**PRACTICAL : 1**

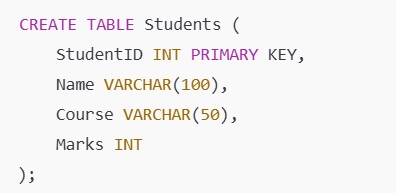
**AIM : Explore core concepts of DBMS and SQL, and perform operations using various SQL command categories such as DDL, DML, DCL, and TCL.**

**TASK 0 : Understand the Scenario: College Student Record System**

**Task 0**

# 1. DDL – Data Definition Language

**Task: Create a table to store student data**



**Task: Add a new column for Email**

ALTER TABLE Students ADD Email VARCHAR(100);

**Task: Modify Marks to support decimal values**

ALTER TABLE Students MODIFY Marks FLOAT;

**Task: Rename the table**

RENAME TABLE Students TO StudentRecords;

**Task:** Removes **all rows** from a table student quickly, but **keeps the table structure**.

TRUNCATE TABLE Students;

**Task: Delete table student from the database permanently.**

DROP TABLE Students;

**2. DML – Data Manipulation Language and TCL - Transaction**

# Control Language Command

**Task: Add, update, delete, and retrieve data**

**Task:** Insert records into student record table

INSERT INTO StudentRecords (StudentID, Name, Course, Marks, Email) VALUES (1, 'Akshita Patel', 'BSc IT', 85.5, 'akshita@gmail.com');

**Task:** Update a record into student record to marks 90 where studentID 1.

UPDATE StudentRecords

SET Marks = 90

WHERE StudentID = 1;

**Task:**  Delete a record from the table studentrecord where student ID is 1.

DELETE FROM StudentRecords

WHERE StudentID = 1;

**Task:** Display the record of studentrecord Table.

SELECT \* FROM StudentRecords;

**Task:** Insert records into student record table

INSERT INTO StudentRecords (StudentID, Name, Course, Marks)

VALUES (101, 'Akshita Patel', 'BSc IT', 78);

**Task: Set a savepoint after insert**

SAVEPOINT sp\_insert;

**Task:** Update a record into student record to marks 90 where studentID 101.

UPDATE StudentRecords

SET Marks = 95

WHERE StudentID = 101;

**Task: Rollback to savepoint**

ROLLBACK TO sp\_insert;

**Task: Commit the correct data**

COMMIT;

**Task: Display the record**

SELECT \* FROM StudentRecords;

# 3. DCL – Data Control Language

**Task: Create users and manage permissions**

-- Create a user

CREATE USER 'librarian\_user'@'localhost' IDENTIFIED BY 'lib123';

-- Grant permissions

GRANT SELECT, INSERT, UPDATE ON LibraryDB.LibraryBooks TO

'librarian\_user'@'localhost';

-- Revoke permissions

REVOKE UPDATE ON LibraryDB.LibraryBooks FROM 'librarian\_user'@'localhost';

# 4. TCL (Transaction Control Language)

1. **COMMIT** 
   * **Purpose**: Saves all changes made by the current transaction permanently to the database.
   * **Syntax**:

COMMIT;

1. **ROLLBACK** 
   * **Purpose**: Reverts all changes made by the current transaction.
   * **Syntax**:

ROLLBACK;

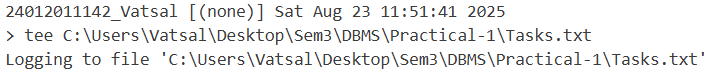
1. **SAVEPOINT** 
   * **Purpose**: Sets a point within a transaction to which you can later roll back.
   * **Syntax**:

SAVEPOINT savepoint\_name;

**TASK 1**

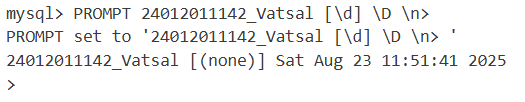
**Prerequisite Steps before Task 1**

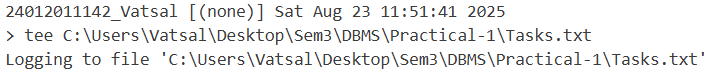
**Step 1 : Use tee PATH\TO\FOLDER\FILENAME.txt in MySQL CLI to start saving all commands and outputs of the current session into a file until stopped with notee or else session is closed.**

****

* **SYNTAX : tee PATH\TO\FOLDER\FILENAME.txt**
* tee in MySQL Command-Line Client enables logging of all commands and their results into a file during the same session. From this point onward, all entered commands and their outputs will be written to the specified file.
* This is useful for: Record keeping, Creating reports of SQL execution, Debugging query results later. The log continues until you turn it off with “notee” Command

**Step 2 : Changing the MySQL Prompt**

****

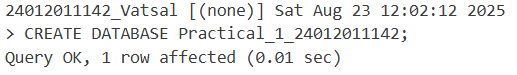
****

PROMPT changes how the MySQL CLI prompt looks.

