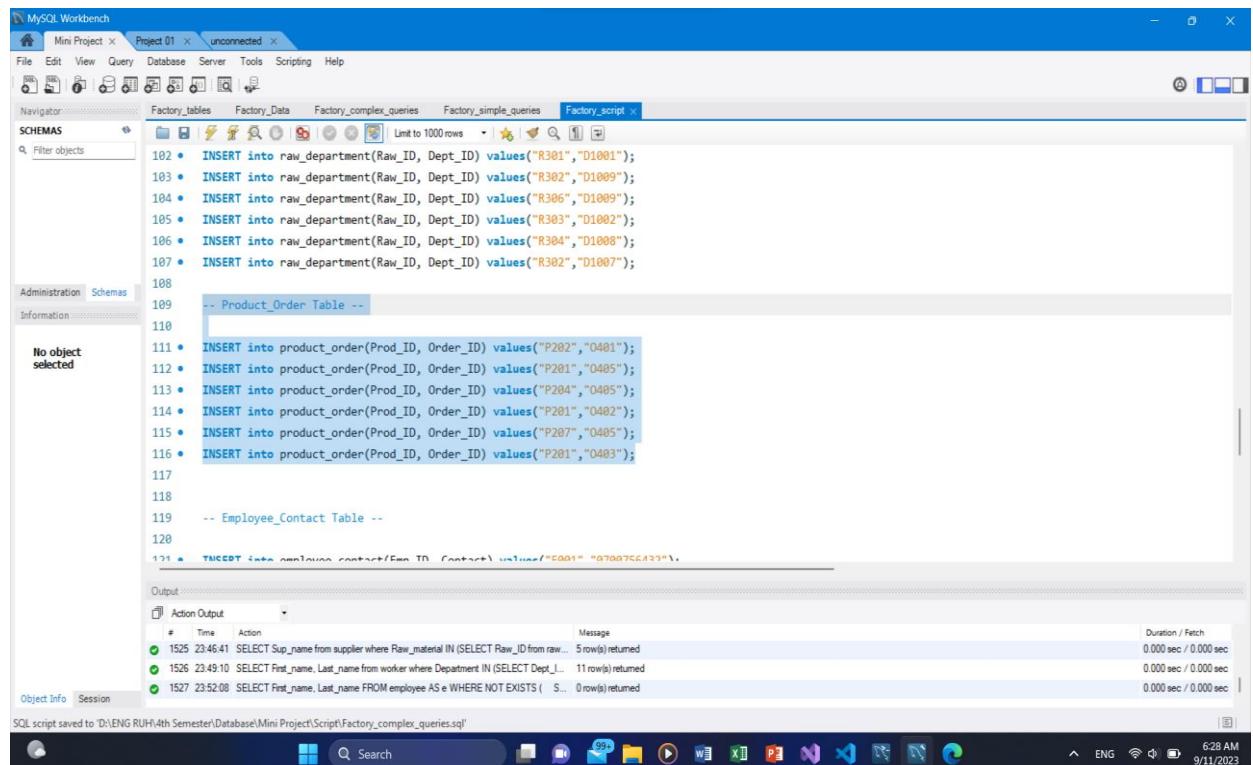
88 • INSERT into seller(Sell_name,Order_ID,Address_No,Street,City) values("Oshan","0406",181,"Otara street","Galle");
89 • INSERT into seller(Sell_name,Order_ID,Address_No,Street,City) values("Oshan","0404",15,"Yatawara road","Kandy");
90
91 -- Machines Table --
92
93 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M501","Sewing machine","Sewing fabrics together","D1008");
94 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M502","Cutting machine","Cutting fabrics","D1001");
95 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M503","Overlovck machine","Finish fabric edges","D1009");
96 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M504","Pressing machine","Press finished garments","D1009");
97 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M505","Pattern maker","Design patterns and cut","D1002");
98 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M506","Heat transfer machine","Apply lables and desings","D1009");
99
100 -- Raw_Department Table --
101
102 • INSERT into raw_department(Raw_ID, Dept_ID) values("R301","D1001");
103 • INSERT into raw_department(Raw_ID, Dept_ID) values("R302","D1009");
104 • INSERT into raw_department(Raw_ID, Dept_ID) values("R306","D1009");
105 • INSERT into raw_department(Raw_ID, Dept_ID) values("R303","D1002");
106 • INSERT into raw_department(Raw_ID, Dept_ID) values("R304","D1008");
107 • TRUNCATE TABLE raw_department/Raw_ID Dept_ID values("R307" "D1007");

Output:
Action Output
# Time Action Message Duration / Fetch
1525 23:46:41 SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw... 5 rows(s) returned 0.000 sec / 0.000 sec
1526 23:49:10 SELECT First_name, Last_name from worker where Department IN (SELECT Dept_... 11 row(s) returned 0.000 sec / 0.000 sec
1527 23:52:08 SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS ( S... 0 row(s) returned 0.000 sec / 0.000 sec

Object Info Session
SQL script saved to 'D:\ENG RUH\4th Semester\Database\Mini Project\Script\Factory_complex_queries.sql'

```

Product-Order Table



The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains SQL scripts for populating the 'raw_department' table and the 'Employee_Contact' table. The 'Output' pane shows the execution log with three successful SELECT statements and their execution times.

```

102 • INSERT into raw_department(Raw_ID, Dept_ID) values("R301","D1001");
103 • INSERT into raw_department(Raw_ID, Dept_ID) values("R302","D1009");
104 • INSERT into raw_department(Raw_ID, Dept_ID) values("R306","D1009");
105 • INSERT into raw_department(Raw_ID, Dept_ID) values("R303","D1002");
106 • INSERT into raw_department(Raw_ID, Dept_ID) values("R304","D1008");
107 • INSERT into raw_department(Raw_ID, Dept_ID) values("R302","D1007");
108
109 -- Product_Order Table --
110
111 • INSERT into product_order(Prod_ID, Order_ID) values("P202","0401");
112 • INSERT into product_order(Prod_ID, Order_ID) values("P201","0405");
113 • INSERT into product_order(Prod_ID, Order_ID) values("P204","0405");
114 • INSERT into product_order(Prod_ID, Order_ID) values("P201","0402");
115 • INSERT into product_order(Prod_ID, Order_ID) values("P207","0405");
116 • INSERT into product_order(Prod_ID, Order_ID) values("P201","0403");
117
118
119 -- Employee_Contact Table --
120
121 • TRUNCATE TABLE employee_contact/Emp_ID Contact values("E001" "0700755837");

Output:
Action Output
# Time Action Message Duration / Fetch
1525 23:46:41 SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw... 5 rows(s) returned 0.000 sec / 0.000 sec
1526 23:49:10 SELECT First_name, Last_name from worker where Department IN (SELECT Dept_... 11 row(s) returned 0.000 sec / 0.000 sec
1527 23:52:08 SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS ( S... 0 row(s) returned 0.000 sec / 0.000 sec

Object Info Session
SQL script saved to 'D:\ENG RUH\4th Semester\Database\Mini Project\Script\Factory_complex_queries.sql'

```

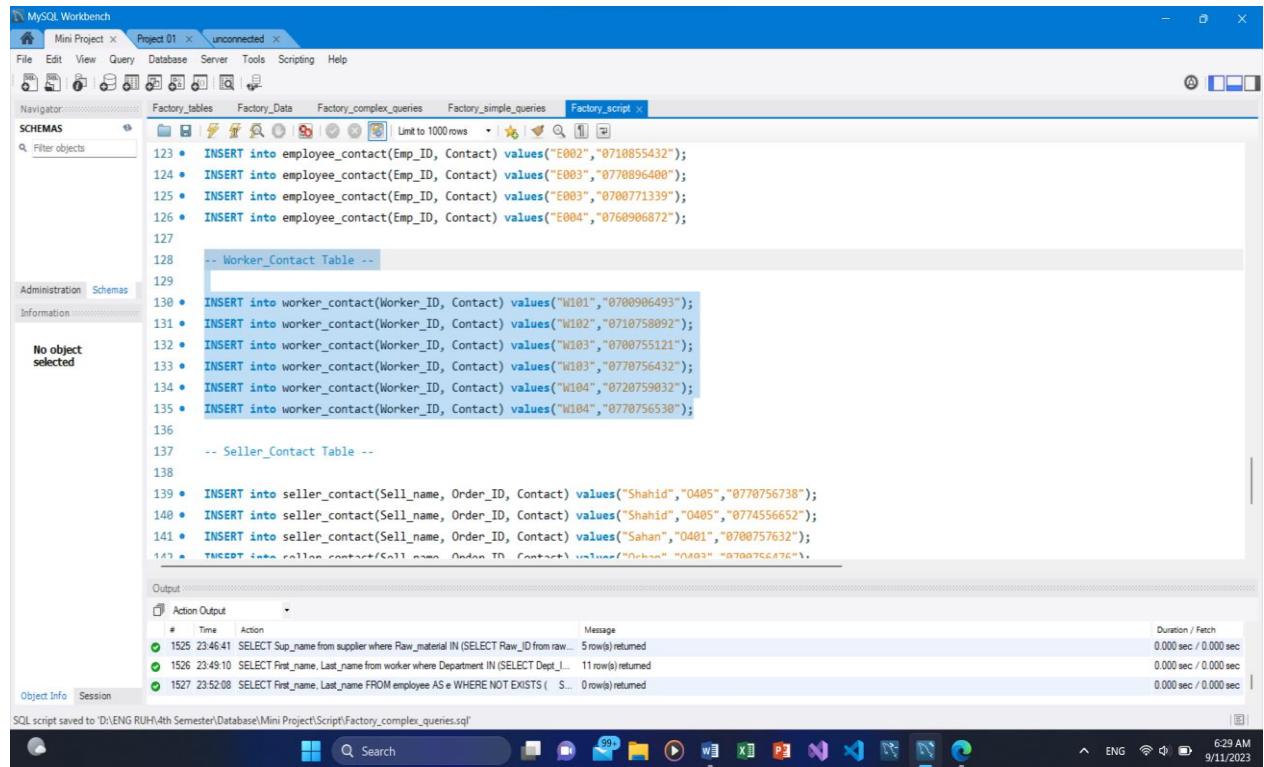
Raw-Department Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code pane contains SQL scripts for creating tables and inserting data into the 'Machine', 'raw_department', 'product_order', and 'Employee_Contact' tables. The output pane shows the execution log with three successful SELECT statements and their execution times.

