AMERICAN INTERNATIONAL UNIVERSITY–BANGLADESH (AIUB)

FACULTY OF SCIENCE & TECHNOLOGY

INTRODUCTION TO DATABASE

Spring 2023-2024

Section: U Group: 01

Project Title

FARMING MANAGEMENT SYSTEM

Supervised By

SYEEDA SHARMIN RAHMAN

Submitted By

Name	ID	Contribution
1. Nusrat Jahan Raina	23-50416-1	20%
2. Emon Biswas	23-50174-1	20%
3. Md. Redwanul Haque	23-50057-1	20%
4. Dipu Roy	23-50420-1	20%
5. Dip Khastagir	23-50346-1	20%

TABLE OF CONTENTS

TOPICS

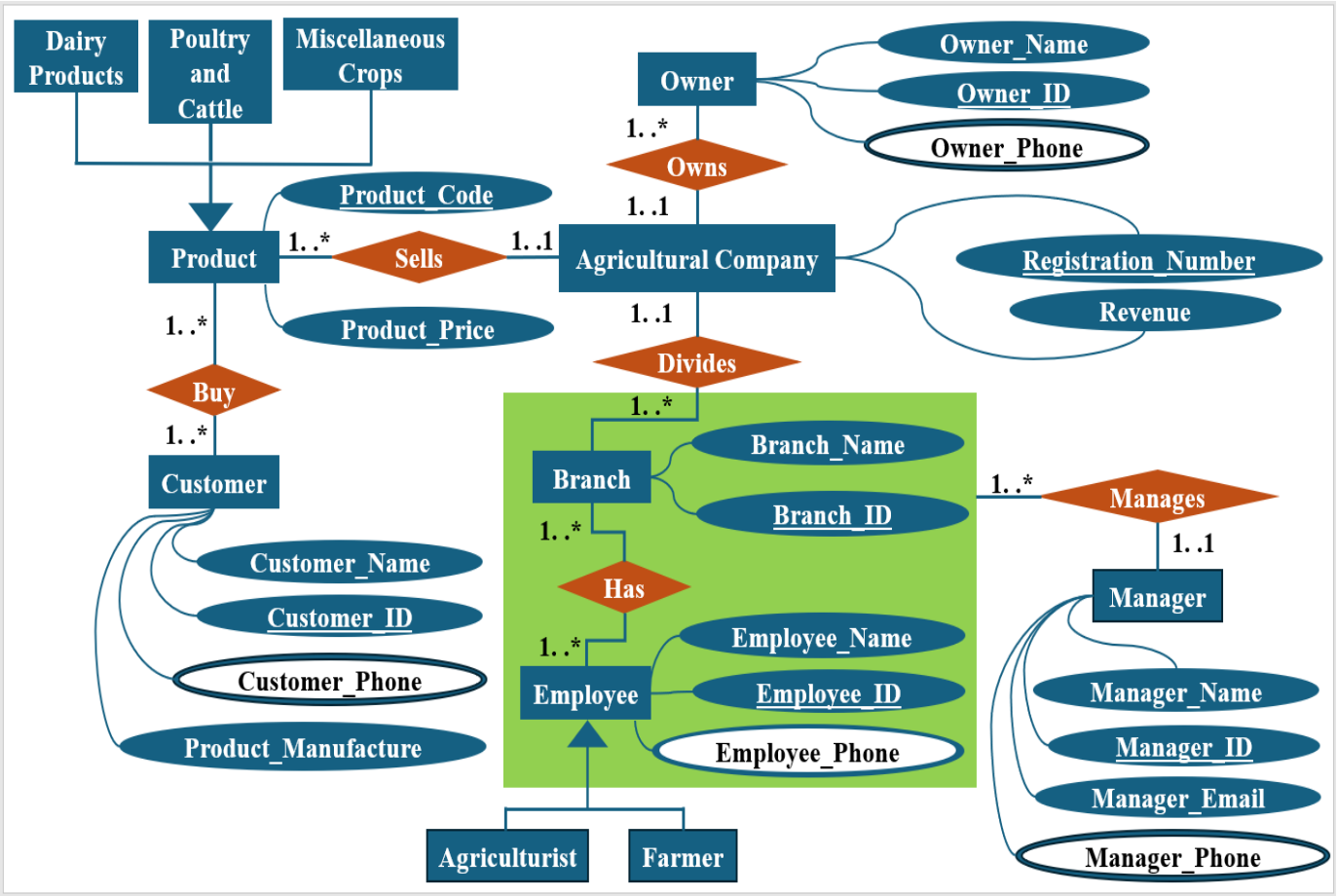
INTRODUCTION:

In an era marked by rapid technological advancements, the realm of agriculture stands as a cornerstone of global sustenance and economic stability. Our project, the Farming Management System, embodies a visionary approach aimed at revolutionizing agricultural operations with efficiency and precision. Rooted in the concept of modernization, this system endeavors to establish a unified platform for the comprehensive management of farming activities. Through the creation of a centralized database, the Farming Management System intends to streamline the intricate processes involved in agricultural endeavors, encompassing crop cultivation, livestock management, resource allocation, and logistical coordination. By seamlessly integrating crucial elements such as land utilization, crop rotation schedules, irrigation systems, and inventory tracking, this innovative solution aspires to empower farmers with the tools necessary to optimize yields, minimize wastage, and foster sustainable agricultural practices. Embracing this paradigm shift promises not only heightened productivity and profitability but also facilitates greater resilience in the face of evolving environmental and market challenges. Through the successful implementation of the Farming Management System, the agricultural sector stands poised to harness the transformative potential of technology, ensuring a bountiful harvest for generations to come.

SCENARIO:

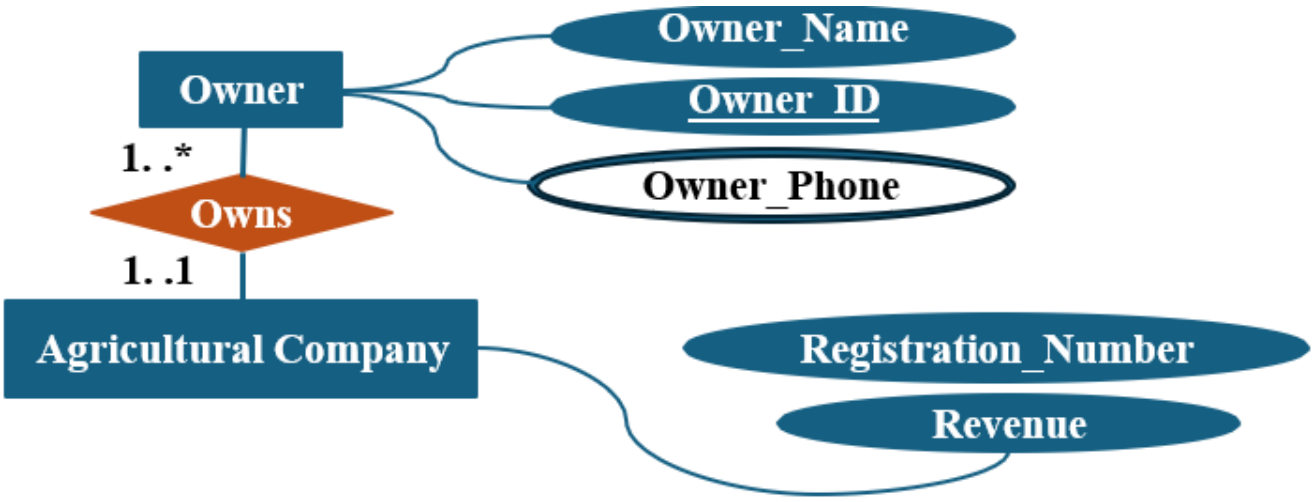
In a farming management system of Bangladesh, many owners can own an agricultural company. Agricultural company contains registration number and revenue. Owner is uniquely identified by owner ID, phone number and name. The agricultural company divides into several branches. Every branch contains a branch ID and address. Manager manages all branches and employees of the company. Every manager has their manager ID, name, e-mail, phone number. The company has different types of employees. Employee can be classified as agriculturist and farmer. Employee can be specified by employee ID, name and phone number. Company sells their farming product to the people. Farming products can be classified as miscellaneous crops, poultry and cattle, dairy products. Product has a product code and price. Customer can buy farming products from the company. Every customer can be identified by customer ID, name, phone number and manufacture date of farming products.

ER DIAGRAM:


Dairy Products
Poultry and Cattle
Miscellaneous Crops
Owner
Owner_Name
Owner_ID
Owner_Phone
1..*
Owns
1..1
Product_Code
Product
1..*
Sells
1..1
Agricultural Company
Registration_Number
Revenue
1..*
Product_Price
1..1
Divides
Buy
1..*
1..*
Customer
Branch
Branch_Name
Branch_ID
1..*
Manages
1..1
Manager
1..*
Has
Customer_Name
Customer_ID
Customer_Phone
Product_Manufacture
1..*
Employee
Employee_Name
Employee_ID
Employee_Phone
Manager_Name
Manager_ID
Manager_Email
Manager_Phone
Agriculturist
Farmer

NORMALIZATION:

1. Owner – owns – Agricultural Company



Relation: Many to One

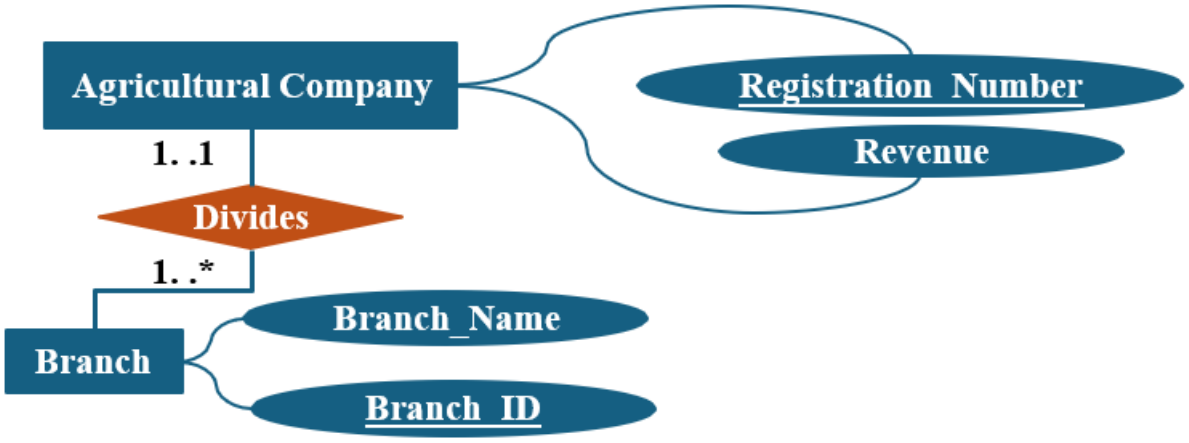
UNF: Owner_Name, Owner_ID, Owner_Phone, Registration_Number, Revenue

1NF: Owner_Phone, Owner_Name, Owner_ID, Registration_Number, Revenue

2NF: 1) Owner_ID (Primary Key), Owner_Name, Owner_Phone, Registration_Number (Foreign Key)
2) Registration_Number (Primary Key), Revenue

3NF: 1) Owner_ID (Primary Key), Owner_Name, Owner_Phone, Registration_Number (Foreign Key)
2) Registration_Number (Primary Key), Revenue