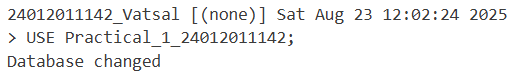
Components:

* FULLENROLLMENT\_FULLNAME → Custom label or project name
* [\d] → Shows the current database in square brackets
* \D → Shows the current date
* \n → Inserts a newline before
* > → The actual command prompt symbol

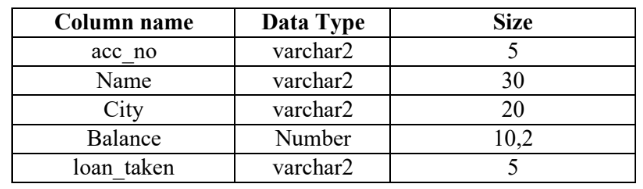
**Step 3 : Create a new Database**



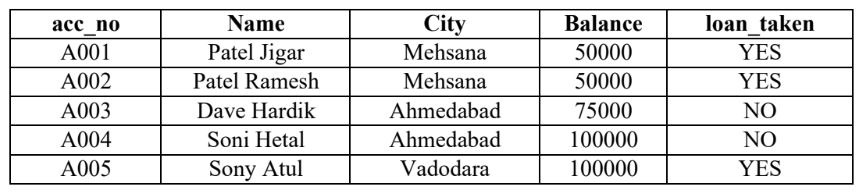
**Step 4 : Select a Database**



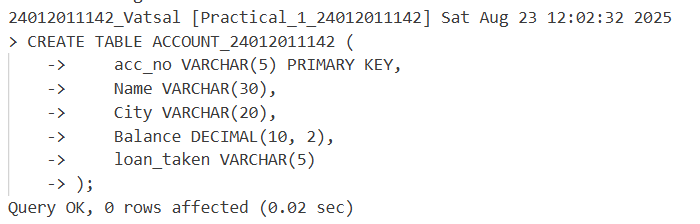
**Create a table ACCOUNT\_FULLENROLLMENTNO**

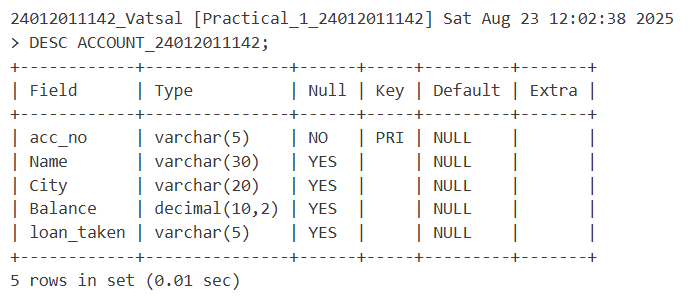
****

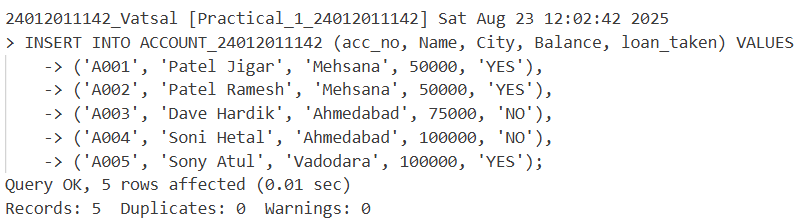
**Insert the following records in ACCOUNT\_FULLENROLLMENTNO table**