```

95 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M503","Overlovck machine","Finish fabric edges","D1009");
96 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M504","Pressing machine","Press finished garments","D1009");
97 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M505","Pattern maker","Design patterns and cut","D1002");
98 • INSERT into machine(Machine_ID,Machine_name,Machine_function,Department) values("M506","Heat transfer machine","Apply lables and desings","D1009");
99
100 -- Raw_Department Table --
101
102 • INSERT into raw_department(Raw_ID, Dept_ID) values("R301","D1001");
103 • INSERT into raw_department(Raw_ID, Dept_ID) values("R302","D1009");
104 • INSERT into raw_department(Raw_ID, Dept_ID) values("R306","D1009");
105 • INSERT into raw_department(Raw_ID, Dept_ID) values("R303","D1002");
106 • INSERT into raw_department(Raw_ID, Dept_ID) values("R304","D1008");
107 • INSERT into raw_department(Raw_ID, Dept_ID) values("R302","D1007");
108
109 -- Product_Order Table --
110
111 • INSERT into product_order(Prod_ID, Order_ID) values("P202","0401");
112 • INSERT into product_order(Prod_ID, Order_ID) values("P201","0405");
113 • INSERT into product_order(Prod_ID, Order_ID) values("P204","0405");
114 • TRUNCATE TABLE product_order
115
116
117
118
119 -- Employee_Contact Table --
120
121 • INSERT into employee_contact(Emp_ID, Contact) values("E001","0700756432");
122 • INSERT into employee_contact(Emp_ID, Contact) values("E001","0776758742");
123 • INSERT into employee_contact(Emp_ID, Contact) values("E001","0710855432");
124 • INSERT into employee_contact(Emp_ID, Contact) values("E003","0770896408");
125 • INSERT into employee_contact(Emp_ID, Contact) values("E003","0700771339");
126 • INSERT into employee_contact(Emp_ID, Contact) values("E004","0760906872");
127
128 -- Worker_Contact Table --
129
130 • INSERT into worker_contact(Worker_ID, Contact) values("W101","0700906493");
131 • INSERT into worker_contact(Worker_ID, Contact) values("W102","0710758892");
132 • INSERT into worker_contact(Worker_ID, Contact) values("W103","0700755121");
133 • INSERT into worker_contact(Worker_ID, Contact) values("W103","0770756432");
134 • INSERT into worker_contact(Worker_ID, Contact) values("W104","0720759032");
135 • TRUNCATE TABLE worker_contact
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1197
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1397
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
21
```

Worker-Contact Table



The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code pane displays SQL queries for populating the 'worker_contact' table:

```

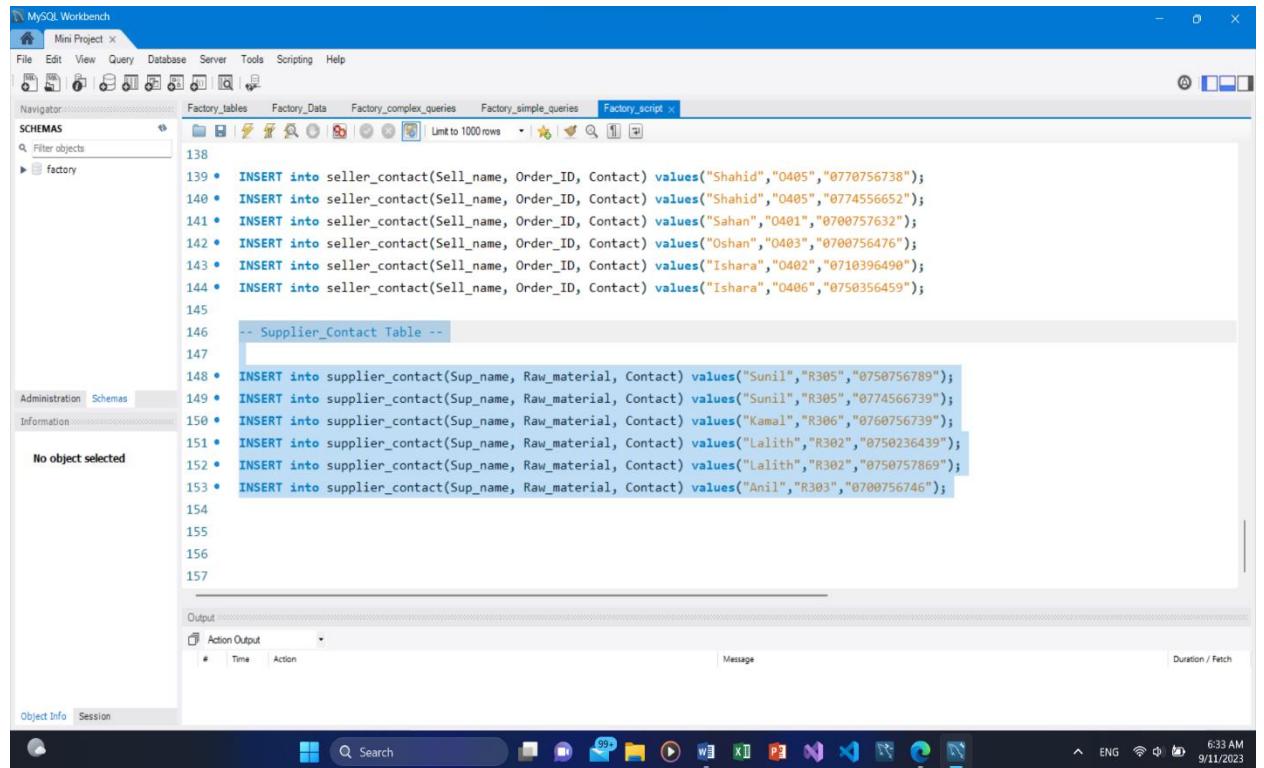
123 • INSERT into employee_contact(Emp_ID, Contact) values("E002","0710855432");
124 • INSERT into employee_contact(Emp_ID, Contact) values("E003","0770896400");
125 • INSERT into employee_contact(Emp_ID, Contact) values("E003","0708771339");
126 • INSERT into employee_contact(Emp_ID, Contact) values("E004","0760906872");
127
128 -- Worker_Contact Table --
129
130 • INSERT into worker_contact(Worker_ID, Contact) values("W101","0700906493");
131 • INSERT into worker_contact(Worker_ID, Contact) values("W102","0710758092");
132 • INSERT into worker_contact(Worker_ID, Contact) values("W103","0700755121");
133 • INSERT into worker_contact(Worker_ID, Contact) values("W103","0770756432");
134 • INSERT into worker_contact(Worker_ID, Contact) values("W104","0720759032");
135 • INSERT into worker_contact(Worker_ID, Contact) values("W104","0770756530");
136
137 -- Seller_Contact Table --
138
139 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Shahid","0405","0770756738");
140 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Shahid","0405","0774556652");
141 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Sahan","0401","0700757632");
142 • TRUNCATE table seller_contact(Sell_name, Order_ID, Contact) values("Oshan","0403","0700756476");
143

```

The output pane shows the execution results for the first three queries:

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...)	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_I...	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Supplier-Contact Table



The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code pane displays SQL queries for populating the 'supplier_contact' table:

```

138
139 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Shahid","0405","0770756738");
140 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Shahid","0405","0774556652");
141 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Sahan","0401","0700757632");
142 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Oshan","0403","0700756476");
143 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Ishara","0402","0710396490");
144 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Ishara","0406","0750356459");
145
146 -- Supplier_Contact Table --
147
148 • INSERT into supplier_contact(Sup_name, Raw_material, Contact) values("Sunil","R305","0750756789");
149 • INSERT into supplier_contact(Sup_name, Raw_material, Contact) values("Sunil","R305","0774566739");
150 • INSERT into supplier_contact(Sup_name, Raw_material, Contact) values("Kamal","R306","0760756739");
151 • INSERT into supplier_contact(Sup_name, Raw_material, Contact) values("Lalith","R302","0750236439");
152 • INSERT into supplier_contact(Sup_name, Raw_material, Contact) values("Lalith","R302","0750757869");
153 • INSERT into supplier_contact(Sup_name, Raw_material, Contact) values("Anil","R303","0700756746");
154
155
156
157

```

The output pane shows the execution results for the first three queries:

#	Time	Action	Message	Duration / Fetch
1525	23:46:41	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw...)	5 row(s) returned	0.000 sec / 0.000 sec
1526	23:49:10	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_I...	11 row(s) returned	0.000 sec / 0.000 sec
1527	23:52:08	SELECT First_name, Last_name FROM employee AS e WHERE NOT EXISTS (S...	0 row(s) returned	0.000 sec / 0.000 sec

Seller-Contact Table

The screenshot shows the MySQL Workbench interface with the following details:

- Project:** Mini Project > Project 01 > unconnected
- Tab:** Factory_script
- Code (SQL):**

```
130 • INSERT into worker_contact(Worker_ID, Contact) values("W101","0700906493");
131 • INSERT into worker_contact(Worker_ID, Contact) values("W102","0710758092");
132 • INSERT into worker_contact(Worker_ID, Contact) values("W103","0700755121");
133 • INSERT into worker_contact(Worker_ID, Contact) values("W103","0770756432");
134 • INSERT into worker_contact(Worker_ID, Contact) values("W104","0720759032");
135 • INSERT into worker_contact(Worker_ID, Contact) values("W104","0770756530");

136
137 -- Seller_Contact Table --
138
139 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Shahid","0405","0770756738");
140 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Shahid","0405","0774556652");
141 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Sahan","0481","0700757632");
142 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Oshan","0483","0700756476");
143 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Ishara","0402","0710396490");
144 • INSERT into seller_contact(Sell_name, Order_ID, Contact) values("Ishara","0406","0750356459");

145
146 -- Supplier_Contact Table --
147
148 • INSERT into supplier_contact(Sup_name, Raw_material, Contact) values("Sunil","R305","0750756789");
149 • TRUNCATE TABLE supplier_contact(Sup_name, Raw_material, Contact) values("Sunil","R305","0774556730");
```

- Output:** Shows log entries for the executed queries.
- Session Tab:** Session
- Bottom Bar:** Shows the Windows taskbar with various pinned icons and system status.

3.4 UPDATING AND DELETING FROM TABLES

Department Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains the following SQL statements:

```
302 • INSERT into supplier_contact(Sup_name, Raw_material, Contact) values("Anil","R303","0700756746");
303
304
305 • DELETE from department
306 where Dept_name = 'Packing';
307
308 • UPDATE department
309 SET Dept_name = 'Sports-material'
310 where Dept_ID = 'D1005';
311
312 • UPDATE department
313 SET Dept_head = 'E011'
314 where Dept_ID = 'D1004';
315
316
317 • DELETE from employee
318 where Emp_ID = 'E012';
```

The 'Output' pane shows the results of the executed queries:

#	Time	Action	Message	Duration / Fetch
1	06:43:48	DELETE from department where Dept_name = 'Packing'	1 row(s) affected	0.125 sec
2	06:43:48	UPDATE department SET Dept_name = 'Sports-material' where Dept_ID = 'D1005'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.047 sec
3	06:43:49	UPDATE department SET Dept_head = 'E011' where Dept_ID = 'D1004'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.032 sec

Employee Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains the following SQL statements:

```
316
317 • DELETE from employee
318 where Emp_ID = 'E012';
319
320 • UPDATE employee
321 SET Last_name = 'Dadigamma'
322 where Emp_ID = 'E003';
323
324 • UPDATE employee
325 SET Age = 44
326 where Emp_ID = 'E005';
327
328
329 • DELETE from worker
330 where Last_name = 'Siril';
331
332 • UPDATE worker
```

The 'Output' pane shows the results of the executed queries:

#	Time	Action	Message	Duration / Fetch
6	06:44:36	UPDATE employee SET Age = 44 where Emp_ID = E005	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.032 sec
7	06:45:01	SELECT * FROM factory.employee LIMIT 0, 1000	11 row(s) returned	0.000 sec / 0.000 sec

Worker Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains several SQL statements:

```
325 SET Age = 44
326 where Emp_ID = 'E005';
327
328
329 • DELETE from worker
330 where Last_name = 'Siril';
331
332 • UPDATE worker
333 SET First_name = 'Hasith'
334 where Worker_ID = 'W104';
335
336 • UPDATE worker
337 SET Age = 36
338 where Worker_ID = 'W103';
339
340
341
```

The 'Output' pane shows the results of the last two UPDATE statements:

#	Time	Action	Message	Duration / Fetch
2	06:46:25	UPDATE worker SET First_name = 'Hasith' where Worker_ID = 'W104'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.047 sec
3	06:46:25	UPDATE worker SET Age = 36 where Worker_ID = 'W103'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.031 sec

Product Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains several SQL statements:

```
338 where Worker_ID = 'W103';
339
340
341
342
343 • DELETE from product
344 where Stock_quantity = 175;
345
346 • UPDATE product
347 SET Prod_type = 'Kids'
348 where Prod_ID = 'P203';
349
350 • UPDATE product
351 SET Prod_name = 'T-shirt'
352 where Prod_ID = 'P205';
353
354
```

The 'Output' pane shows the results of the last two UPDATE statements:

#	Time	Action	Message	Duration / Fetch
21	06:57:36	UPDATE product SET Prod_type = 'Kids' where Prod_ID = 'P203'	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec
22	06:57:36	UPDATE product SET Prod_name = 'T-shirt' where Prod_ID = 'P205'	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec

Orders Table

The screenshot shows the MySQL Workbench interface with a script editor window titled "Factory_script". The script contains several SQL statements for the "orders" table:

```
367 • DELETE from orders  
368 where Price = 7500;  
369  
370 • UPDATE orders  
371 SET Quantity = 200  
372 where Order_ID = '0403' ;  
373  
374 • UPDATE orders  
375 SET Price = 200000  
376 where Order_ID = '0405';  
377  
378  
379 • DELETE from supplier  
380 where Raw_material = 'R304';  
381  
382 • UPDATE supplier
```

The "Output" pane shows the results of the executed queries:

#	Time	Action	Message	Duration / Fetch
4	07:00:36	UPDATE orders SET Quantity = 200 where Order_ID = '0403'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.031 sec
5	07:00:36	UPDATE orders SET Price = 200000 where Order_ID = '0405'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.031 sec

Raw Materials Table

The screenshot shows the MySQL Workbench interface with a script editor window titled "Factory_script". The script contains several SQL statements for the "raw_material" table:

```
352 where Prod_ID = 'P205';  
353  
354  
355 • DELETE from raw_material  
356 where Stock_quantity < 100;  
357  
358 • UPDATE raw_material  
359 SET Raw_name = 'Lables and Tags'  
360 where Raw_ID = 'R306';  
361  
362 • UPDATE raw_material  
363 SET Stock_quantity = 300  
364 where Raw_ID = 'R302';  
365  
366  
367 • DELETE from orders  
368 where Price = 7500;
```

The "Output" pane shows the results of the executed queries:

#	Time	Action	Message	Duration / Fetch
1	07:00:04	UPDATE raw_material SET Raw_name = 'Lables and Tags' where Raw_ID = 'R306'	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec
2	07:00:04	UPDATE raw_material SET Stock_quantity = 300 where Raw_ID = 'R302'	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec

Seller Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains several SQL statements:

```
387 SET Address_No = 50
388 where Raw_material = 'R305';
389
390
391 • DELETE from seller
392 where Order_ID = '0403';
393
394 • UPDATE seller
395 SET City = 'Ampara'
396 where Sell_name = 'Sahan';
397
398 • UPDATE seller
399 SET Street = 'Wijaya Mawatha'
400 where Order_ID = '0401';
401
402
403 • DELETE from machine
```

The 'Information' pane on the left shows the 'employee' table with the following columns:

Emp_ID	varchar(5) PK
First_name	varchar(50)
Last_name	varchar(50)
Age	int
Department	varchar(5)
Manager_ID	varchar(10)

The 'Output' pane at the bottom displays the results of the executed queries:

#	Time	Action	Message	Duration / Fetch
8	07:03:05	UPDATE seller SET City = 'Ampara' where Sell_name = 'Sahan'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.079 sec
9	07:03:05	UPDATE seller SET Street = 'Wijaya Mawatha' where Order_ID = '0401'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.062 sec

Supplier Table

The screenshot shows the MySQL Workbench interface with the 'Factory_script' tab selected. The code editor contains several SQL statements:

```
376 where Order_ID = '0405';
377
378
379 • DELETE from supplier
380 where Raw_material = 'R304';
381
382 • UPDATE supplier
383 SET Sup_name = 'Dadigamma'
384 where Sup_name = 'Sunil';
385
386 • UPDATE supplier
387 SET Address_No = 50
388 where Raw_material = 'R305';
389
390
391 • DELETE from seller
392 where Sell_name = 'Ishara';
393
```

The 'Information' pane on the left shows the 'employee' table with the following columns:

Emp_ID	varchar(5) PK
First_name	varchar(50)
Last_name	varchar(50)
Age	int
Department	varchar(5)
Manager_ID	varchar(10)

The 'Output' pane at the bottom displays the results of the executed queries:

#	Time	Action	Message	Duration / Fetch
16	06:53:55	UPDATE supplier SET Sup_name = 'Dadigamma' where Sup_name = 'Sunil'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.062 sec
17	06:55:08	UPDATE supplier SET Address_No = 50 where Raw_material = R305	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.047 sec

4. CHAPTER IV – TRANSACTIONS

4.1 SIMPLE QUERIES

1. Select Operation

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
3
4 -- Select Operation --
5 • SELECT * from employee
6 where Department = 'D1002';
7
8 • SELECT * from product
9 where Stock_quantity > 200;
10
```

The results grid shows data from the 'product' table:

Prod_ID	Prod_name	Prod_type	Department	Stock_quantity
P201	Blouse	Ladies	D1007	400
P203	Gearns	Ladies	D1006	275
P205	Shirt	Mens	D1005	325
P207	Blouse	Ladies	D1006	225
*	NULL	NULL	NULL	NULL

The output pane shows two actions:

#	Time	Action	Message	Duration / Fetch
144	07:06:10	SELECT * from employee where Department = 'D1002' LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
145	07:06:10	SELECT * from product where Stock_quantity > 200 LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec

2. Project Operation

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
11
12 -- Project Operation --
13 • SELECT First_name, Last_name from worker
14 where Department = 'D1002';
15
16
17 -- Cartesian Product Operation --
18 • SELECT * from worker
```

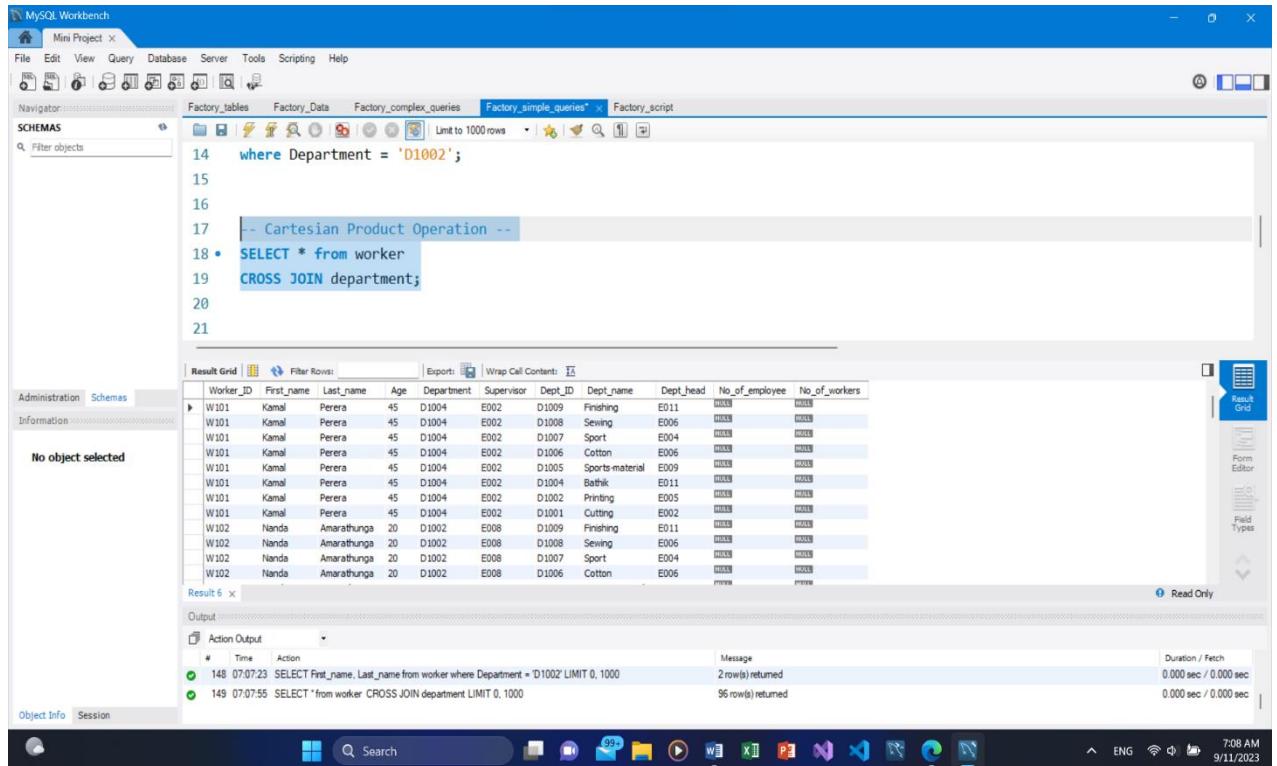
The results grid shows data from the 'worker' table:

First_name	Last_name
Nanda	Anorathunga
Sanderu	Silva

The output pane shows two actions:

#	Time	Action	Message	Duration / Fetch
147	07:07:18	SELECT Machine_name, Machine_function from machine where Department = 'D1003' LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
148	07:07:23	SELECT First_name, Last_name from worker where Department = 'D1002' LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec

3. Cartesian Product Operation



The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```

14     where Department = 'D1002';
15
16
17 -- Cartesian Product Operation --
18 •   SELECT * from worker
19     CROSS JOIN department;
20
21

```

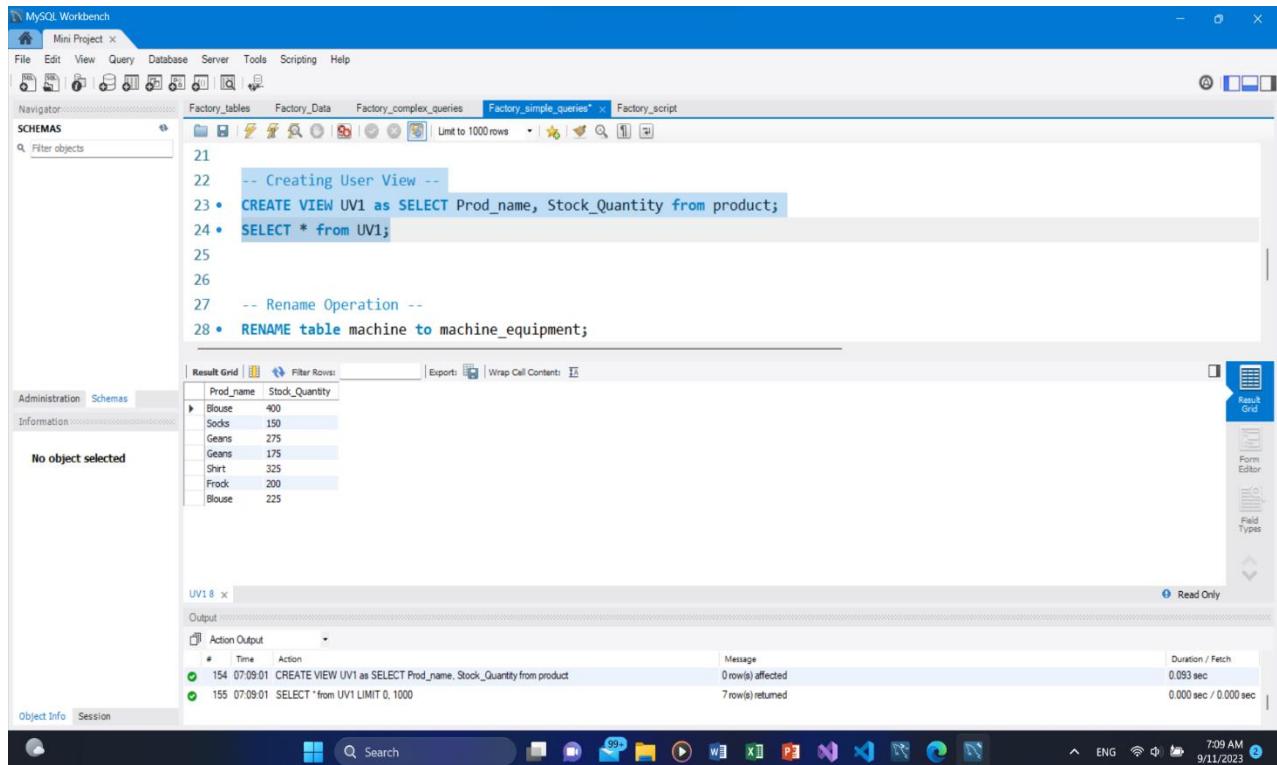
The result grid displays the Cartesian product of the worker and department tables. The columns are: Worker_ID, First_name, Last_name, Age, Department, Supervisor, Dept_ID, Dept_name, Dept_head, No_of_employee, and No_of_workers. The data shows every worker from the worker table paired with every department from the department table.