2. Agricultural Company – divides – Branch



Relation: One to Many

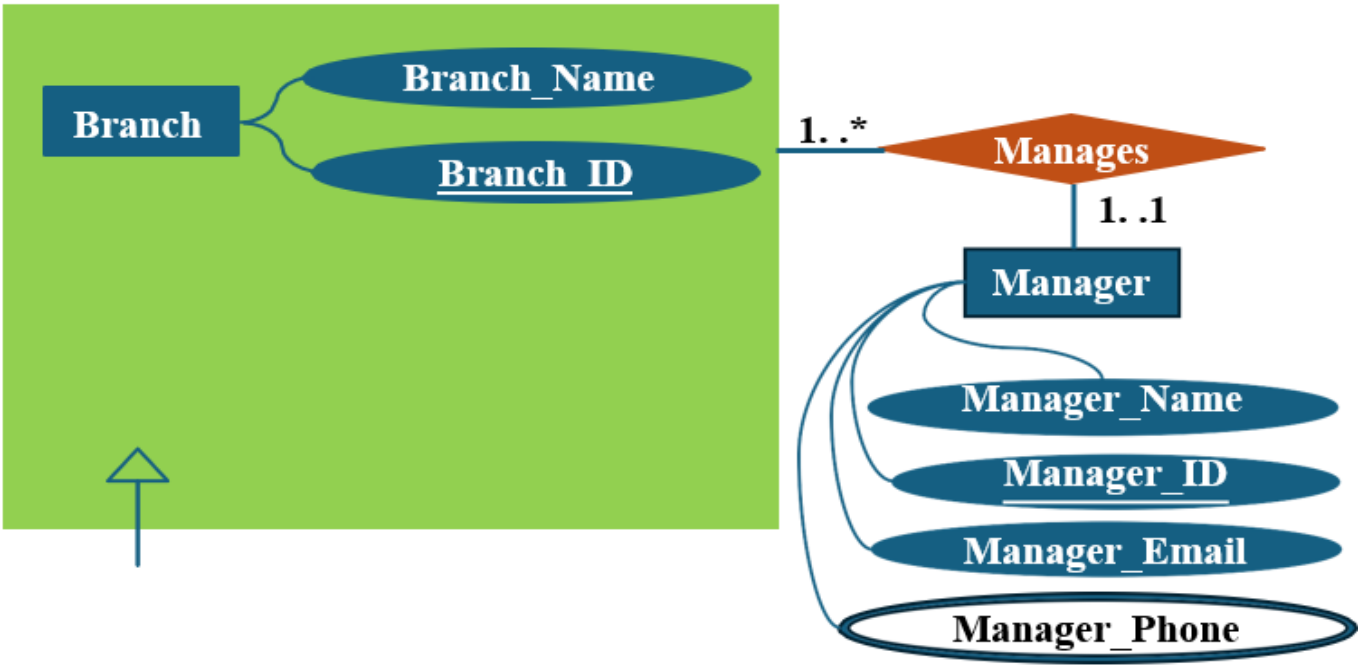
UNF: Registration_Number, Revenue, Branch_ID, Branch_Name

1NF: Registration_Number, Revenue, Branch_ID, Branch_Name

2NF: 1) Registration_Number (Primary Key), Revenue
2) Branch_ID (Primary Key), Branch_Name, Registration_Number (Foreign Key)

3NF: 1) Registration_Number (Primary Key), Revenue
2) Branch_ID (Primary Key), Branch_Name, Registration_Number (Foreign Key)

3. Branch – manages – Manager



Relation: One to Many

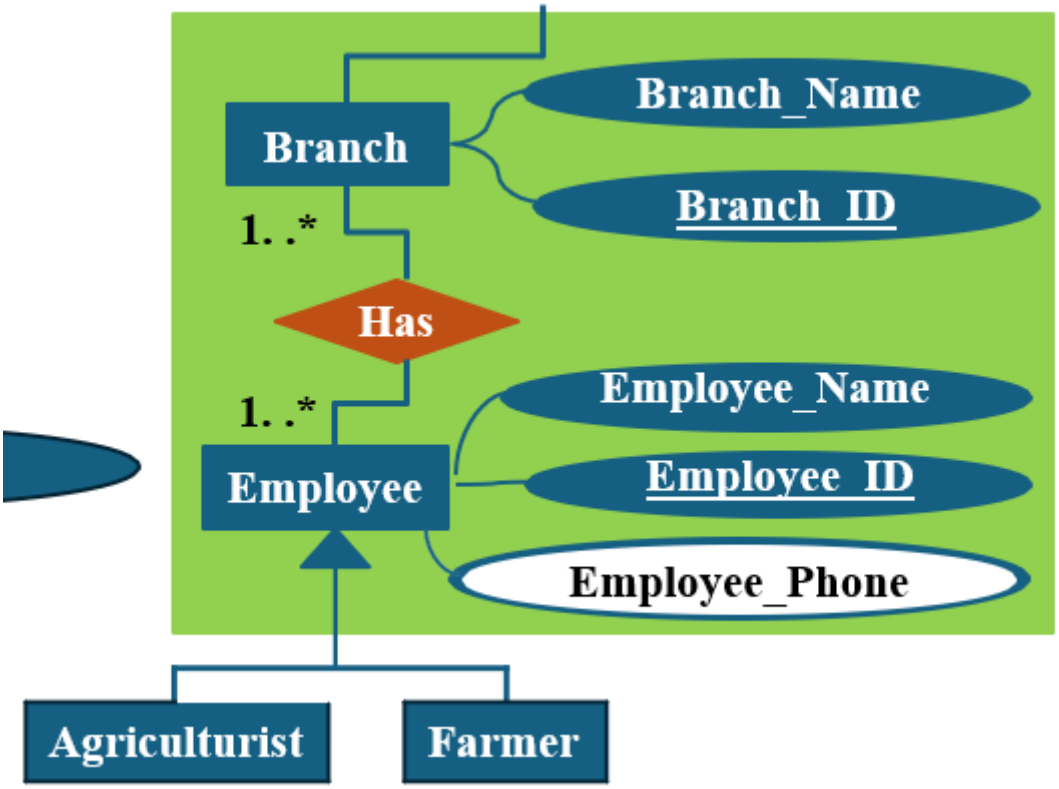
UNF: Branch_ID, Branch_Name, Manager_Name, Manager_ID, Manager_Email, Manager_Phone, Employee_ID, Employee_Name, Employee_Phone.

1NF: Branch_ID, Branch_Name, Manager_Name, Manager_ID, Manager_Email, Manager_Phone, Employee_ID, Employee_Name, Employee_Phone.

2NF: 1) Manager_ID (Primary Key), Manager_Name, Manager_Email, Manager_Phone.
2) Branch_ID (Primary Key), Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID (Foreign Key)

3NF: 1) Manager_ID (Primary Key), Manager_Name, Manager_Email, Manager_Phone.
2) Branch_ID (Primary Key), Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID (Foreign Key)

4. Branch – has – Employee



Relation: Many to Many

UNF: Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone

1NF: Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone

2NF: 1) Branch_ID (Primary Key), Branch_Name
2) Employee_ID (Primary Key), Employee_Name, Employee_Phone
3) Branch_ID (Primary Key), Employee_ID (Foreign Key)

3NF: 1) Branch_ID (Primary Key), Branch_Name
2) Employee_ID (Primary Key), Employee_Name, Employee_Phone
3) Branch_ID (Primary Key), Employee_ID (Foreign Key)

5. Agricultural Company – sells – Product

Relation: One to Many

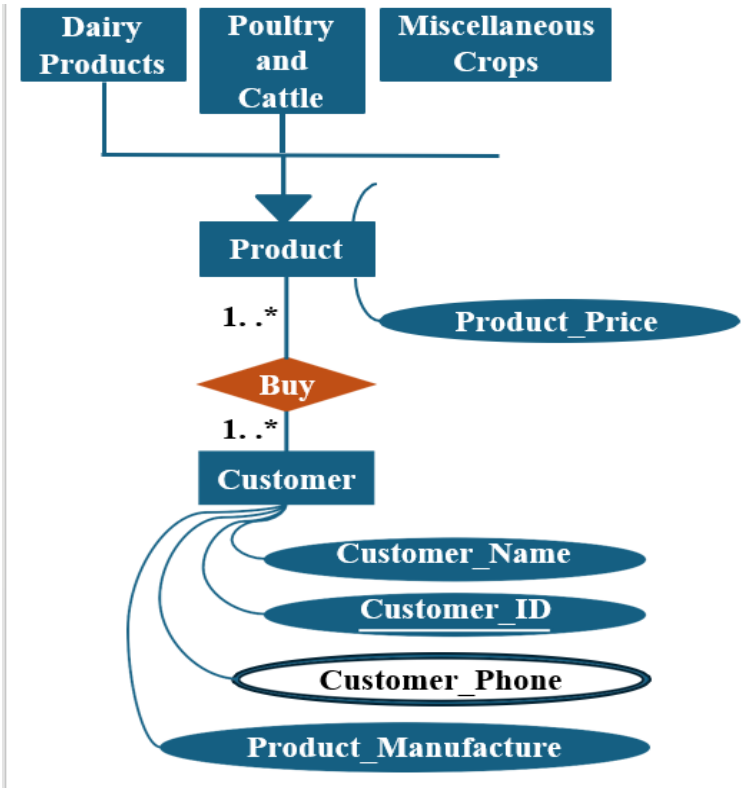
UNF: Registration_Number, Revenue, Product_Code, Product_Price

1NF: Registration_Number, Revenue, Product_Code, Product_Price

2NF: 1) Product_Code (Primary Key), Product_Price, Registration_Number (Foreign Key)
2) Registration_Number (Primary Key), Revenue

3NF: 1) Product_Code (Primary Key), Product_Price, Registration_Number (Foreign Key)
2) Registration_Number (Primary Key), Revenue

6. Customer – buy - Product



Relation: Many to Many

UNF: Product_Code, Product_Price, Product_Manufacture, Customer_Phone, Customer_Name, Customer_ID.

1NF: Product_Code, Product_Price, Product_Manufacture, Customer_Phone, Customer_Name, Customer_ID.

2NF: 1) Customer_ID (Primary Key), Product_Manufacture, Customer_Phone, Customer_Name.
2) Product_Code (Primary Key), Product_Price.
3) Customer_ID (Primary Key), Product_Code (Foreign Key)

3NF: 1) Customer_ID (Primary Key), Product_Manufacture, Customer_Phone, Customer_Name.
2) Product_Code (Primary Key), Product_Price.
3) Customer_ID (Primary Key), Product_Code (Foreign Key

FINALIZATION:

i. Owner_ID (Primary Key), Owner_Name, Owner_Phone, Registration_Number (Foreign Key).
ii. Registration_Number (Primary Key), Revenue.
~~iii. Registration_Number (Primary Key), Revenue.~~
iv. Branch_ID (Primary Key), Branch_Name, Registration_Number (Foreign Key).
v. Branch_ID (Primary Key), Branch_Name.
vi. Manager_ID (Primary Key), Manager_Name, Manager_Email, Manager_Phone.
vii. Branch_ID (Primary Key), Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID (Foreign Key)
viii. Employee_ID (Primary Key), Employee_Name, Employee_Phone.
ix. Branch_ID (Primary Key), Employee_ID (Foreign Key).
x. Customer_ID (Primary Key), Product_Manufacture, Customer_Phone, Customer_Name.
xi. Product_Code (Primary Key), Product_Price.
xii. Customer_ID (Primary Key), Product_Code (Foreign Key).
xiii. Product_Code (Primary Key), Product_Price, Registration_Number (Foreign Key).
~~xiv. Registration_Number (Primary Key), Revenue.~~

OPTIMIZATION:

i. Owner_ID (Primary Key), Owner_Name, Owner_Phone, Registration_Number (Foreign Key) – (Owns)
ii. Registration_Number (Primary Key), Revenue – (Agricultural Company)
iii. Branch_ID (Primary Key), Branch_Name, Registration_Number (Foreign Key) – (Divides)
iv. Branch_ID (Primary Key), Branch_Name – (Branch)
v. Manager_ID (Primary Key), Manager_Name, Manager_Email, Manager_Phone – (Manager)
vi. Employee_ID (Primary Key), Employee_Name, Employee_Phone – (Employee)
vii. Branch_ID (Primary Key), Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID (Foreign Key) – (Manages)
viii. Branch_ID (Primary Key), Employee_ID (Foreign Key) – (Has)
ix. Customer_ID (Primary Key), Product_Manufacture, Customer_Phone, Customer_Name – (Customer)
x. Product_Code (Primary Key), Product_Price – (Product)
xi. Customer_ID (Primary Key), Product_Code (Foreign Key) – (Buy)
xii. Product_Code (Primary Key), Product_Price, Registration_Number (Foreign Key) – (Sells).