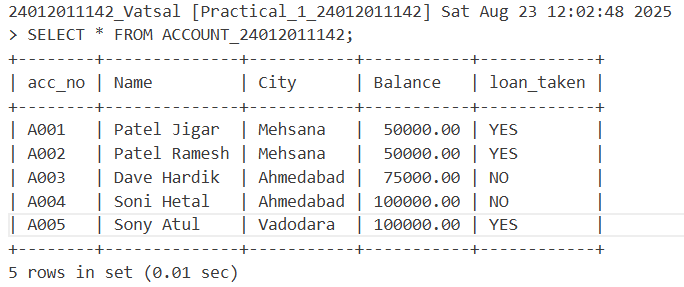
****

**Query with Output :**

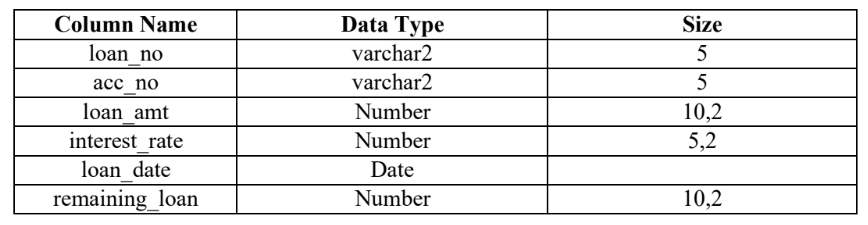
****

****

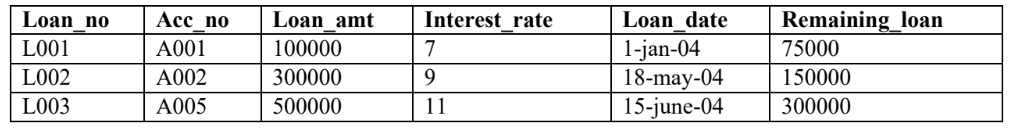
****

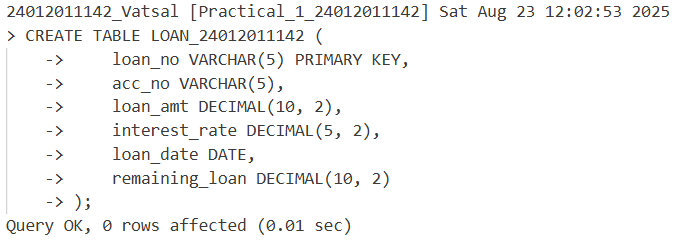
****

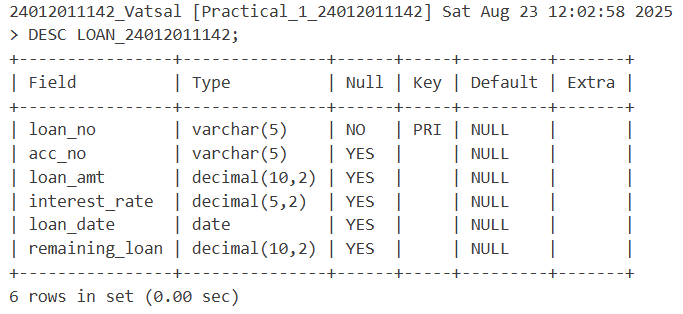
**Create a table LOAN\_FULLENROLLMENTNO**

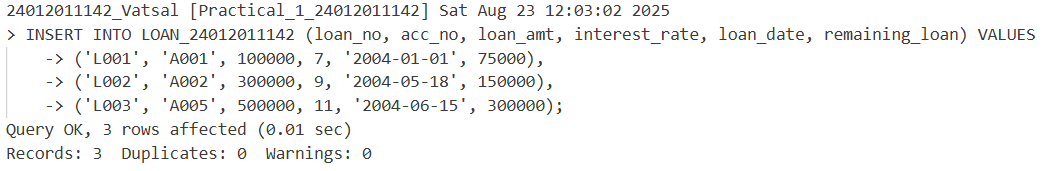
****

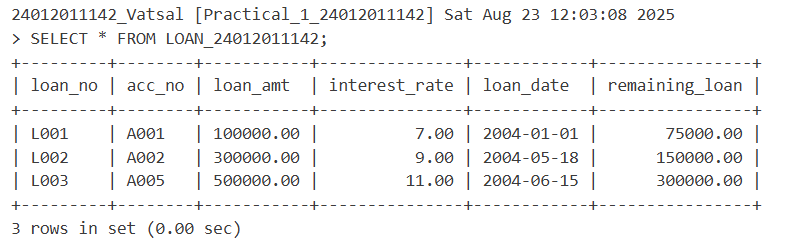
**Insert the following records in LOAN\_FULLENROLLMENTNO table**