Worker_ID	First_name	Last_name	Age	Department	Supervisor	Dept_ID	Dept_name	Dept_head	No_of_employee	No_of_workers
W101	Kamal	Perera	45	D1004	E002	D1009	Finishing	E011	NULL	NULL
W101	Kamal	Perera	45	D1004	E002	D1008	Sewing	E006	NULL	NULL
W101	Kamal	Perera	45	D1004	E002	D1007	Sport	E004	NULL	NULL
W101	Kamal	Perera	45	D1004	E002	D1006	Cotton	E008	NULL	NULL
W101	Kamal	Perera	45	D1004	E002	D1005	Sports-material	E009	NULL	NULL
W101	Kamal	Perera	45	D1004	E002	D1004	Bathik	E011	NULL	NULL
W101	Kamal	Perera	45	D1004	E002	D1002	Printing	E005	NULL	NULL
W101	Kamal	Perera	45	D1004	E002	D1001	Cutting	E002	NULL	NULL
W102	Nanda	Amarathunga	20	D1002	E008	D1009	Finishing	E011	NULL	NULL
W102	Nanda	Amarathunga	20	D1002	E008	D1008	Sewing	E006	NULL	NULL
W102	Nanda	Amarathunga	20	D1002	E008	D1007	Sport	E004	NULL	NULL
W102	Nanda	Amarathunga	20	D1002	E008	D1006	Cotton	E006	NULL	NULL

The output pane shows two log entries:

- 148 07:07:23 SELECT First_name, Last_name from worker where Department = 'D1002' LIMIT 0, 1000 2 row(s) returned 0.000 sec / 0.000 sec
- 149 07:07:55 SELECT * from worker CROSS JOIN department LIMIT 0, 1000 96 row(s) returned 0.000 sec / 0.000 sec

4. Creating User View



The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```

21
22 -- Creating User View --
23 •   CREATE VIEW UV1 as SELECT Prod_name, Stock_Quantity from product;
24 •   SELECT * from UV1;
25
26
27 -- Rename Operation --
28 •   RENAME table machine to machine_equipment;

```

The result grid displays the data from the product table, which includes Prod_name and Stock_Quantity. The data is:

Prod_name	Stock_Quantity
Blouse	400
Socks	150
Geans	275
Geans	175
Shrt	325
Frock	200
Blouse	225

The output pane shows two log entries:

- 154 07:09:01 CREATE VIEW UV1 as SELECT Prod_name, Stock_Quantity from product 0 row(s) affected 0.093 sec
- 155 07:09:01 SELECT * from UV1 LIMIT 0, 1000 7 row(s) returned 0.000 sec / 0.000 sec

5. Rename Operation

The screenshot shows the MySQL Workbench interface. In the top-left pane, there's a 'Navigator' section with 'SCHEMAS' and a 'Filter objects' search bar. The main area contains a script editor with the following SQL code:

```
21
22 -- Creating User View --
23 • CREATE VIEW UV1 AS SELECT Prod_name, Stock_Quantity FROM product;
24 • SELECT * FROM UV1;
25
26
27 -- Rename Operation --
28 • RENAME TABLE machine TO machine_equipment;
29
30
31 -- Aggregation Function --
32 • SELECT Department, COUNT(Department) AS No_of_workers FROM worker GROUP BY Department;
33
34
35 -- Like Keyword --
36 • SELECT * FROM employee_contact WHERE Contact LIKE '070%';
37
```

Below the script editor is an 'Output' pane showing the execution results:

#	Time	Action	Message	Duration / Fetch
155	07:09:01	SELECT * FROM UV1 LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec
156	07:09:28	RENAME TABLE machine TO machine_equipment;	0 row(s) affected	0.047 sec

The bottom status bar shows the date and time: 9/11/2023 7:09 AM.

6. Aggregation Function

The screenshot shows the MySQL Workbench interface. In the top-left pane, there's a 'Navigator' section with 'SCHEMAS' and a 'Filter objects' search bar. The main area contains a script editor with the following SQL code:

```
28 • RENAME TABLE machine TO machine_equipment;
29
30
31 -- Aggregation Function --
32 • SELECT Department, COUNT(Department) AS No_of_workers FROM worker GROUP BY Department;
33
34
35 -- Like Keyword --
```

Below the script editor is a 'Result Grid' pane showing the output of the aggregation query:

Department	No_of_workers
HQ	0
D1001	1
D1002	2
D1004	1
D1005	1
D1006	3
D1007	2
D1008	1

Below the result grid is an 'Output' pane showing the execution results:

#	Time	Action	Message	Duration / Fetch
156	07:09:28	RENAME TABLE machine TO machine_equipment;	0 row(s) affected	0.047 sec
157	07:09:53	SELECT Department, COUNT(Department) AS No_of_workers FROM worker GROUP BY Department LIMIT 0, 1000	8 row(s) returned	0.016 sec / 0.000 sec

The bottom status bar shows the date and time: 9/11/2023 7:10 AM.

7. Function using LIKE keyword

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The main window has tabs for Navigator, Schemas, and Result Grid. The Result Grid tab is active, displaying a table with columns Emp_ID and Contact. The table contains three rows: E001 (0700756432), E003 (0700771339), and a fourth row with a redacted contact number. Below the table is an Output panel titled 'employee_contact 10 x' which shows two log entries. The first entry is a SELECT statement for department counts, and the second is a SELECT statement using the LIKE keyword to find contacts starting with '070'. The bottom status bar shows the date and time as 9/11/2023 7:10 AM.

```
31 -- Aggregation Function --
32 • SELECT Department , count(Department) as No_of_workers from worker group by Department;
33
34
35 -- Like Keyword --
36 • SELECT * from employee_contact where Contact like '070%';
37

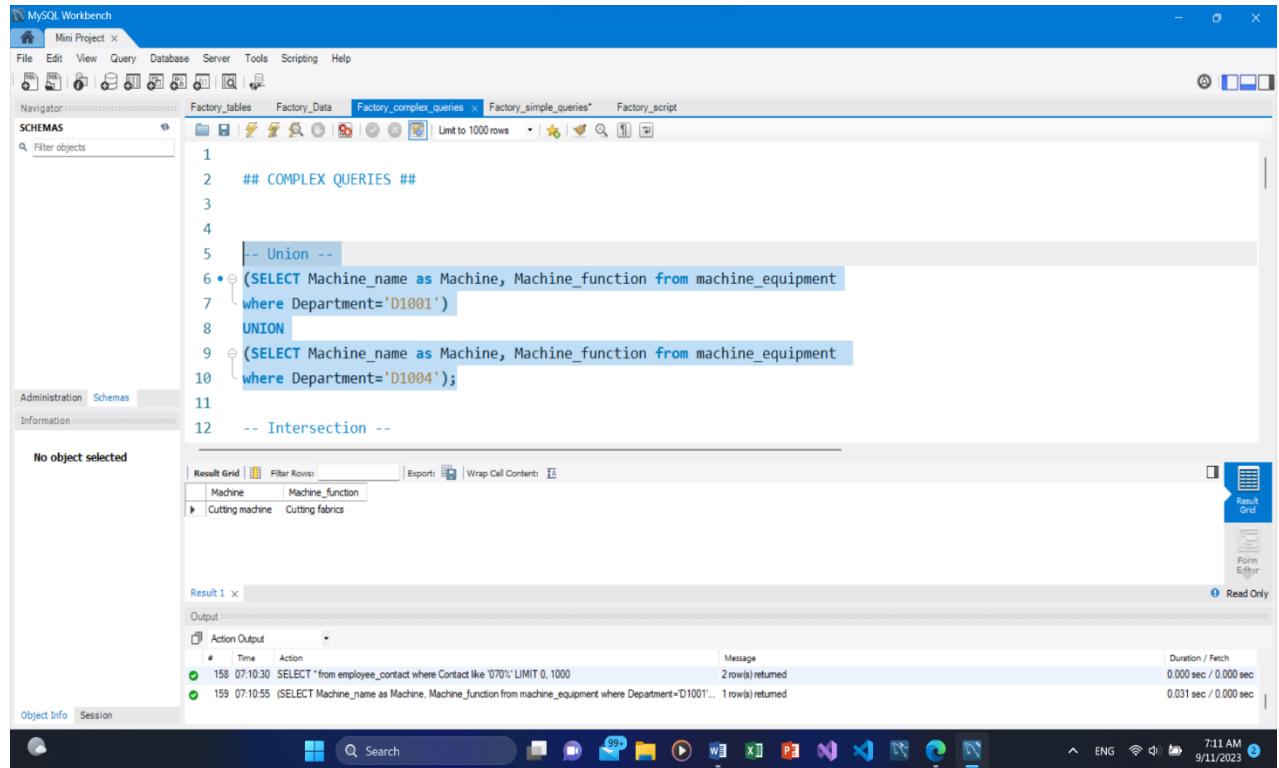
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |
Administration Schemas Information No object selected

employee_contact 10 x
Output
Action Output
# Time Action Message Duration / Fetch
157 07:09:53 SELECT Department , count(Department) as No_of_workers from worker group by Department LIMIT 0, 1000 8 row(s) returned 0.016 sec / 0.000 sec
158 07:10:30 SELECT * from employee_contact where Contact like '070%' LIMIT 0, 1000 2 row(s) returned 0.000 sec / 0.000 sec

Object Info Session
```

4.2 COMPLEX QUERIES

1. Union Operation



The screenshot shows the MySQL Workbench interface with a query editor window titled "Factory_complex_queries". The code entered is:

```
1
2  ## COMPLEX QUERIES ##
3
4
5  -- Union --
6 • (SELECT Machine_name as Machine, Machine_function from machine_equipment
7  where Department='D1001')
8 UNION
9 • (SELECT Machine_name as Machine, Machine_function from machine_equipment
10 where Department='D1004');
11
12 -- Intersection --
```

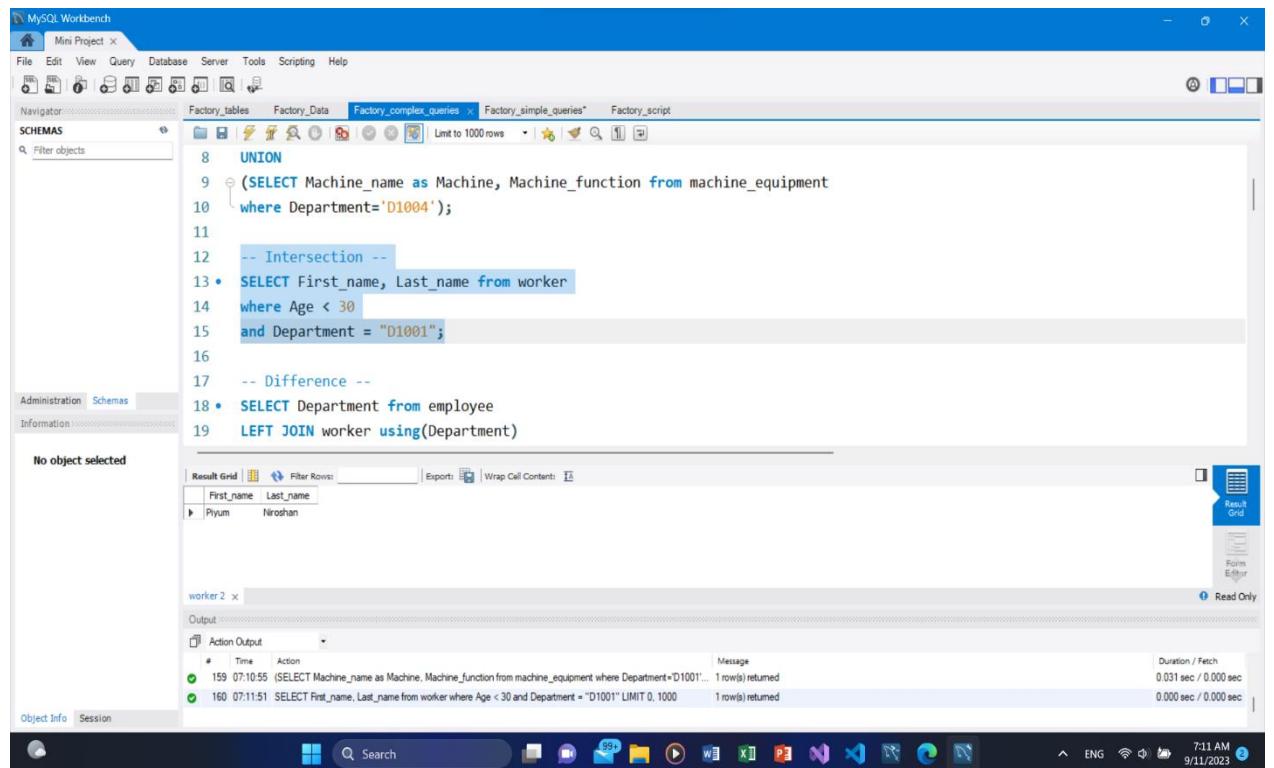
The result grid shows the output of the UNION query:

Machine	Machine_function
Cutting machine	Cutting fabrics

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
158	07:10:30	SELECT * from employee_contact where Contact like '070%' LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
159	07:10:55	(SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1001'... 1 row(s) returned		0.031 sec / 0.000 sec

2. Intersection Operation



The screenshot shows the MySQL Workbench interface with a query editor window titled "Factory_complex_queries". The code entered is:

```
8 UNION
9 • (SELECT Machine_name as Machine, Machine_function from machine_equipment
10 where Department='D1004');
11
12 -- Intersection --
13 • SELECT First_name, Last_name from worker
14 where Age < 30
15 and Department = "D1001";
16
17 -- Difference --
18 • SELECT Department from employee
19 LEFT JOIN worker using(Department)
```

The result grid shows the output of the intersection query:

First_name	Last_name
Pyum	Niroshan

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
159	07:10:55	(SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1001'... 1 row(s) returned		0.031 sec / 0.000 sec
160	07:11:51	SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001" LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

3. Set Difference Operation

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
15    and Department = "D1001";
16
17    -- Difference --
18 •   SELECT Department from employee
19     LEFT JOIN worker using(Department)
20   where Department = "D1001";
21
22    -- Division --
23 •   SELECT distinct P.Prod_ID from product_order as P
24   where not exists (SELECT Order_ID from product_order as O);
25
26    -- Inner Join --
```

The results pane shows a table with one row:

Department
D1001

The log pane shows two actions:

#	Time	Action	Message	Duration / Fetch
160	07:11:51	SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001" LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
161	07:12:14	SELECT Department from employee LEFT JOIN worker using(Department) where Department = "D1001" LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec

4. Division Operation

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:

```
17    -- Difference --
18 •   SELECT Department from employee
19     LEFT JOIN worker using(Department)
20   where Department = "D1001";
21
22    -- Division --
23 •   SELECT distinct P.Prod_ID from product_order as P
24   where not exists (SELECT Order_ID from product_order as O);
25
26    -- Inner Join --
27 •   CREATE VIEW UV2 as (SELECT P.Prod_name, P.Department from product as P);
28 •   CREATE VIEW UV3 as (SELECT D.Dept_name, D.Dept_ID from department as D);
```

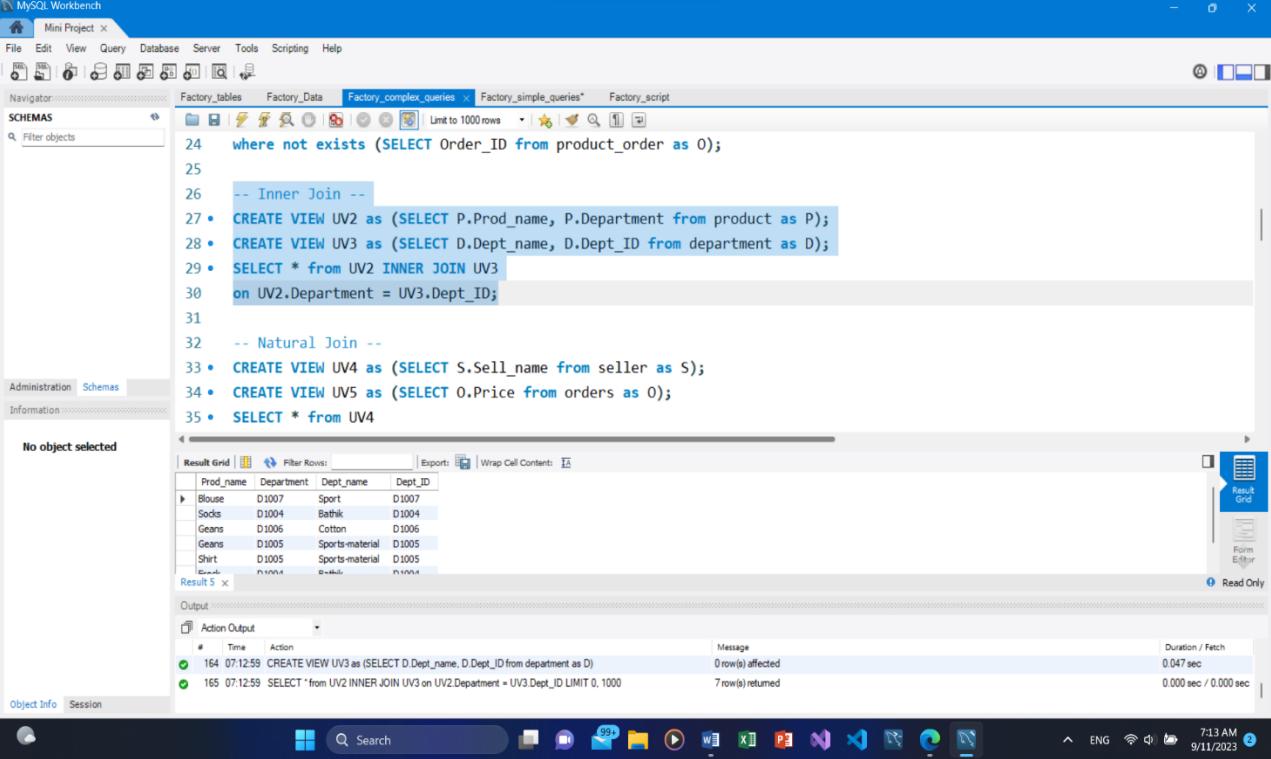
The results pane shows a table with one row:

Prod_ID

The log pane shows two actions:

#	Time	Action	Message	Duration / Fetch
161	07:12:14	SELECT Department from employee LEFT JOIN worker using(Department) where Department = "D1001" LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
162	07:12:39	SELECT distinct P.Prod_ID from product_order as P where not exists (SELECT Order_ID from product_order as O)	0 row(s) returned	0.000 sec / 0.000 sec

5. Inner Join Operation



The screenshot shows the MySQL Workbench interface with a query editor window. The query being run is:

```
24 where not exists (SELECT Order_ID from product_order as O);
25
26 -- Inner Join --
27 • CREATE VIEW UV2 as (SELECT P.Prod_name, P.Department from product as P);
28 • CREATE VIEW UV3 as (SELECT D.Dept_name, D.Dept_ID from department as D);
29 • SELECT * from UV2 INNER JOIN UV3
30 on UV2.Department = UV3.Dept_ID;
31
32 -- Natural Join --
33 • CREATE VIEW UV4 as (SELECT S.Sell_name from seller as S);
34 • CREATE VIEW UV5 as (SELECT O.Price from orders as O);
35 • SELECT * from UV4
```

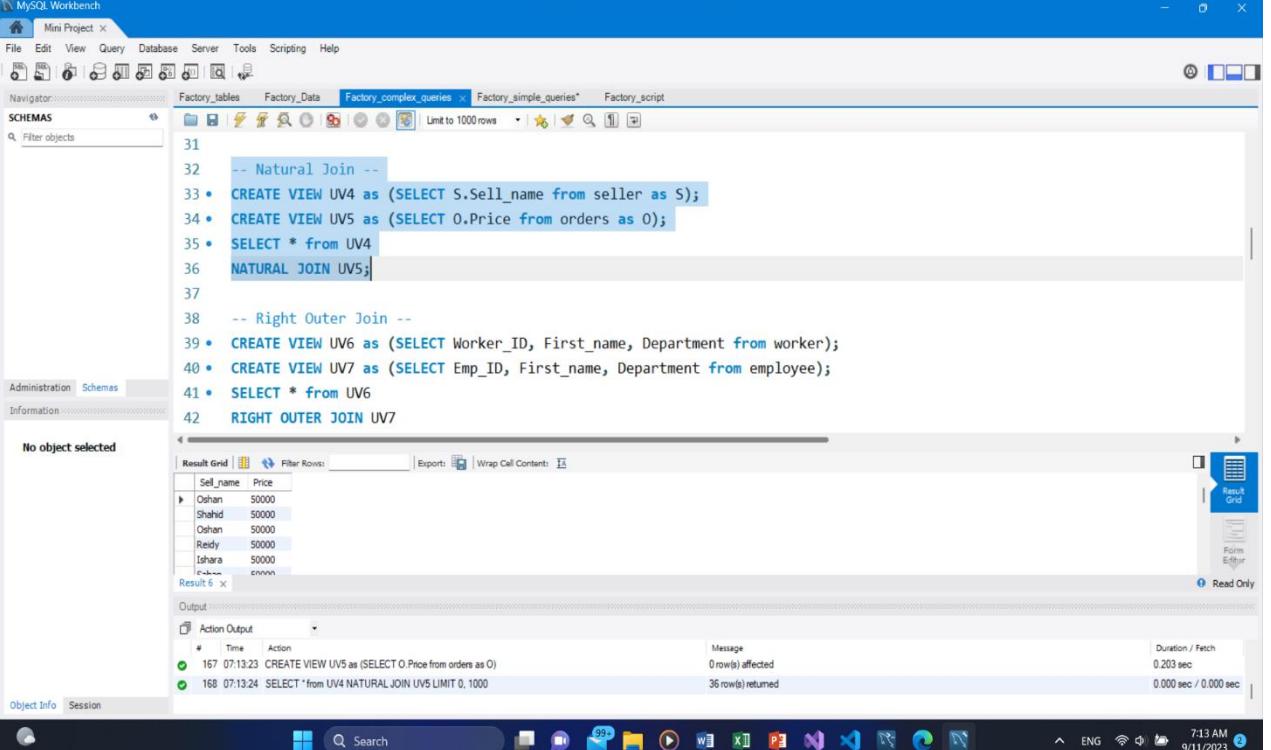
The results grid shows the following data:

Prod_name	Department	Dept_name	Dept_ID
Blouse	D1007	Sport	D1007
Socks	D1004	Bathik	D1004
Gears	D1006	Cotton	D1006
Gears	D1005	Sports-material	D1005
Shirt	D1005	Sports-material	D1005
Blouse	D1004	Bathik	D1004

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
164	07:12:59	CREATE VIEW UV3 as (SELECT D.Dept_name, D.Dept_ID from department as D)	0 row(s) affected	0.047 sec
165	07:12:59	SELECT * from UV2 INNER JOIN UV3 on UV2.Department = UV3.Dept_ID LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec

6. Natural Join Operation



The screenshot shows the MySQL Workbench interface with a query editor window. The query being run is:

```
31
32 -- Natural Join --
33 • CREATE VIEW UV4 as (SELECT S.Sell_name from seller as S);
34 • CREATE VIEW UV5 as (SELECT O.Price from orders as O);
35 • SELECT * from UV4
36 NATURAL JOIN UV5;
```

The results grid shows the following data:

Sel_name	Price
Oshan	50000
Shahid	50000
Oshan	50000
Redy	50000
Ishara	50000
Shahid	50000

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
167	07:13:23	CREATE VIEW UV5 as (SELECT O.Price from orders as O)	0 row(s) affected	0.203 sec
168	07:13:24	SELECT * from UV4 NATURAL JOIN UV5 LIMIT 0, 1000	36 row(s) returned	0.000 sec / 0.000 sec

7. Right Outer Join Operation

The screenshot shows the MySQL Workbench interface with the 'Factory_complex_queries' tab selected. The code editor contains the following SQL statements:

```
34 • CREATE VIEW UV5 AS (SELECT O.Price FROM orders AS O);
35 • SELECT * FROM UV4
36 NATURAL JOIN UV5;
37
38 -- Right Outer Join --
39 • CREATE VIEW UV6 AS (SELECT Worker_ID, First_name, Department FROM worker);
40 • CREATE VIEW UV7 AS (SELECT Emp_ID, First_name, Department FROM employee);
41 • SELECT * FROM UV6
42 RIGHT OUTER JOIN UV7
43 ON UV6.Department = UV7.Department;
44
45 -- Left Outer Join --
```

The 'Result Grid' pane displays the results of the query in step 42, which is a right outer join between UV6 and UV7. The data is as follows:

Worker_ID	First_name	Department	Emp_ID	First_name	Department
W105			E001	Menaka	
W106	Udara	D1006	E002	Rasika	D1006
W107	Umatha	D1006	E002	Rasika	D1006
W110	Udara	D1006	E002	Rasika	D1006
W104	Hasith	D1008	E003	Nayana	D1008

The 'Output' pane shows the execution log:

- 170 07:13:56 CREATE VIEW UV7 AS (SELECT Emp_ID, First_name, Department FROM employee)
- 171 07:13:56 SELECT * FROM UV6 RIGHT OUTER JOIN UV7 ON UV6.Department = UV7.Department LIMIT 0, 1000

8. Left Outer Join Operation

The screenshot shows the MySQL Workbench interface with the 'Factory_complex_queries' tab selected. The code editor contains the following SQL statements:

```
41 • SELECT * FROM UV6
42 RIGHT OUTER JOIN UV7
43 ON UV6.Department = UV7.Department;
44
45 -- Left Outer Join --
46 • CREATE VIEW UV8 AS (SELECT Worker_ID, First_name, Department FROM worker);
47 • CREATE VIEW UV9 AS (SELECT Emp_ID, First_name, Department FROM employee);
48 • SELECT * FROM UV8
49 LEFT OUTER JOIN UV9
50 ON UV8.Department = UV9.Department;
51
52 -- Full Outer Join --
```

The 'Result Grid' pane displays the results of the query in step 49, which is a left outer join between UV8 and UV9. The data is as follows:

Worker_ID	First_name	Department	Emp_ID	First_name	Department
W101	Kanal	D1004			
W102	Nanda	D1002	E008	Imesh	D1002
W103	Sri	D1005	E010	Udara	D1005
W104	Hasith	D1008	E003	Nayana	D1008
W104	Hasith	D1008	E006	Yohan	D1008

The 'Output' pane shows the execution log:

- 173 07:14:24 CREATE VIEW UV9 AS (SELECT Emp_ID, First_name, Department FROM employee)
- 174 07:14:24 SELECT * FROM UV8 LEFT OUTER JOIN UV9 ON UV8.Department = UV9.Department LIMIT 0, 1000

9. Full Outer Join Operation

The screenshot shows the MySQL Workbench interface with a query editor window. The code is as follows:

```
51
52  -- Full Outer Join --
53 • CREATE VIEW UV10 as (SELECT Worker_ID, First_name, Department from worker);
54 • CREATE VIEW UV11 as (SELECT Emp_ID, First_name, Department from employee);
55 • (SELECT * from UV10
56   RIGHT OUTER JOIN UV11
57   on UV10.Department = UV11.Department)
58 UNION
59 • (SELECT * from UV10
60   LEFT OUTER JOIN UV11
61   on UV10.Department = UV11.Department);
62
```

The Result Grid shows the following data:

Worker_ID	First_name	Department	Emp_ID	First_name	Department
W106	Udari	D1006	E001	Menuka	
W107	Umatha	D1006	E002	Rasika	D1006
W110	Udari	D1006	E002	Rasika	D1006
W104	Hasthi	D1008	E003	Nayana	D1008

The Output pane shows the following log entries:

#	Time	Action	Message	Duration / Fetch
176	07:15:00	CREATE VIEW UV11 as (SELECT Emp_ID, First_name, Department from employee)	0 row(s) affected	0.031 sec
177	07:15:00	(SELECT * from UV10 RIGHT OUTER JOIN UV11 on UV10.Department = UV11.Department) UNION (SELECT * from UV10 LEFT OUTER JOIN UV11 on UV10.Department = UV11.Department);	17 row(s) returned	0.000 sec / 0.000 sec

10. Outer Union Operation

The screenshot shows the MySQL Workbench interface with a query editor window. The code is as follows:

```
62
63  -- Outer Union Join --
64 • CREATE VIEW UV12 as (SELECT Worker_ID, First_name, Department from worker);
65 • CREATE VIEW UV13 as (SELECT Emp_ID, First_name, Department from employee);
66 • (SELECT * from UV12
67   RIGHT OUTER JOIN UV13
68   on UV12.Department = UV13.Department)
69 UNION ALL
70 • (SELECT * from UV12
71   LEFT OUTER JOIN UV13
72   on UV12.Department = UV13.Department);
73
```

The Result Grid shows the following data:

Worker_ID	First_name	Department	Emp_ID	First_name	Department
W106	Udari	D1006	E001	Menuka	
W107	Umatha	D1006	E002	Rasika	D1006
W110	Udari	D1006	E002	Rasika	D1006
W104	Hasthi	D1008	E003	Nayana	D1008

The Output pane shows the following log entries:

#	Time	Action	Message	Duration / Fetch
179	07:15:53	CREATE VIEW UV13 as (SELECT Emp_ID, First_name, Department from employee)	0 row(s) affected	0.031 sec
180	07:15:53	(SELECT * from UV12 RIGHT OUTER JOIN UV13 on UV12.Department = UV13.Department) UNION ALL (SELECT * from UV12 LEFT OUTER JOIN UV13 on UV12.Department = UV13.Department);	29 row(s) returned	0.000 sec / 0.000 sec

11. Nested Query 01

The screenshot shows the MySQL Workbench interface with a script editor containing a nested query. The query selects employees from the 'employee' table where their department is 'D1001' and their manager's ID is in a subquery that selects manager IDs for employees whose last name is 'Perera'. The results are displayed in a grid:

First_name	Last_name
T	Thenachayn
Omkha	de Silva

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
583	09:23:11	INSERT into supplier_contact(Sup_name, Raw_material, Contact) values("Anil","R303","0700756746")	1 row(s) affected	0.031 sec
584	09:23:15	SELECT First_name, Last_name from employee where Department = 'D1001' AND Manager_ID IN (SELECT Emp_ID from employee WHERE Last_name = 'Perera')	2 row(s) returned	0.000 sec / 0.000 sec

12. Nested Query 02

The screenshot shows the MySQL Workbench interface with a script editor containing a more complex nested query. It selects suppliers from the 'supplier' table where their raw material ID is in a subquery that selects raw material IDs where stock quantity is greater than 150. The results are displayed in a grid:

Sup_name
Anith
Lalith
Anil
Sunil
Kamal

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
181	07:16:20	SELECT First_name, Last_name from employee where Department = 'D1001' AND Manager_ID IN (SELECT Emp_ID from employee WHERE Last_name = 'Perera')	2 row(s) returned	0.000 sec / 0.016 sec
182	07:16:45	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw_material where Stock_quantity > 150)	5 row(s) returned	0.000 sec / 0.000 sec

13. Nested Query 03

The screenshot shows the MySQL Workbench interface with a query editor window. The query being run is:

```
84 | SELECT Raw_ID from raw_material  
85 | where Stock_quantity > 150 ;  
86 |  
87 | -- Nested Query 3 --  
88 • SELECT First_name, Last_name from worker  
89 • where Department IN (  
90 |   SELECT Dept_ID from department  
91 |   where Dept_head IN (  
92 |     SELECT Emp_ID from employee  
93 |     where Age < 40));  
94 |  
95 |
```

The result grid shows the following data:

First_name	Last_name
Kamal	Perera
Siri	Peris
Udari	Silva
Umatha	Yehan
Udari	Madumanthika

The output pane shows the following log entries:

#	Time	Action	Message	Duration / Fetch
182	07:16:45	SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw_material where Stock_Q...)	5 row(s) returned	0.000 sec / 0.000 sec
183	07:17:04	SELECT First_name, Last_name from worker where Department IN (SELECT Dept_ID from department where ...)	8 row(s) returned	0.000 sec / 0.000 sec

5. CHAPTER V – TUNING

Union

Without Indexing

```
MySQL 8.0 Command Line Client
Enter password: *****
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 132
Server version: 8.0.33 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use factory
Database changed
mysql> explain ((SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1001')UNION(SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1004'));
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | PRIMARY | machine_equipment | NULL | ref | FK_DeptID | FK_DeptID | 43 | const |
| 2 | UNION | machine_equipment | NULL | ref | FK_DeptID | FK_DeptID | 43 | const |
| 3 | UNION RESULT | <union1,2> | NULL | ALL | NULL | NULL | NULL | NULL |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.06 sec)

mysql>
```

With Indexing

```
MySQL 8.0 Command Line Client
Enter password: *****
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 132
Server version: 8.0.33 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use factory
Database changed
mysql> explain ((SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1001')UNION(SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1004'));
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | PRIMARY | machine_equipment | NULL | ref | FK_DeptID | FK_DeptID | 43 | const |
| 2 | UNION | machine_equipment | NULL | ref | FK_DeptID | FK_DeptID | 43 | const |
| 3 | UNION RESULT | <union1,2> | NULL | ALL | NULL | NULL | NULL | NULL |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.06 sec)

mysql> create index department_index using BTREE on machine_equipment(department);
Query OK, 0 rows affected (0.23 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> explain ((SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1001')UNION(SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1004'));
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | PRIMARY | machine_equipment | NULL | ref | department_index | department_index | 43 | const |
| 2 | UNION | machine_equipment | NULL | ref | department_index | department_index | 43 | const |
| 3 | UNION RESULT | <union1,2> | NULL | ALL | NULL | NULL | NULL | NULL |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.00 sec)

mysql>
```

Difference

Without Indexing

```
MySQL 8.0 Command Line Client
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | PRIMARY | machine_equipment | NULL | ref | department_index | department_index | 43 | const | 1 | 100.00 | NULL |
| 2 | UNION | machine_equipment | NULL | ref | department_index | department_index | 43 | const | 1 | 100.00 | NULL |
| 3 | UNION RESULT | union1,2> | NULL | ALL | NULL | NULL | NULL | NULL | NULL | NULL | Using temporary |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.00 sec)

mysql> explain First_name, last_name from worker where Age < 30 and Department = "D1001";
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'Department = "D1001"' at line 1
mysql> explain First_name, last_name from worker where Age < 30 and Department = "D1001";
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'Department = "D1001"' at line 1
mysql> explain (SELECT First_name, last_name from worker where Age < 30 and Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | worker | NULL | ref | FK_Worker_DeptID | FK_Worker_DeptID | 23 | const | 1 | 33.33 | Using where |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql> create index index_department_age using BTREE on Worker(department,age);
Query OK, 0 rows affected (0.59 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | worker | NULL | range | index_department_age | index_department_age | 28 | NULL | 1 | 100.00 | Using index condition |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql> explain (SELECT Department from employee LEFT JOIN worker using(Department) where Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | employee | NULL | ref | FK_Employee_DeptID | FK_Employee_DeptID | 23 | const | 2 | 100.00 | Using index |
| 1 | SIMPLE | worker | NULL | ref | index_department_age | index_department_age | 23 | const | 1 | 100.00 | Using index |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

With Indexing