TABLE CREATION:

1. Table: Agricultural Company

CREATE TABLE Agricultural_Company (
Registration_Number NUMBER PRIMARY KEY,
Revenue NUMBER
);

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Agricultural_Company (
    Registration_Number NUMBER PRIMARY KEY,
    Revenue NUMBER
);

DESCRIBE Agricultural_Company;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **AGRICULTURAL_COMPANY**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
AGRICULTURAL_COMPANY	REGISTRATION_NUMBER	Number	-	-	-	1	-	-	-
	REVENUE	Number	-	-	-	-	✓	-	-

1 - 2

2. Table : Owns

```
CREATE TABLE Owns (
    Owner_ID NUMBER PRIMARY KEY,
    Owner_Name VARCHAR2(100),
    Owner_Phone VARCHAR2(20),
    Registration_Number NUMBER,
    FOREIGN KEY (Registration_Number) REFERENCES
Agricultural_Company(Registration_Number)
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Owns (
    Owner_ID NUMBER PRIMARY KEY,
    Owner_Name VARCHAR2(100),
    Owner_Phone VARCHAR2(20),
    Registration_Number NUMBER,
    FOREIGN KEY (Registration_Number) REFERENCES Agricultural_Company(Registration_Number)
);

DESCRIBE Owns;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **OWNS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
OWNS	OWNER_ID	Number	-	-	-	1	-	-	-
	OWNER_NAME	Varchar2	100	-	-	-	✓	-	-
	OWNER_PHONE	Varchar2	20	-	-	-	✓	-	-
	REGISTRATION_NUMBER	Number	-	-	-	-	✓	-	-

1 - 4

3. Table : Branch

```
CREATE TABLE Branch (
    Branch_ID NUMBER PRIMARY KEY,
    Branch_Name VARCHAR2(100)
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Branch (
    Branch_ID NUMBER PRIMARY KEY,
    Branch_Name VARCHAR2(100)
);

DESCRIBE Branch;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **BRANCH**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
BRANCH	BRANCH_ID	Number	-	-	-	1	-	-	-
	BRANCH_NAME	Varchar2	100	-	-	-	✓	-	-
									1 - 2

4. Table : Divides

```
CREATE TABLE Divides (
    Branch_ID NUMBER PRIMARY KEY,
    Branch_Name VARCHAR2(100),
    Registration_Number NUMBER,
    FOREIGN KEY (Registration_Number) REFERENCES
Agricultural_Company(Registration_Number)
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Divides (
    Branch_ID NUMBER PRIMARY KEY,
    Branch_Name VARCHAR2(100),
    Registration_Number NUMBER,
    FOREIGN KEY (Registration_Number) REFERENCES Agricultural_Company(Registration_Number)
);

DESCRIBE Divides;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **DIVIDES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DIVIDES	BRANCH_ID	Number	-	-	-	1	-	-	-
	BRANCH_NAME	Varchar2	100	-	-	-	✓	-	-
	REGISTRATION_NUMBER	Number	-	-	-	-	✓	-	-

1 - 3

5. Table : Manager

```
CREATE TABLE Manager (
    Manager_ID NUMBER PRIMARY KEY,
    Manager_Name VARCHAR2(100),
    Manager_Email VARCHAR2(100),
    Manager_Phone VARCHAR2(20)
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Manager (
    Manager_ID NUMBER PRIMARY KEY,
    Manager_Name VARCHAR2(100),
    Manager_Email VARCHAR2(100),
    Manager_Phone VARCHAR2(20)
);

DESCRIBE Manager;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **MANAGER**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MANAGER	MANAGER_ID	Number	-	-	-	1	-	-	-
	MANAGER_NAME	Varchar2	100	-	-	-	✓	-	-
	MANAGER_EMAIL	Varchar2	100	-	-	-	✓	-	-
	MANAGER_PHONE	Varchar2	20	-	-	-	✓	-	-
									1 - 4

6. Table : Employee

```
CREATE TABLE Employee (
    Employee_ID NUMBER PRIMARY KEY,
    Employee_Name VARCHAR2(100),
    Employee_Phone VARCHAR2(20)
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Employee (
    Employee_ID NUMBER PRIMARY KEY,
    Employee_Name VARCHAR2(100),
    Employee_Phone VARCHAR2(20)
);

DESCRIBE Employee;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **EMPLOYEE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMPLOYEE	EMPLOYEE_ID	Number	-	-	-	1	-	-	-
	EMPLOYEE_NAME	Varchar2	100	-	-	-	✓	-	-
	EMPLOYEE_PHONE	Varchar2	20	-	-	-	✓	-	-

1 - 3

7. Table : Manages

```
CREATE TABLE Manages (
    Branch_ID NUMBER PRIMARY KEY,
    Branch_Name VARCHAR2(100),
    Employee_ID NUMBER,
    Employee_Name VARCHAR2(100),
    Employee_Phone VARCHAR2(20),
    Manager_ID NUMBER,
    FOREIGN KEY (Manager_ID) REFERENCES Manager(Manager_ID)
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Manages (
    Branch_ID NUMBER PRIMARY KEY,
    Branch_Name VARCHAR2(100),
    Employee_ID NUMBER,
    Employee_Name VARCHAR2(100),
    Employee_Phone VARCHAR2(20),
    Manager_ID NUMBER,
    FOREIGN KEY (Manager_ID) REFERENCES Manager(Manager_ID)
);

DESCRIBE Manages;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **MANAGES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MANAGES	BRANCH_ID	Number	-	-	-	1	-	-	-
	BRANCH_NAME	Varchar2	100	-	-	-	✓	-	-
	EMPLOYEE_ID	Number	-	-	-	-	✓	-	-
	EMPLOYEE_NAME	Varchar2	100	-	-	-	✓	-	-
	EMPLOYEE_PHONE	Varchar2	20	-	-	-	✓	-	-
	MANAGER_ID	Number	-	-	-	-	✓	-	-

1 - 6

8. Table : Has

```
CREATE TABLE Has (
    Branch_ID NUMBER PRIMARY KEY,
    Employee_ID NUMBER,
    FOREIGN KEY (Branch_ID) REFERENCES Branch(Branch_ID),
    FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID)
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Has (
    Branch_ID NUMBER PRIMARY KEY,
    Employee_ID NUMBER,
    FOREIGN KEY (Branch_ID) REFERENCES Branch(Branch_ID),
    FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID)
);

DESCRIBE Has;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **HAS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
HAS	BRANCH_ID	Number	-	-	-	1	-	-	-
	EMPLOYEE_ID	Number	-	-	-	-	✓	-	-
									1 - 2

9. Table : Customer

```
CREATE TABLE Customer (
    Customer_ID NUMBER PRIMARY KEY,
    Product_Manufacture VARCHAR2(100),
    Customer_Phone VARCHAR2(20),
    Customer_Name VARCHAR2(100)
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Customer (
    Customer_ID NUMBER PRIMARY KEY,
    Product_Manufacture VARCHAR2(100),
    Customer_Phone VARCHAR2(20),
    Customer_Name VARCHAR2(100)
);

DESCRIBE Customer;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **CUSTOMER**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CUSTOMER	CUSTOMER_ID	Number	-	-	-	1	-	-	-
	PRODUCT_MANUFACTURE	Varchar2	100	-	-	-	✓	-	-
	CUSTOMER_PHONE	Varchar2	20	-	-	-	✓	-	-
	CUSTOMER_NAME	Varchar2	100	-	-	-	✓	-	-
									1 - 4

10. Table : Product

```
CREATE TABLE Product (
    Product_Code NUMBER PRIMARY KEY,
    Product_Price NUMBER
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Product (
    Product_Code NUMBER PRIMARY KEY,
    Product_Price NUMBER
);

DESCRIBE Product;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **PRODUCT**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PRODUCT	PRODUCT_CODE	Number	-	-	-	1	-	-	-
	PRODUCT_PRICE	Number	-	-	-	-	✓	-	-

1 - 2

11. Table: Buy

```
CREATE TABLE Buy (
    Customer_ID NUMBER,
    Product_Code NUMBER,
    FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID),
    FOREIGN KEY (Product_Code) REFERENCES Product(Product_Code)
);
```

Home > SQL > SQL Commands

Autocommit Display 10

```
CREATE TABLE Buy (
    Customer_ID NUMBER,
    Product_Code NUMBER,
    FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID),
    FOREIGN KEY (Product_Code) REFERENCES Product(Product_Code)
);

DESCRIBE Product;
```

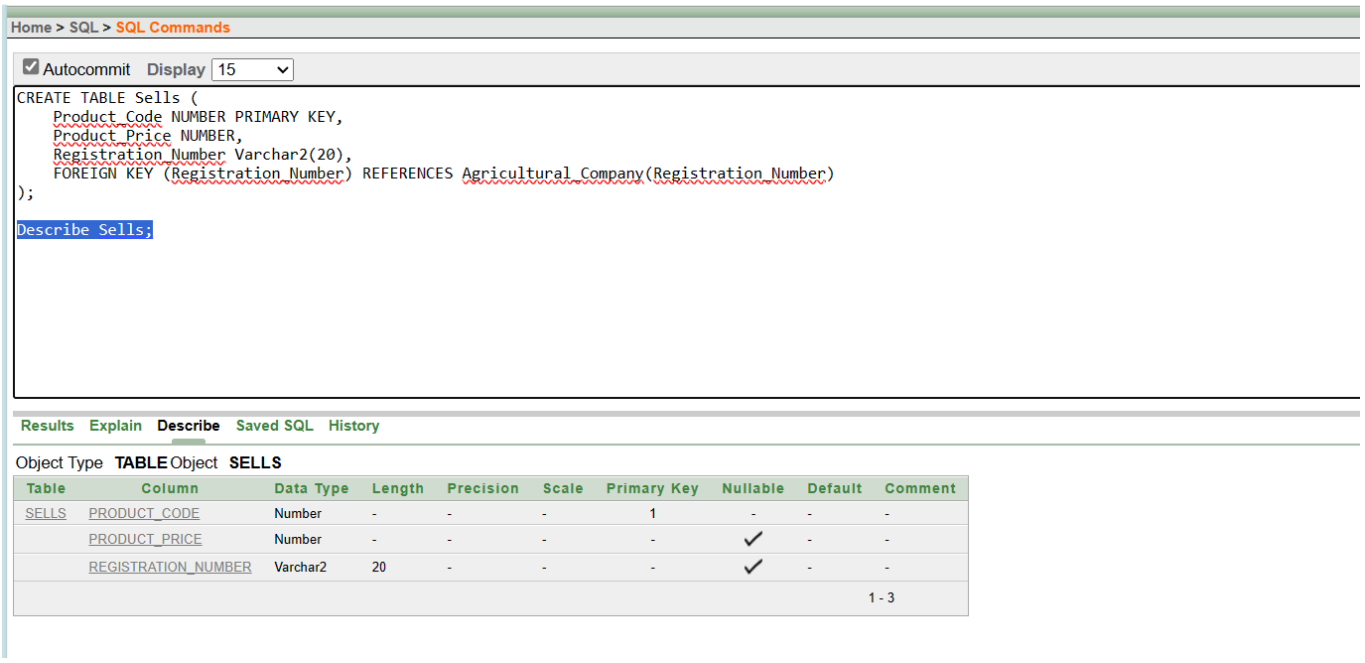
Results Explain Describe Saved SQL History

Object Type **TABLE** Object **PRODUCT**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PRODUCT	PRODUCT_CODE	Number	-	-	-	1	-	-	-
	PRODUCT_PRICE	Number	-	-	-	-	✓	-	-
									1 - 2

12. Table : Sells

```
CREATE TABLE Sells (
   Product_Code NUMBER PRIMARY KEY,
   Product_Price NUMBER,
   Registration_Number Varchar2(20),
   FOREIGN KEY (Registration_Number) REFERENCES
Agricultural_Company(Registration_Number)
);
```

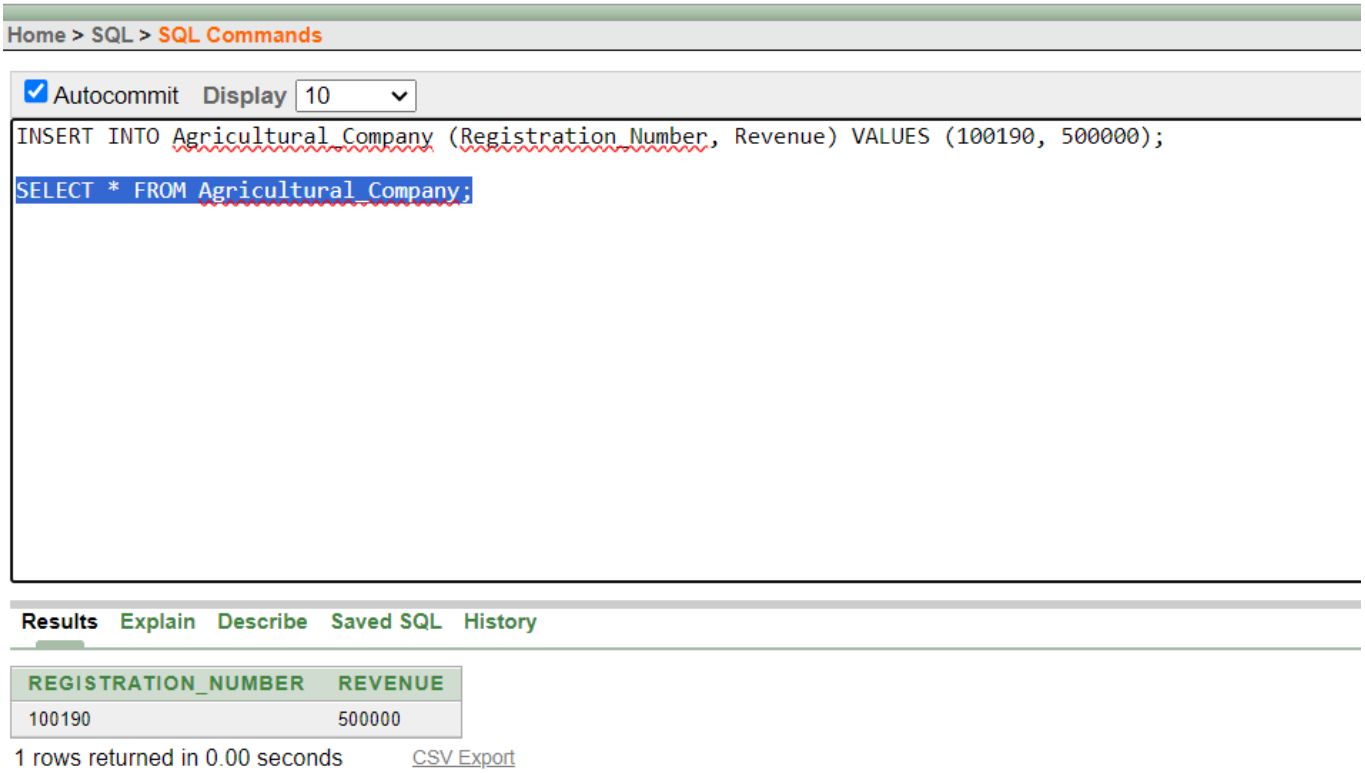


DATA INSERTION:

1. Information: Agricultural Company

INSERT INTO Agricultural_Company (Registration_Number, Revenue) VALUES (100190, 500000);

SELECT * FROM Agricultural_Company;



2. Information: Owns

INSERT INTO Owns (Owner_ID, Owner_Name, Owner_Phone, Registration_Number) VALUES (9001, 'DIPU ROY', '01316-008929', 100190);

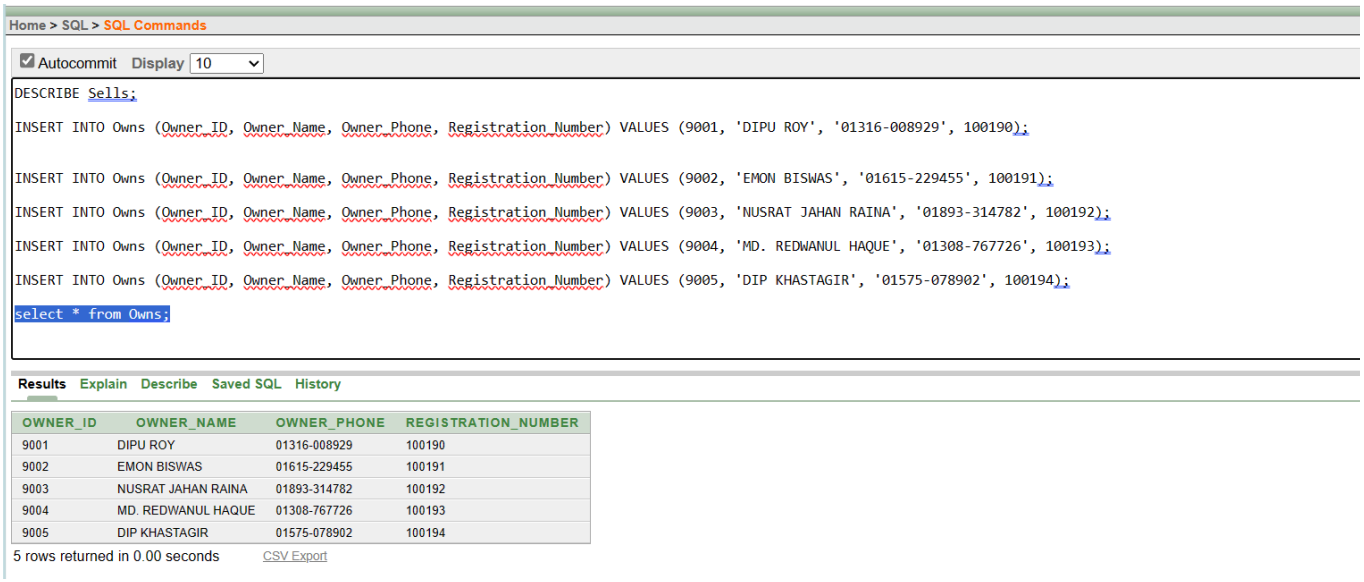
INSERT INTO Owns (Owner_ID, Owner_Name, Owner_Phone, Registration_Number) VALUES (9002, 'EMON BISWAS', '01615-229455', 100191);

INSERT INTO Owns (Owner_ID, Owner_Name, Owner_Phone, Registration_Number) VALUES (9003, 'NUSRAT JAHAN RAINA', '01893-314782', 100192);

INSERT INTO Owns (Owner_ID, Owner_Name, Owner_Phone, Registration_Number) VALUES (9004, 'MD. REDWANUL HAQUE', '01308-767726', 100193);

INSERT INTO Owns (Owner_ID, Owner_Name, Owner_Phone, Registration_Number) VALUES (9005, 'DIP KHASTAGIR', '01575-078902', 100194);

SELECT * FROM Owns;



3. Information: Divides

INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (101, 'RAJSHAHI', 100190);

INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (102, 'MAGURA', 100190);

INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (103, 'MUNSHIGANJ', 100190);

INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (104, 'FENI', 100190);

INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (105, 'CHITTAGONG', 100190);

SELECT * FROM Divides;

Home > SQL > SQL Commands

Autocommit Display 10

```
INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (101, 'RAJSHAHI', 100190);

INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (102, 'MAGURA', 100190);

INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (103, 'MUNSHIGANJ', 100190);

INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (104, 'FENI', 100190);

INSERT INTO Divides (Branch_ID, Branch_Name, Registration_Number) VALUES (105, 'CHITTAGONG', 100190);

SELECT * FROM Divides;
```

Results Explain Describe Saved SQL History

BRANCH_ID	BRANCH_NAME	REGISTRATION_NUMBER
101	RAJSHAHI	100190
102	MAGURA	100190
103	MUNSHIGANJ	100190
104	FENI	100190
105	CHITTAGONG	100190

5 rows returned in 0.00 seconds CSV Export

4. Information: Branch

INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (101, 'RAJSHAHI');

INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (102, 'MAGURA');

INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (103, 'MUNSHIGANJ');

INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (104, 'FENI');

INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (105, 'CHITTAGONG');

SELECT * FROM Branch;

Home > SQL > SQL Commands

Autocommit Display 10

```
INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (101, 'RAJSHAHI');

INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (102, 'MAGURA');

INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (103, 'MUNSHIGANJ');

INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (104, 'FENI');

INSERT INTO Branch (Branch_ID, Branch_Name) VALUES (105, 'CHITTAGONG');

SELECT * FROM Branch;
```

Results Explain Describe Saved SQL History

BRANCH_ID	BRANCH_NAME
101	RAJSHAHI
102	MAGURA
103	MUNSHIGANJ
104	FENI
105	CHITTAGONG

5 rows returned in 0.00 seconds CSV Export

5. Information: Manager

INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Phone) VALUES (101, 'David Wilson ', 'david@example.com', '01388-445978');

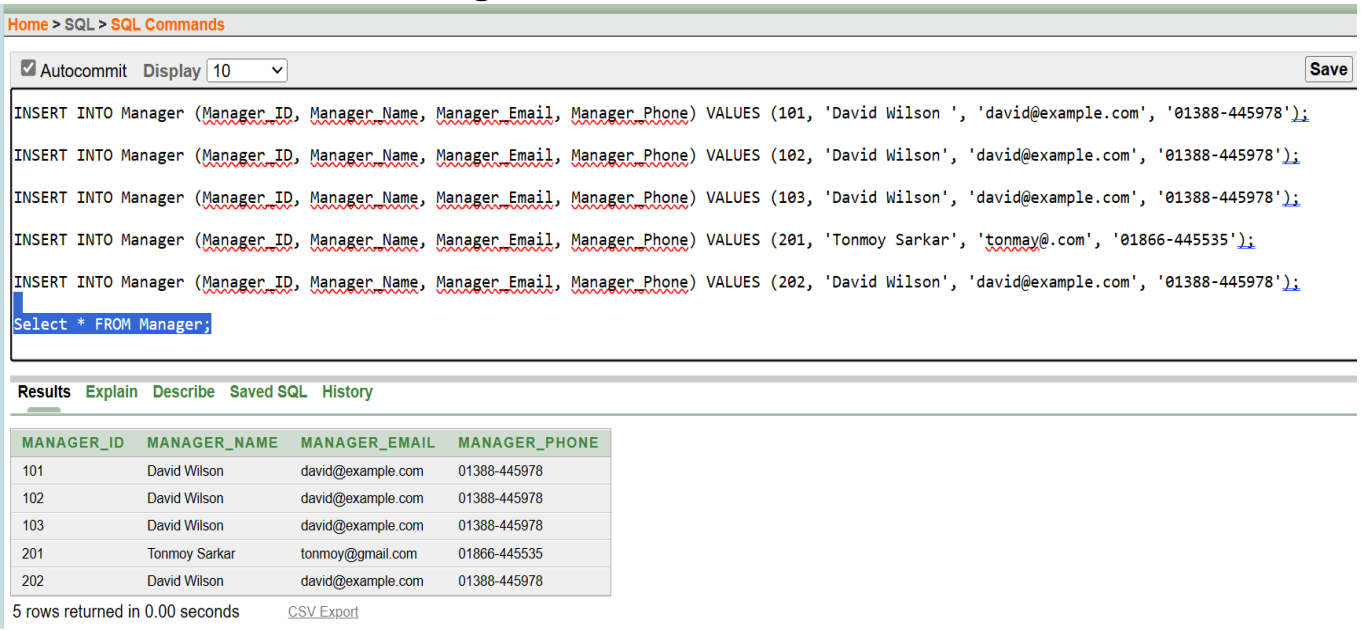
INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Phone) VALUES (102, 'David Wilson', 'david@example.com', '01388-445978');

INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Phone) VALUES (103, 'David Wilson', 'david@example.com', '01388-445978');

INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Phone) VALUES (201, 'Tonmoy Sarkar', 'tonmay@.com', '01866-445535');

INSERT INTO Manager (Manager_ID, Manager_Name, Manager_Email, Manager_Phone) VALUES (202, 'David Wilson', 'david@example.com', '01388-445978');

Select * FROM Manager;



6. Information: Employee