```
MySQL 8.0 Command Line Client
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'Department = "D1001"' at line 1
mysql> explain (SELECT First_name, last_name from worker where Age < 30 and Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | worker | NULL | ref | FK_Worker_DeptID | FK_Worker_DeptID | 23 | const | 1 | 33.33 | Using where |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql> create index index_department_age using BTREE on Worker(department,age);
Query OK, 0 rows affected (0.59 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | worker | NULL | range | index_department_age | index_department_age | 28 | NULL | 1 | 100.00 | Using index condition |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql> explain (SELECT Department from employee LEFT JOIN worker using(Department) where Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | employee | NULL | ref | FK_Employee_DeptID | FK_Employee_DeptID | 23 | const | 2 | 100.00 | Using index |
| 1 | SIMPLE | worker | NULL | ref | index_department_age | index_department_age | 23 | const | 1 | 100.00 | Using index |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql> create index index_department using BTREE on employee(department);
Query OK, 0 rows affected (0.19 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> explain (SELECT Department from employee LEFT JOIN worker using(Department) where Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | employee | NULL | ref | index_department | index_department | 23 | const | 2 | 100.00 | Using index |
| 1 | SIMPLE | worker | NULL | ref | index_department_age | index_department_age | 23 | const | 1 | 100.00 | Using index |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

Intersection

Without Indexing

```
MySQL 8.0 Command Line Client
Database changed
mysql> explain ((SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1001')UNION(SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1004'));
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | PRIMARY | machine_equipment | NULL | ref | FK_DeptID | FK_DeptID | 43 | const | 1 | 100.00 | NULL |
| 2 | UNION | machine_equipment | NULL | ref | FK_DeptID | FK_DeptID | 43 | const | 1 | 100.00 | NULL |
| 3 | UNION RESULT | union1,> | NULL | ALL | NULL | NULL | NULL | NULL | NULL | NULL | Using temporary |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.06 sec)

mysql> create index department_index using BTREE on machine_equipment(department);
Query OK, 0 rows affected (0.23 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> explain ((SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1001')UNION(SELECT Machine_name as Machine, Machine_function from machine_equipment where Department='D1004'));
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | PRIMARY | machine_equipment | NULL | ref | department_index | department_index | 43 | const | 1 | 100.00 | NULL |
| 2 | UNION | machine_equipment | NULL | ref | department_index | department_index | 43 | const | 1 | 100.00 | NULL |
| 3 | UNION RESULT | <union1,> | NULL | ALL | NULL | NULL | NULL | NULL | NULL | Using temporary |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.00 sec)

mysql> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'Department = "D1001"' at line 1
mysql> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'Department = "D1001"' at line 1
mysql> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | worker | NULL | ref | FK_Worker_DeptID | FK_Worker_DeptID | 23 | const | 1 | 33.33 | Using where |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql>
```

With Indexing

```
MySQL 8.0 Command Line Client
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | PRIMARY | machine_equipment | NULL | ref | department_index | department_index | 43 | const | 1 | 100.00 | NULL |
| 2 | UNION | machine_equipment | NULL | ref | department_index | department_index | 43 | const | 1 | 100.00 | NULL |
| 3 | UNION RESULT | <union1,> | NULL | ALL | NULL | NULL | NULL | NULL | NULL | Using temporary |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.00 sec)

mysql> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'Department = "D1001"' at line 1
mysql> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'Department = "D1001"' at line 1
mysql> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | worker | NULL | ref | FK_Worker_DeptID | FK_Worker_DeptID | 23 | const | 1 | 33.33 | Using where |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql> create index index_department_age using BTREE on Worker(department,age);
Query OK, 0 rows affected (0.59 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | worker | NULL | range | index_department_age | index_department_age | 28 | NULL | 1 | 100.00 | Using index condition |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

mysql>
```

Inner Join

Without Indexing

```
MySQL> create index index_department_age using BTREE on Worker(department,age);
Query OK, 0 rows affected (0.59 sec)
Records: 0 Duplicates: 0 Warnings: 0

MySQL> explain (SELECT First_name, Last_name from worker where Age < 30 and Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | worker | NULL | range | index_department_age | index_department_age | 28 | NULL |
| 1 | SIMPLE | worker | NULL | range | index_department_age | index_department_age | 28 | NULL |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set, 1 warning (0.00 sec)

MySQL> explain (SELECT Department from employee LEFT JOIN worker using(Department) where Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | employee | NULL | ref | FK_Employee_DeptID | FK_Employee_DeptID | 23 | const |
| 1 | SIMPLE | worker | NULL | ref | index_department_age | index_department_age | 23 | const |
+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

MySQL> create index index_department using BTREE on employee(department);
Query OK, 0 rows affected (0.19 sec)
Records: 0 Duplicates: 0 Warnings: 0

MySQL> explain (SELECT Department from employee LEFT JOIN worker using(Department) where Department = "D1001");
+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | employee | NULL | ref | index_department | index_department | 23 | const |
| 1 | SIMPLE | worker | NULL | ref | index_department_age | index_department_age | 23 | const |
+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

MySQL> explain(SELECT * from UV2 INNER JOIN UV3 on UV2.Department = UV3.Dept_ID);
+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | p | NULL | ALL | FK_Product_DeptID | NULL | NULL | NULL |
| 1 | SIMPLE | d | NULL | eq_ref | PRIMARY | PRIMARY | 42 | factory.p.Department |
+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

->
```

With Indexing

```
MySQL> create index index_department using BTREE on product(department);
Query OK, 0 rows affected (0.36 sec)
Records: 0 Duplicates: 0 Warnings: 0

MySQL> create index index_dept_id using BTREE on department(dept_id);
Query OK, 0 rows affected (0.17 sec)
Records: 0 Duplicates: 0 Warnings: 0

MySQL> explain(SELECT * from UV2 INNER JOIN UV3 on UV2.Department = UV3.Dept_ID);
+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | p | NULL | ALL | index_department | NULL | NULL | NULL |
| 1 | SIMPLE | d | NULL | eq_ref | PRIMARY,index_dept_id | PRIMARY | 42 | factory.p.Department |
+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

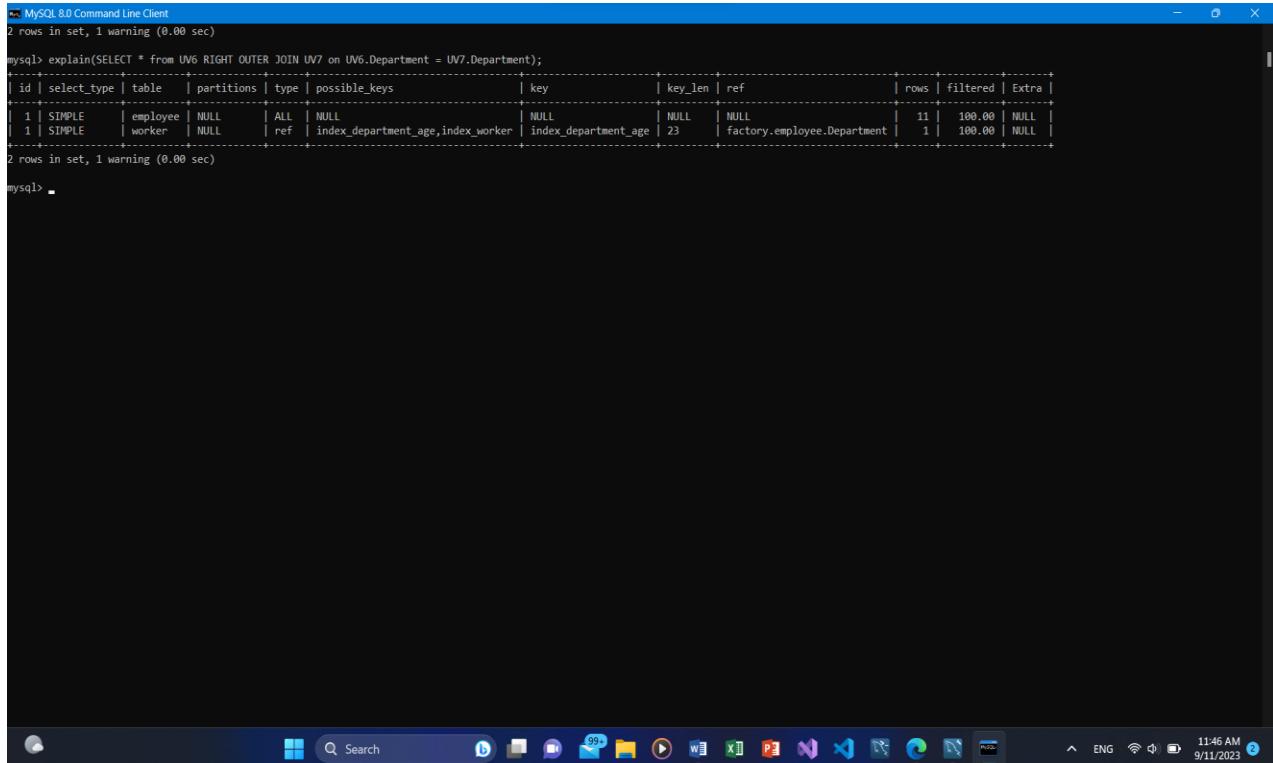
MySQL> -
```

Right Outer Join

Without Indexing

```
MySQL> explain(SELECT * from UV6 RIGHT OUTER JOIN UV7 on UV6.Department = UV7.Department);
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | employee | NULL | ALL | NULL | NULL | NULL | NULL |
| 1 | SIMPLE | worker | NULL | ref | index_department_age,index_worker | index_department_age | 23 | factory.employee.Department |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql>
```