```
INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010001, 'Michael Johnson', '01789-345909');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010002, 'Rachel White', '01234-456984');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010003, 'Daniel Davis', '01345-226778');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010004, 'Olivia Taylor', '01987-348229');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010005, 'Matthew Martinez', '01892-493111');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010006, 'Debojit Khastagir', '01937-348229');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010007, 'Amit Sarkar', '01899-493511');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010008, 'Anamul Haque', '01987-3487356');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010009, 'Md. Golam Rabbani', '01892-493381');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010010, 'Anika Tabassum', '01767-348221');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone)
VALUES (3010011, 'Nirjhor Rahman', '01882-873111');

SELECT * FROM Employee;
```

Home > SQL > SQL Commands

Autocommit Display 10

```
INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010001, 'Michael Johnson', '01789-345909');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010002, 'Rachel White', '01234-456984');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010003, 'Daniel Davis', '01345-226778');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010004, 'Olivia Taylor', '01987-348229');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010005, 'Matthew Martinez', '01892-493111');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010006, 'Debojit Khastagir', '01937-348229');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010007, 'Amit Sarkar', '01899-493511');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010008, 'Anamul Haque', '01987-3487356');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010009, 'Md. Golam Rabbani', '01892-493381');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010010, 'Anika Tabassum', '01767-348221');

INSERT INTO Employee (Employee_ID, Employee_Name, Employee_Phone) VALUES (3010011, 'Nirjhor Rahman', '01882-873111');

SELECT * FROM Employee;
```

Results Explain Describe Saved SQL History

EMPLOYEE_ID	EMPLOYEE_NAME	EMPLOYEE_PHONE
3010001	Michael Johnson	01789-345909
3010002	Rachel White	01234-456984
3010003	Daniel Davis	01345-226778
3010004	Olivia Taylor	01987-348229
3010005	Matthew Martinez	01892-493111
3010006	Debojit Khastagir	01937-348229
3010007	Amit Sarkar	01899-493511
3010008	Anamul Haque	01987-3487356
3010009	Md. Golam Rabbani	01892-493381
3010010	Anika Tabassum	01767-348221
More than 10 rows available. Increase rows selector to view more rows.		

10 rows returned in 0.02 seconds CSV Export

7. Information: Manages

INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (108, 'RAJSHAHI', 3010011, 'Nirjhor Rahman', '01882-873111', 101);

INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (109, 'MAGURA', 3010001, 'Michael Johnson', '01789-345909', 102);

INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (110, 'MAGURA', 3010002, 'Rachel White', '01234-456984', 103);

INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (101, 'RAJSHAHI', 3010002, 'Debojit khastagir', '01937-348229', 201);

INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (102, 'RAJSHAHI', 3010008, 'Anamul Haque', '01987-3487356', 201);

INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (103, 'RAJSHAHI', 3010009, 'Md. Golam Rabbani', '01892-493381', 201);

INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (105, 'RAJSHAHI', 3010011, 'Nirjhor Rahman', '01882-873111', 201);

Select * FROM Manages;

Home > SQL > SQL Commands

Autocommit Display 10 Save Run

```
456984', 103);
INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (101, 'RAJSHAHI', 3010002, 'Debojit khastagir',
'01937-348229', 201);
INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (102, 'RAJSHAHI', 3010008, 'Anamul Haque',
'01987-3487356', 201);
INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (103, 'RAJSHAHI', 3010009, 'Md. Golam Rabbani',
'01892-493381', 201);
INSERT INTO Manages (Branch_ID, Branch_Name, Employee_ID, Employee_Name, Employee_Phone, Manager_ID) VALUES (105, 'RAJSHAHI', 3010011, 'Nirjhor Rahman',
'01882-873111', 201);
Select * FROM Manages;
```

Results Explain Describe Saved SQL History

BRANCH_ID	BRANCH_NAME	EMPLOYEE_ID	EMPLOYEE_NAME	EMPLOYEE_PHONE	MANAGER_ID
108	RAJSHAHI	3010011	Nirjhor Rahman	01882-873111	101
109	MAGURA	3010001	Michael Johnson	01789-345909	102
110	MAGURA	3010002	Rachel White	01234-456984	103
101	RAJSHAHI	3010006	Debojit Khastagir	01937-348229	201
102	RAJSHAHI	3010008	Anamul Haque	01987-3487356	201
103	RAJSHAHI	3010009	Md. Golam Rabbani	01892-493381	201
105	RAJSHAHI	3010011	Nirjhor Rahman	01882-873111	201

7 rows returned in 0.02 seconds CSV Export

8. Information: HAS

INSERT INTO Has (Branch_ID, Employee_ID)
VALUES (101, 3010001);

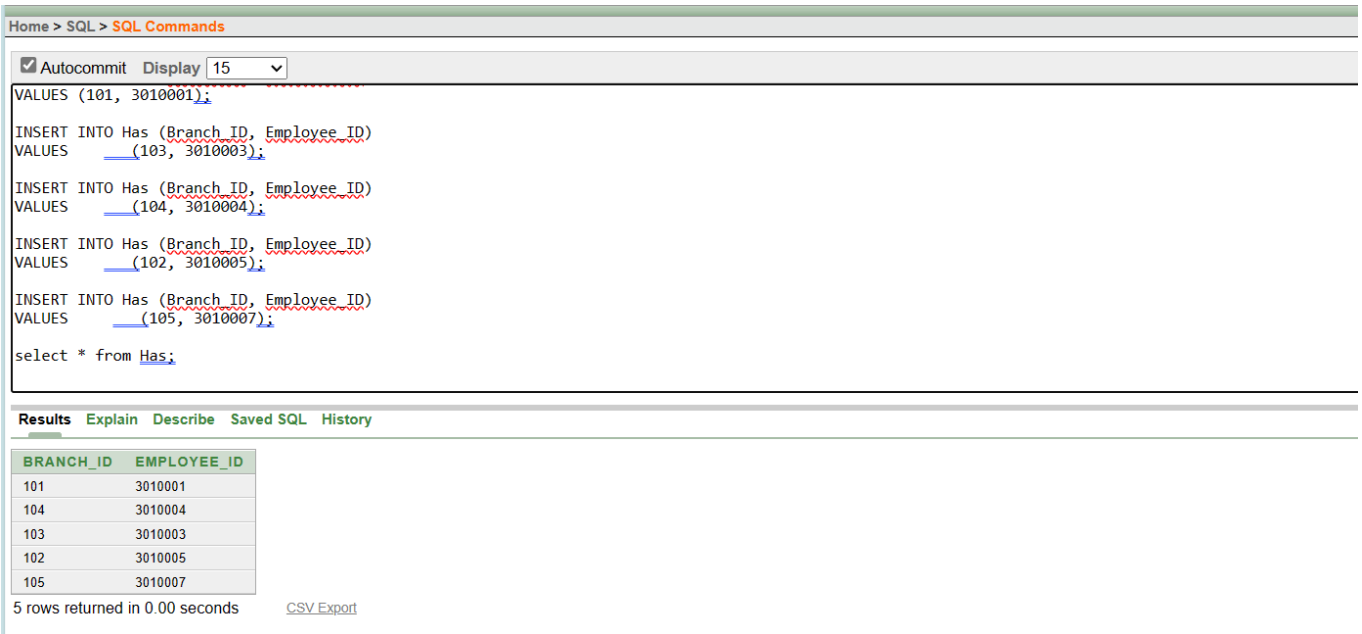
INSERT INTO Has (Branch_ID, Employee_ID)
VALUES (103, 3010003);

INSERT INTO Has (Branch_ID, Employee_ID)
VALUES (104, 3010004);

INSERT INTO Has (Branch_ID, Employee_ID)
VALUES (102, 3010005);

INSERT INTO Has (Branch_ID, Employee_ID)
VALUES (105, 3010007);

select * from Has;



9. Information: Customer

INSERT INTO Customer (Customer_ID, Product_Manufacture, Customer_Phone, Customer_Name)
VALUES (401, 'Garlic', '1112223333', 'Pial');

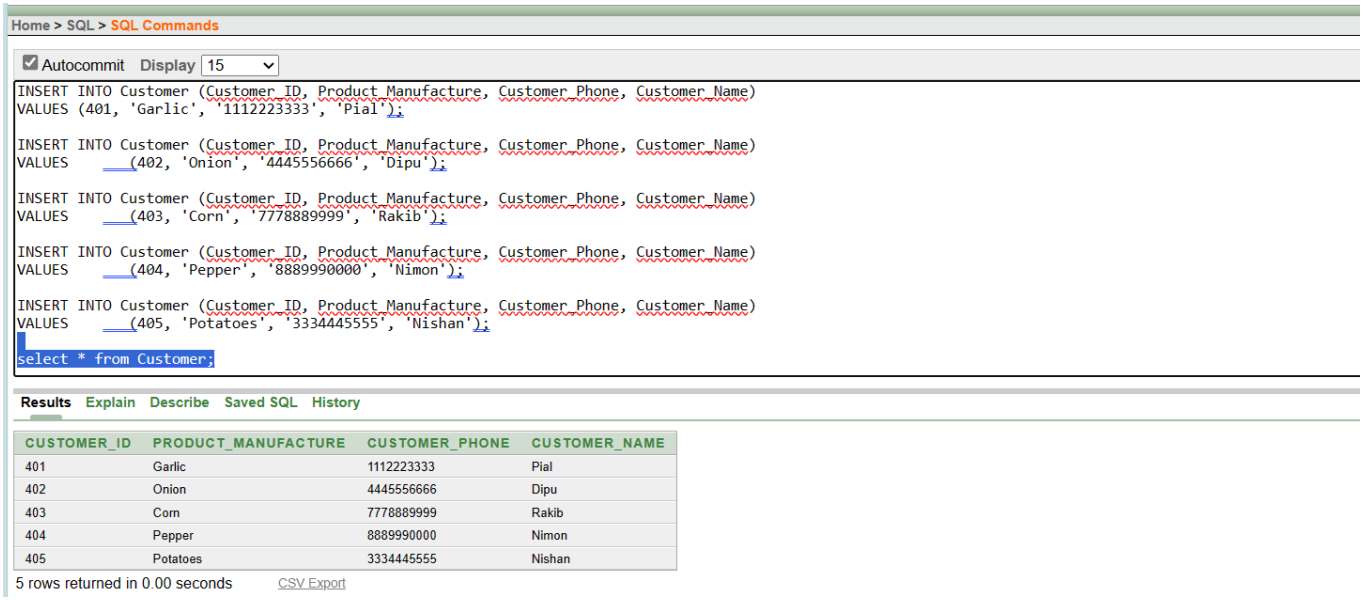
INSERT INTO Customer (Customer_ID, Product_Manufacture, Customer_Phone, Customer_Name)
VALUES (402, 'Onion', '4445556666', 'Dipu');

INSERT INTO Customer (Customer_ID, Product_Manufacture, Customer_Phone, Customer_Name)
VALUES (403, 'Corn', '7778889999', 'Rakib');

INSERT INTO Customer (Customer_ID, Product_Manufacture, Customer_Phone, Customer_Name)
VALUES (404, 'Pepper', '8889990000', 'Nimon');

INSERT INTO Customer (Customer_ID, Product_Manufacture, Customer_Phone, Customer_Name)
VALUES (405, 'Potatoes', '3334445555', 'Nishan');

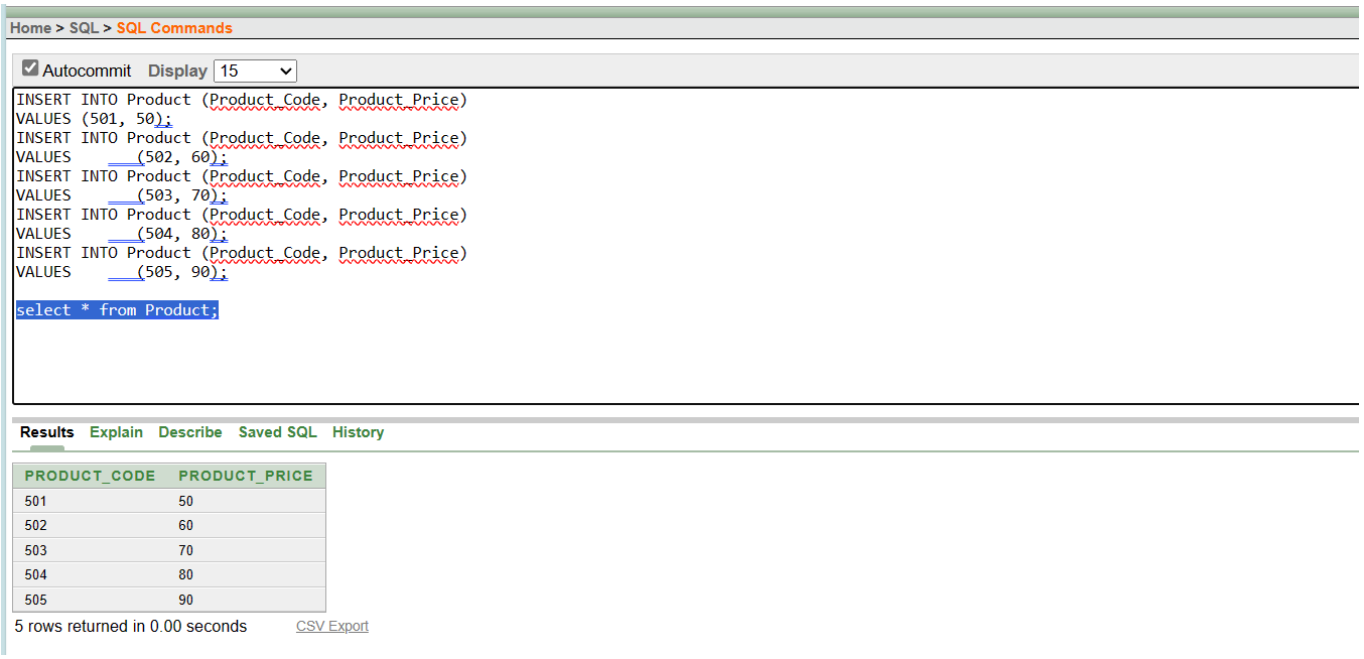
select * from Customer;



10. Information Product

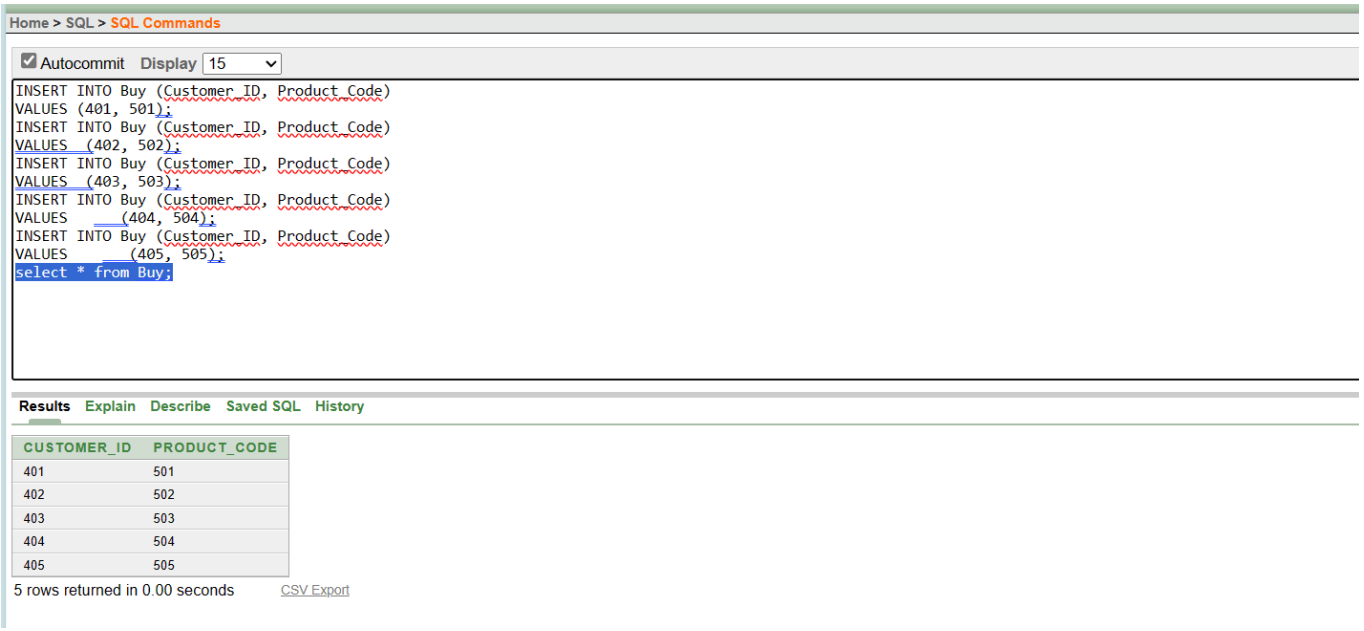
```
INSERT INTO Product (Product_Code, Product_Price)
 VALUES (501, 50);
 INSERT INTO Product (Product_Code, Product_Price)
 VALUES      (502, 60);
 INSERT INTO Product (Product_Code, Product_Price)
 VALUES      (503, 70);
 INSERT INTO Product (Product_Code, Product_Price)
 VALUES      (504, 80);
 INSERT INTO Product (Product_Code, Product_Price)
 VALUES      (505, 90);