With Indexing

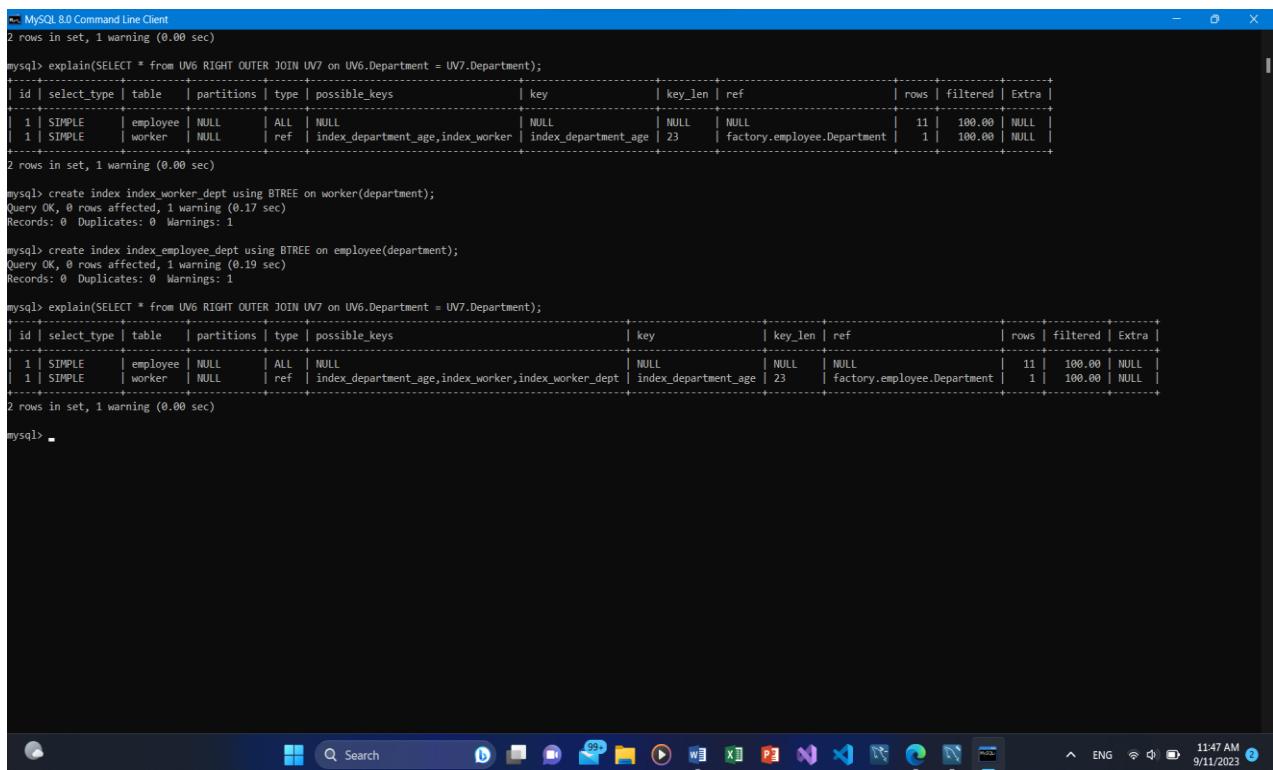
```
MySQL> explain(SELECT * from UV6 RIGHT OUTER JOIN UV7 on UV6.Department = UV7.Department);
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | employee | NULL | ALL | NULL | NULL | NULL | NULL |
| 1 | SIMPLE | worker | NULL | ref | index_department_age,index_worker | index_department_age | 23 | factory.employee.Department |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql> create index index_worker_dept using BTREE on worker(department);
Query OK, 0 rows affected, 1 warning (0.17 sec)
Records: 0 Duplicates: 0 Warnings: 1

mysql> create index index_employee_dept using BTREE on employee(department);
Query OK, 0 rows affected, 1 warning (0.19 sec)
Records: 0 Duplicates: 0 Warnings: 1

mysql> explain(SELECT * from UV6 RIGHT OUTER JOIN UV7 on UV6.Department = UV7.Department);
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE | employee | NULL | ALL | NULL | NULL | NULL | NULL |
| 1 | SIMPLE | worker | NULL | ref | index_department_age,index_worker,index_worker_dept | index_department_age | 23 | factory.employee.Department |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql>
```



Left Outer Join

Without Indexing

```
MySQL 8.0 Command Line Client
+-----+
2 rows in set, 1 warning (0.00 sec)

mysql> explain (SELECT * from UV8 LEFT OUTER JOIN UV9 on UV8.Department = UV9.Department);
+-----+
| id | select_type | table   | partitions | type | possible_keys | key      | key_len | ref           | rows | filtered | Extra |
+-----+
| 1 | SIMPLE      | worker  | NULL      | ALL  | NULL          | NULL     | NULL    | NULL          | 12  | 100.00   | NULL |
| 1 | SIMPLE      | employee | NULL      | ref   | index_department | index_department | 23     | factory.worker.Department | 1   | 100.00   | NULL |
+-----+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

With Indexing

```
MySQL 8.0 Command Line Client
+-----+
2 rows in set, 1 warning (0.00 sec)

mysql> explain (SELECT * from UV8 LEFT OUTER JOIN UV9 on UV8.Department = UV9.Department);
+-----+
| id | select_type | table   | partitions | type | possible_keys | key      | key_len | ref           | rows | filtered | Extra |
+-----+
| 1 | SIMPLE      | worker  | NULL      | ALL  | NULL          | NULL     | NULL    | NULL          | 12  | 100.00   | NULL |
| 1 | SIMPLE      | employee | NULL      | ref   | index_department | index_department | 23     | factory.worker.Department | 1   | 100.00   | NULL |
+-----+
2 rows in set, 1 warning (0.00 sec)

mysql> create index index_worker using BTREE on worker(department);
Query OK, 0 rows affected (0.14 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> create index index_employee using BTREE on employee(department);
Query OK, 0 rows affected, 1 warning (0.15 sec)
Records: 0  Duplicates: 0  Warnings: 1

mysql> explain (SELECT * from UV8 LEFT OUTER JOIN UV9 on UV8.Department = UV9.Department);
+-----+
| id | select_type | table   | partitions | type | possible_keys | key      | key_len | ref           | rows | filtered | Extra |
+-----+
| 1 | SIMPLE      | worker  | NULL      | ALL  | NULL          | NULL     | NULL    | NULL          | 12  | 100.00   | NULL |
| 1 | SIMPLE      | employee | NULL      | ref   | index_department,index_employee | index_department | 23     | factory.worker.Department | 1   | 100.00   | NULL |
+-----+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

Nested Query 01

Without Indexing

```
MySQL 8.0 Command Line Client

mysql> explain(SELECT First_name, Last_name from employee where Department = 'D1001' AND Manager_ID IN (SELECT Emp_ID from employee WHERE Last_name = 'Perera'));
+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys
+-----+-----+-----+-----+-----+
| 1 | SIMPLE     | employee | NULL      | ref  | FK_Manager,index_department,index_employee,index_employee_dept
| 1 | SIMPLE     | employee | NULL      | eq_ref| PRIMARY
+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

With Indexing

```
MySQL 8.0 Command Line Client

mysql> explain(SELECT First_name, Last_name from employee where Department = 'D1001' AND Manager_ID IN (SELECT Emp_ID from employee WHERE Last_name = 'Perera'));
+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys
+-----+-----+-----+-----+-----+
| 1 | SIMPLE     | employee | NULL      | ref  | FK_Manager,index_department,index_employee,index_employee_dept
| 1 | SIMPLE     | employee | NULL      | eq_ref| PRIMARY
+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql> create index index_dept using BTREE on Employee(department);
Query OK, 0 rows affected, 1 warning (0.23 sec)
Records: 0 Duplicates: 0 Warnings: 1

mysql> create index index_last_name using BTREE on Employee(last_name);
Query OK, 0 rows affected (0.18 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> explain(SELECT First_name, Last_name from employee where Department = 'D1001' AND Manager_ID IN (SELECT Emp_ID from employee WHERE Last_name = 'Perera'));
+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys
+-----+-----+-----+-----+-----+
| 1 | SIMPLE     | employee | NULL      | ref  | FK_Manager,index_department,index_employee,index_employee_dept,index_dept
| 1 | SIMPLE     | employee | NULL      | eq_ref| PRIMARY, index_last_name
+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

Nested Query 02

Without Indexing

```
Select MySQL 8.0 Command Line Client
-----+
2 rows in set, 1 warning (0.00 sec)

mysql> explain(SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw_material where Stock_quantity > 150 ));
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table      | partitions | type   | possible_keys | key      | key_len | ref      | rows | filtered | Extra      |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1  | SIMPLE      | raw_material | NULL      | ALL    | PRIMARY     | NULL     | NULL    |          | 5    | 33.33    | Using where |
| 1  | SIMPLE      | supplier    | NULL      | ref    | PRIMARY     | PRIMARY  | 42     | factory.raw_material.Raw_ID | 1    | 100.00    | Using index |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql> -
```

With Indexing

```
MySQL 8.0 Command Line Client
-----+
2 rows in set, 1 warning (0.00 sec)

mysql> explain(SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw_material where Stock_quantity > 150 ));
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table      | partitions | type   | possible_keys | key      | key_len | ref      | rows | filtered | Extra      |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1  | SIMPLE      | raw_material | NULL      | ALL    | PRIMARY     | NULL     | NULL    |          | 5    | 33.33    | Using where |
| 1  | SIMPLE      | supplier    | NULL      | ref    | PRIMARY     | PRIMARY  | 42     | factory.raw_material.Raw_ID | 1    | 100.00    | Using index |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql> create index index_raw_material using BTREE on supplier(raw_material);
Query OK, 0 rows affected (0.18 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> create index index_stock_quantity using BTREE on raw_material(stock_quantity);
Query OK, 0 rows affected (0.17 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> explain(SELECT Sup_name from supplier where Raw_material IN (SELECT Raw_ID from raw_material where Stock_quantity > 150 ));
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table      | partitions | type   | possible_keys | key      | key_len | ref      | rows | filtered | Extra      |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1  | SIMPLE      | supplier    | NULL      | index  | PRIMARY, index_raw_material | index_raw_material | 42     | NULL    | 4    | 100.00    | Using index |
| 1  | SIMPLE      | raw_material | NULL      | eq_ref | PRIMARY, index_stock_quantity | PRIMARY           | 42     | factory.supplier.Raw_material | 1    | 100.00    | Using where |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set, 1 warning (0.00 sec)

mysql>
```

Nested Query 03

Without Indexing

```
MySQL 8.0 Command Line Client
2 rows in set, 1 warning (0.00 sec)

mysql> explain(SELECT First_name, Last_name from worker where Department IN (SELECT Dept_ID from department where Dept_head IN (SELECT Emp_ID from employee where Age < 40)));
+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE     | department | NULL    | ALL   | PRIMARY, index_dept_id | NULL | NULL    | NULL | 8    | 100.00 | Using where
| 1 | SIMPLE     | employee   | NULL    | eq_ref | PRIMARY          | PRIMARY | 22     | factory.department.Dept_head | 1    | 33.33 | Using where
| 1 | SIMPLE     | worker    | NULL    | ref   | index_department_age, index_worker, index_worker_dept | index_department_age | 23   | factory.department.Dept_ID | 1    | 100.00 | Using index conditionals
+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.01 sec)

mysql>
```

With Indexing

```
MySQL 8.0 Command Line Client
2 rows in set, 1 warning (0.00 sec)

mysql> explain(SELECT First_name, Last_name from worker where Department IN (SELECT Dept_ID from department where Dept_head IN (SELECT Emp_ID from employee where Age < 40)));
+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE     | department | NULL    | ALL   | PRIMARY, index_dept_id | NULL | NULL    | NULL | 8    | 100.00 | Using where
| 1 | SIMPLE     | employee   | NULL    | eq_ref | PRIMARY          | PRIMARY | 22     | factory.department.Dept_head | 1    | 33.33 | Using where
| 1 | SIMPLE     | worker    | NULL    | ref   | index_department_age, index_worker, index_worker_dept | index_department_age | 23   | factory.department.Dept_ID | 1    | 100.00 | Using index conditionals
+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.01 sec)

mysql> create index index_age using BTREE on employee(age);
Query OK, 0 rows affected (0.17 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> create index index_dept_head using BTREE on department(dept_head);
Query OK, 0 rows affected (0.22 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> explain(SELECT First_name, Last_name from worker where Department IN (SELECT Dept_ID from department where Dept_head IN (SELECT Emp_ID from employee where Age < 40)));
+-----+-----+-----+-----+-----+-----+
| id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE     | department | NULL    | index | PRIMARY, index_dept_id, index_dept_head | index_dept_head | 43   | NULL    | 8    | 100.00 | Using where; Using index
| 1 | SIMPLE     | employee   | NULL    | eq_ref | PRIMARY, index_age          | PRIMARY | 22   | factory.department.Dept_head | 1    | 72.73 | Using where
| 1 | SIMPLE     | worker    | NULL    | ref   | index_department_age, index_worker, index_worker_dept | index_department_age | 23   | factory.department.Dept_ID | 1    | 100.00 | Using index conditionals
+-----+-----+-----+-----+-----+-----+
3 rows in set, 1 warning (0.01 sec)

mysql>
```