  select * from Product;
```



11. Information Buy

```
INSERT INTO Buy (Customer_ID, Product_Code)
VALUES (401, 501);
INSERT INTO Buy (Customer_ID, Product_Code)
VALUES  (402, 502);
 INSERT INTO Buy (Customer_ID, Product_Code)
VALUES  (403, 503);
INSERT INTO Buy (Customer_ID, Product_Code)
VALUES      (404, 504);
INSERT INTO Buy (Customer_ID, Product_Code)
VALUES      (405, 505);
select * from Buy;
```



12. Information Sells

```
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES (501, 50, 100190);
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES    (502, 60, 100190);
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES (503, 70, 100190);
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES      (504, 80, 100190);
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES      (505, 90, 100190);

select * from Sells;
```

Home > SQL > SQL Commands

Autocommit Display 15 Save

```
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES (501, 50, 100190);
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES    (502, 60, 100190);
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES (503, 70, 100190);
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES      (504, 80, 100190);
INSERT INTO Sells (Product_Code, Product_Price, Registration_Number)
VALUES      (505, 90, 100190);

select * from Sells;
```

Results Explain Describe Saved SQL History

PRODUCT_CODE	PRODUCT_PRICE	REGISTRATION_NUMBER
501	50	100190
502	60	100190
503	70	100190
504	80	100190
505	90	100190

5 rows returned in 0.00 seconds CSV Export

Query Test

1. Simple Query

- **Show id, name, phone and registration number from Owns table.**

Home > SQL > SQL Commands

Autocommit Display 15 Save

```
Select Owner_ID,Owner_Name,Owner_Phone,Registration_Number
from Owns;
```

Results Explain Describe Saved SQL History

OWNER_ID	OWNER_NAME	OWNER_PHONE	REGISTRATION_NUMBER
9001	DIPU ROY	01316-008929	100190
9002	EMON BISWAS	01615-229455	100191
9003	NUSRAT JAHAN RAINA	01893-314782	100192
9004	MD. REDWANUL HAQUE	01308-767726	100193
9005	DIP KHASTAGIR	01575-078902	100194

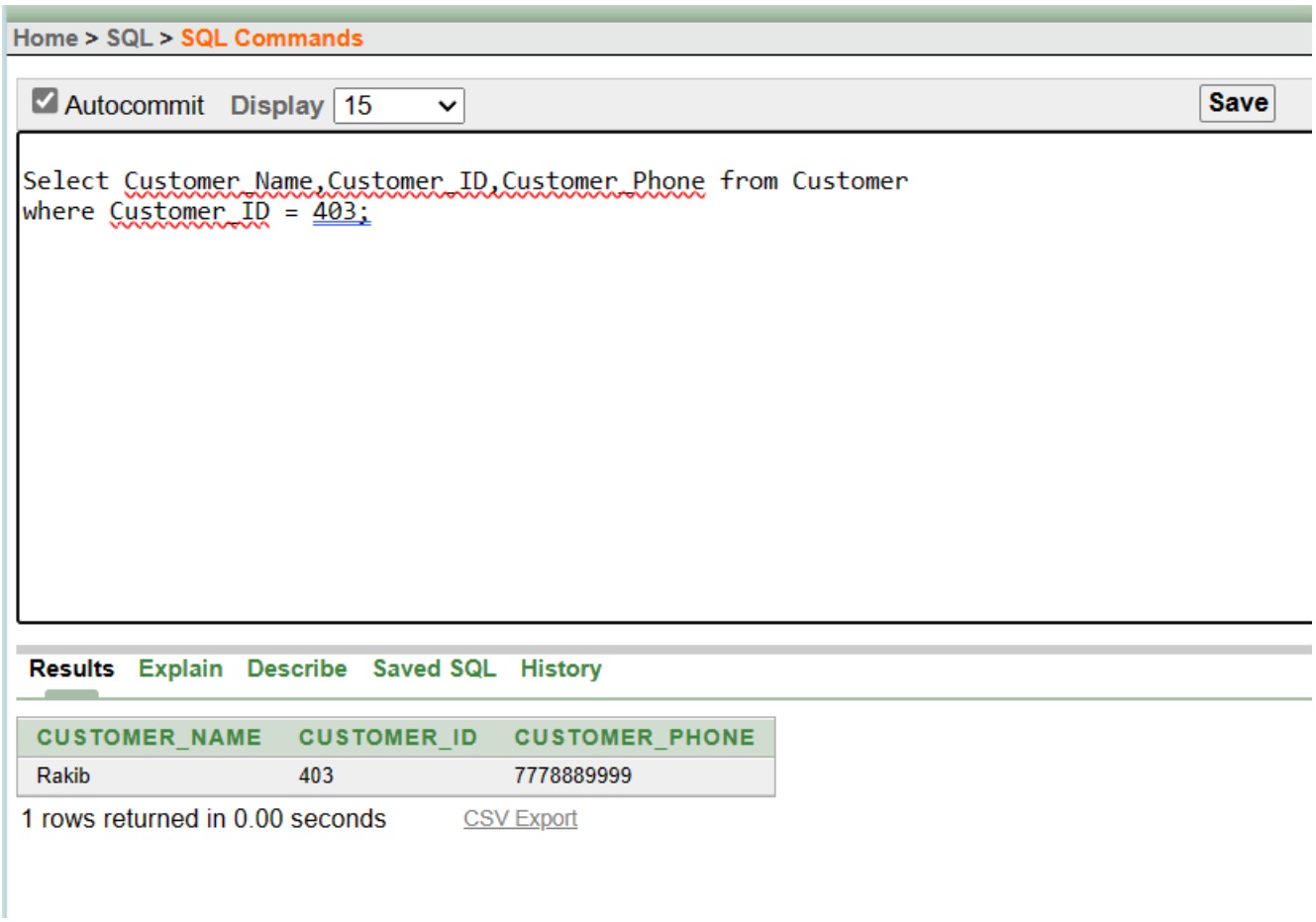
5 rows returned in 0.00 seconds CSV Export

2. Single Row Function

- **Show name, id, phone number in the Customer table where id is 403.**

Ans:

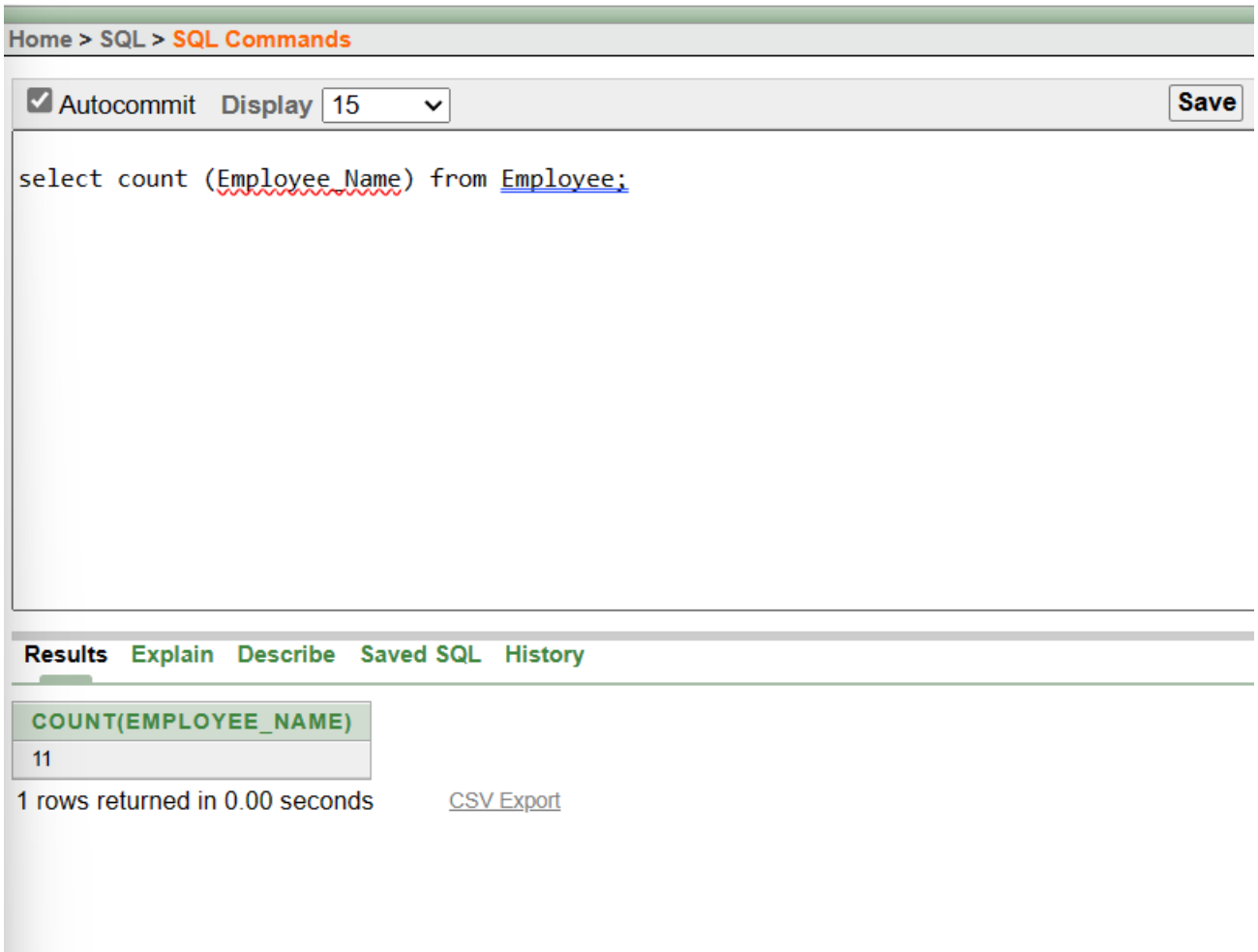
Select Customer_Name,Customer_ID,Customer_Phone from Customer where Customer_ID = 403;



3. Multiple Row Function

- **Count the number of names from the Employee table.**

Ans: select count (Employee_Name) from Employee;

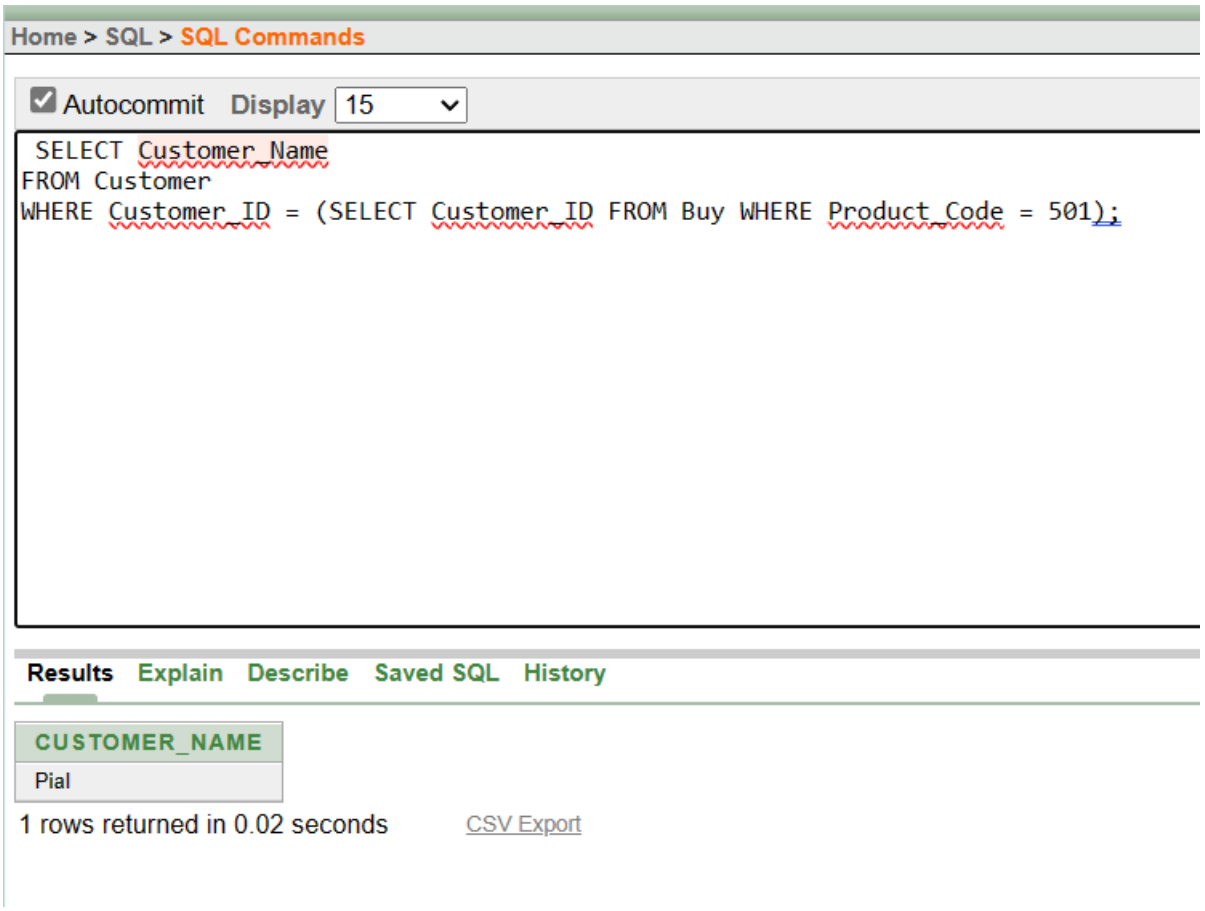


4. Single Row Subquery

- **Show the name of the customer who purchased the product from a product table where the product code is 501**

Ans:

```
SELECT Customer_Name
FROM Customer
WHERE Customer_ID = (SELECT Customer_ID FROM Buy WHERE
 Product_code = 501;
```

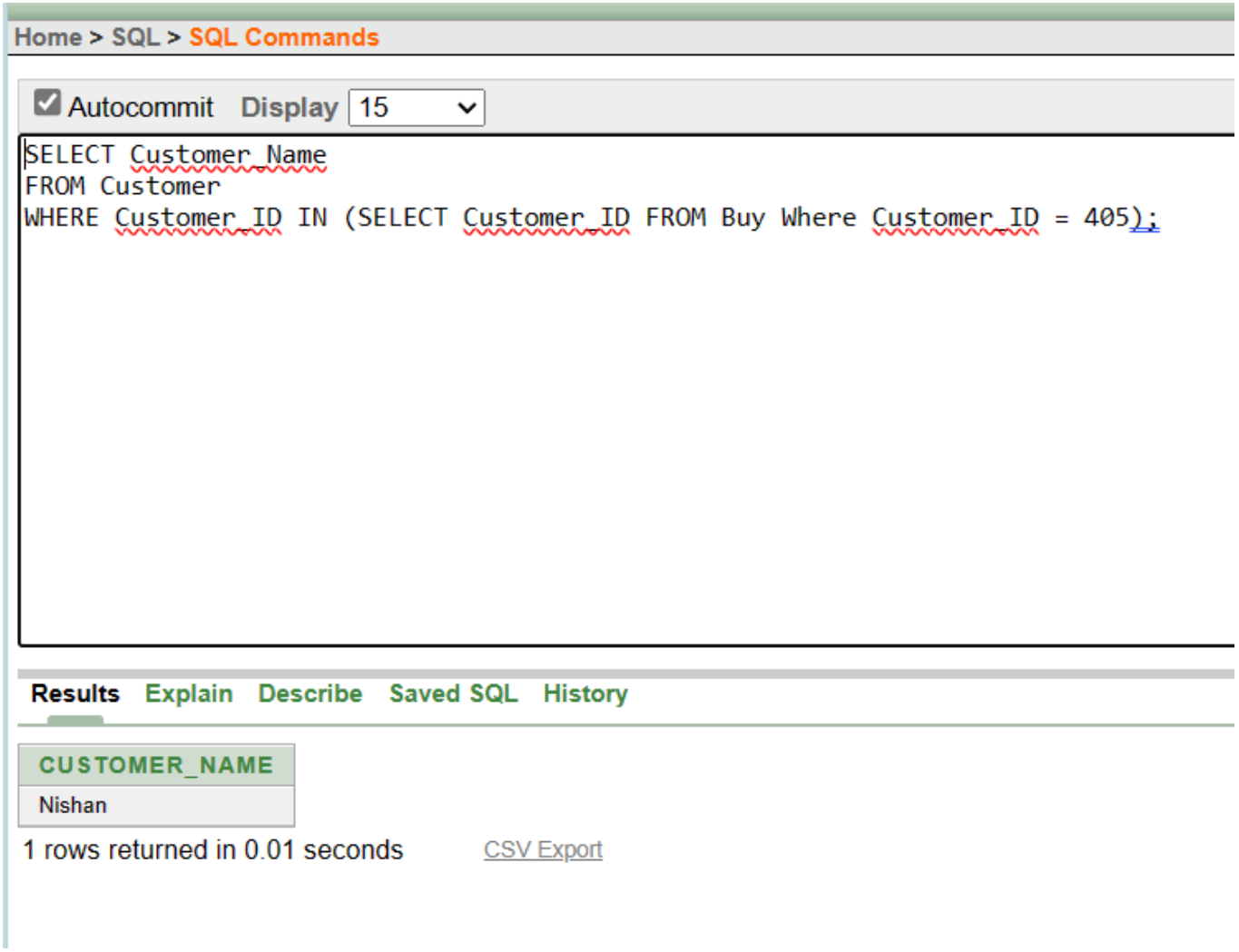


5. Multiple Row Subquery

- **Show the customers that have made purchases and share the same customer ID as customer 405**

Ans:

```
SELECT Customer_Name
FROM Customer
WHERE Customer_ID IN (SELECT Customer_ID FROM Buy Where Customer_ID = 405);
```

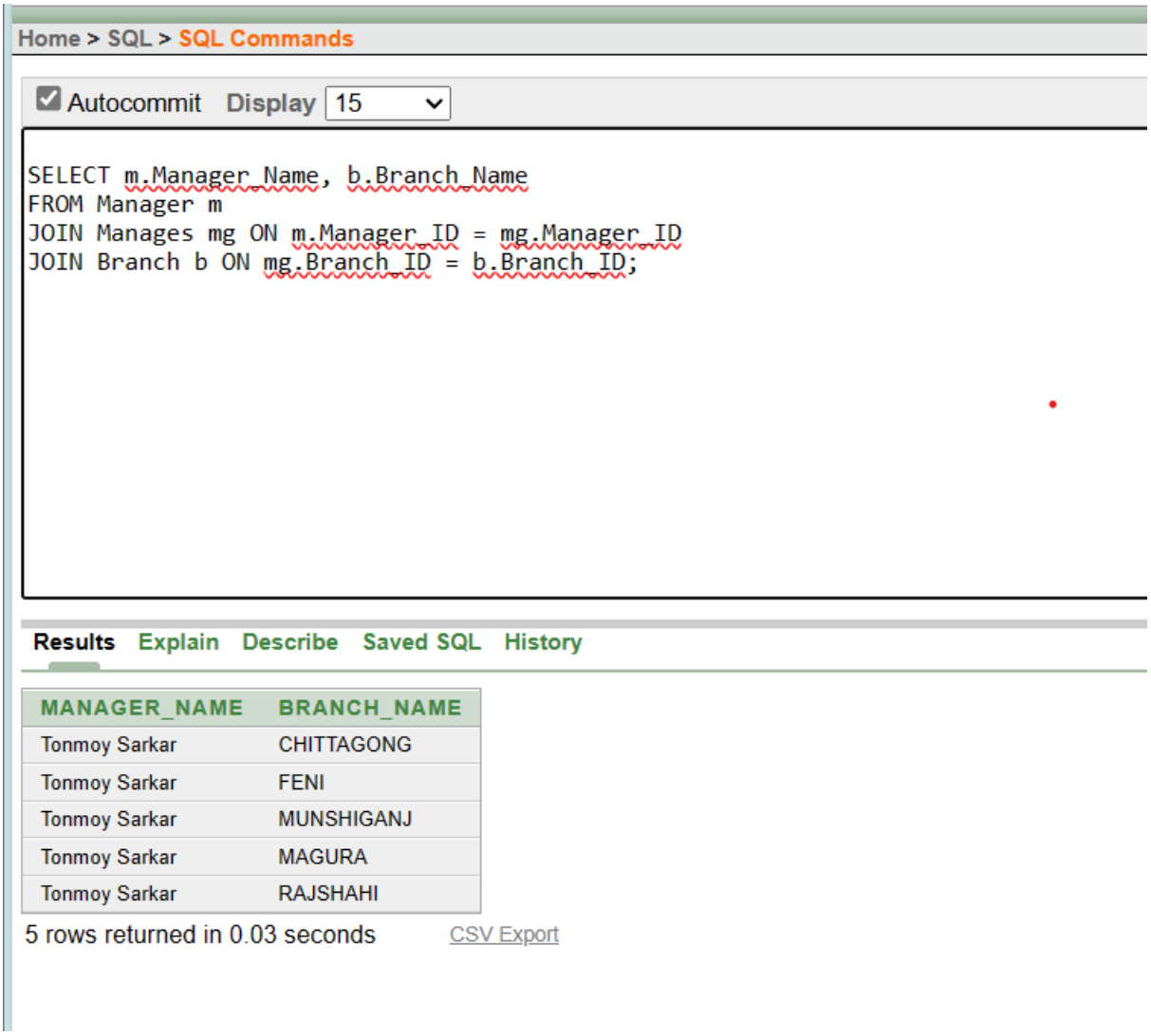


6. Self-Join

- **Who are the managers and what are the names of the branches they are responsible for managing (Show the Self-Joining)?**

Ans:

```
SELECT b.Branch_Name, m.Manager_Name
FROM Branch b
LEFT JOIN Manages mg ON b.Branch_ID = mg.Branch_ID
LEFT JOIN Manager m ON mg.Manager_ID = m.Manager_ID;
```



7. Outer Join

- **Which branch, along with its manager's name and phone number, is listed in the database, considering that some branches might not have an assigned manager?**

Ans:

```
SELECT a.BRANCH_NAME , b.EMPLOYEE_NAME , b.EMPLOYEE_PHONE
FROM Branch a , Manages b
where a.BRANCH_ID = b.MANAGER_ID(+)
```

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT a.BRANCH_NAME , b.EMPLOYEE_NAME , b.EMPLOYEE_PHONE FROM Branch a, Manages b
where a.BRANCH_ID = b.MANAGER_ID(+)
```

Results Explain Describe Saved SQL History

BRANCH_NAME	EMPLOYEE_NAME	EMPLOYEE_PHONE
RAJSHAHI	Nirjhor Rahman	01882-873111
MAGURA	Michael Johnson	01789-345909
MUNSHIGANJ	Rachel White	01234-456984
CHITTAGONG	-	-
FENI	-	-

5 rows returned in 0.06 seconds CSV Export

8. Simple view

- **Create a simple view and show the branch name, employee ID, and employee name for the employees managing the branch with ID 102?**

Ans:

create VIEW sview as select BRANCH_NAME , EMPLOYEE_ID , EMPLOYEE_NAME FROM MANAGES WHERE BRANCH_ID = 102;

select * from Sview;

Home > SQL > SQL Commands

Autocommit Display 10 Save Run

```
create VIEW  sview as select BRANCH_NAME , EMPLOYEE_ID , EMPLOYEE_NAME
FROM MANAGES WHERE BRANCH_ID = 102;

select * from sview
```

Results Explain Describe Saved SQL History

BRANCH_NAME	EMPLOYEE_ID	EMPLOYEE_NAME
RAJSHAHI	3010008	Anamul Haque

1 rows returned in 0.00 seconds CSV Export

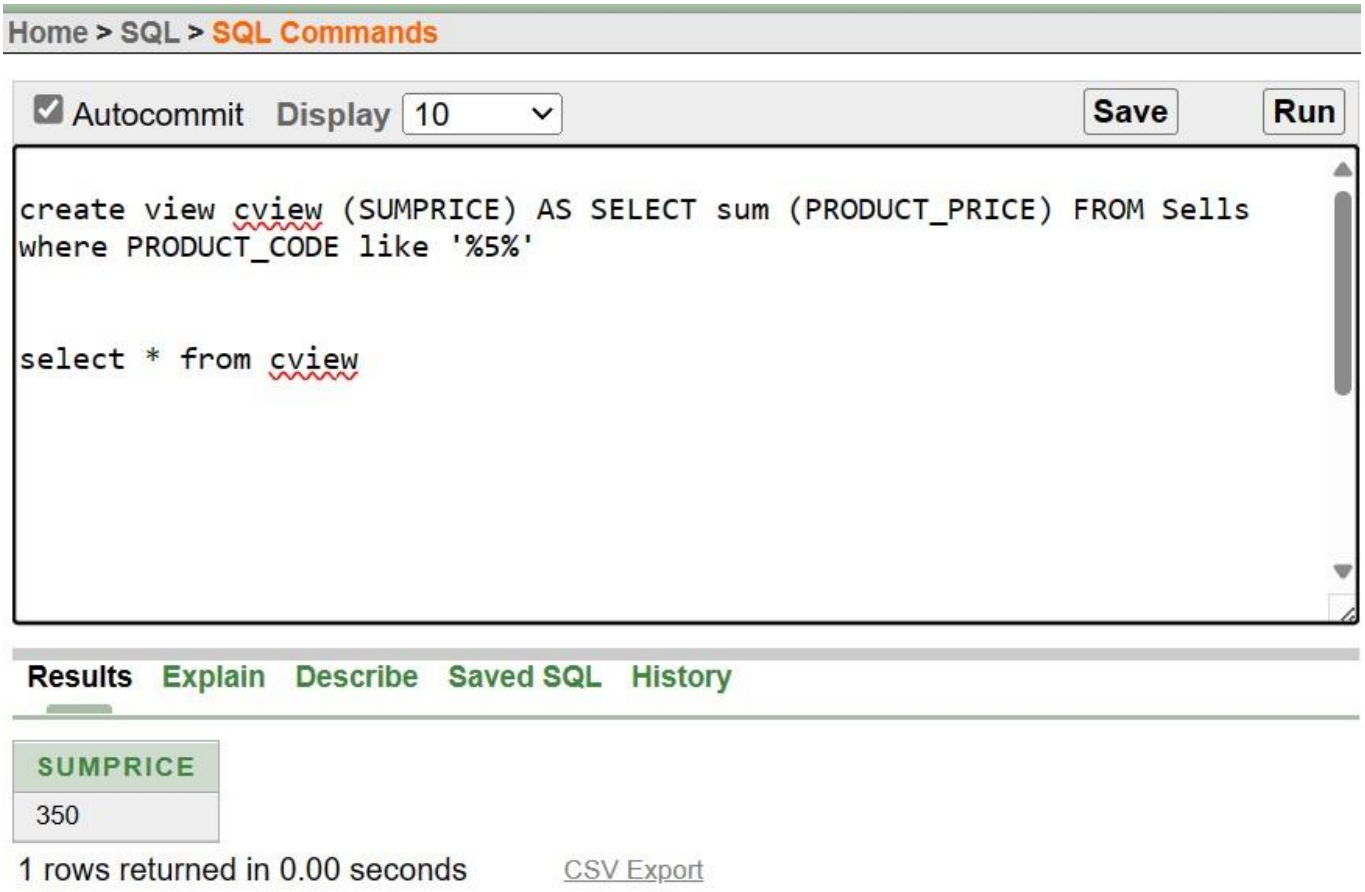
9. Complex View

- **Create the complex view show the total sum of product prices for items whose product codes contain the digit 5 across all sales?**

Ans:

create view cview (SUMPRICE) AS SELECT sum (PRODUCT_PRICE) FROM Sells where PRODUCT_CODE like '%5%'

select * from cview;



Conclusion:

The Agricultural Management System is a transformative tool designed to optimize farming operations through advanced technology integration. By leveraging features such as crop monitoring, inventory management, and market analysis, the system empowers farmers to make data-driven decisions, maximize productivity, and mitigate risks. Its real-time insights into crop health, soil conditions, and market trends enable proactive adjustments to cultivation strategies, ultimately contributing to food security, economic growth, and environmental sustainability. As technology continues to evolve, further enhancements to the system hold the promise of revolutionizing agriculture, driving innovation, and ensuring the resilience of farming practices in the face of emerging challenges.