****

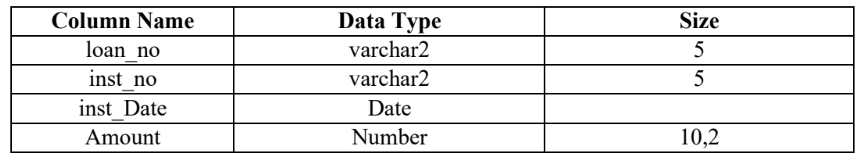
****

****

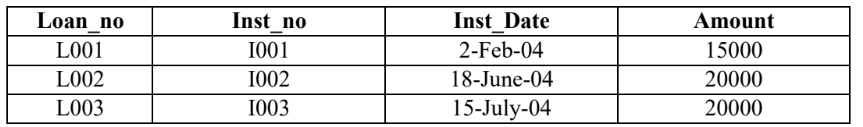
****

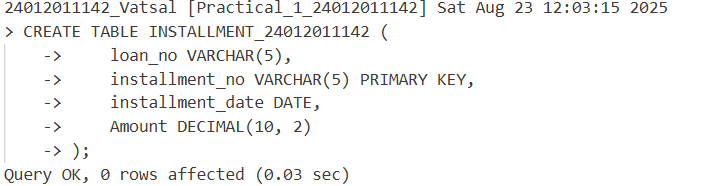
****

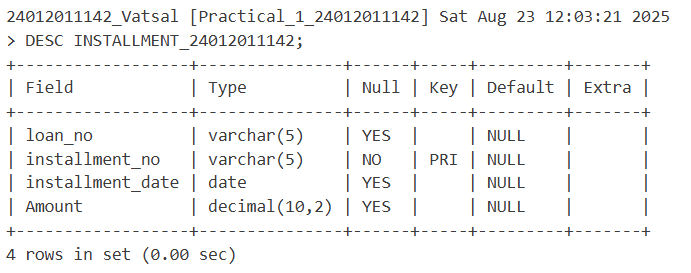
**Create a table INSTALLMENT\_FULLENROLLMENTNO**

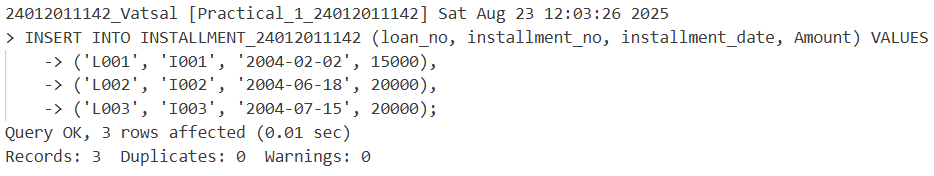
****

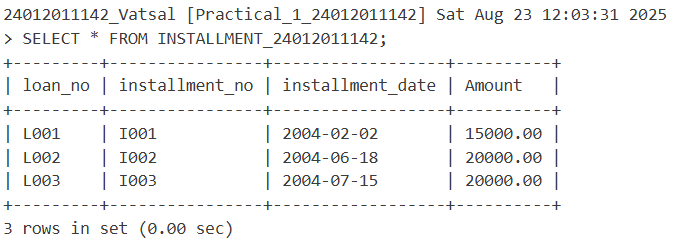
**Insert the following records in INSTALLMENT\_FULLENROLLMENTNO table**

****

****

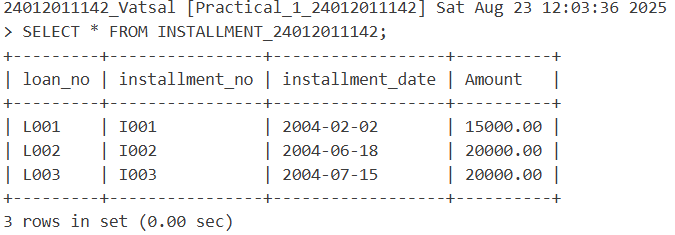
****

****

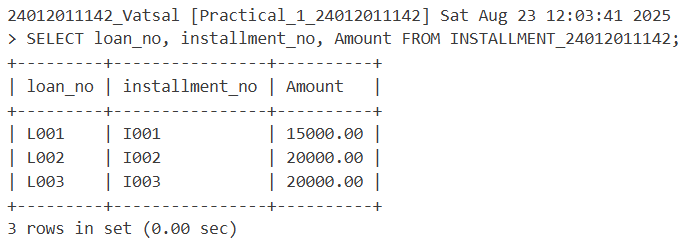
****

**Answer following Queries based on above 3 tables.**

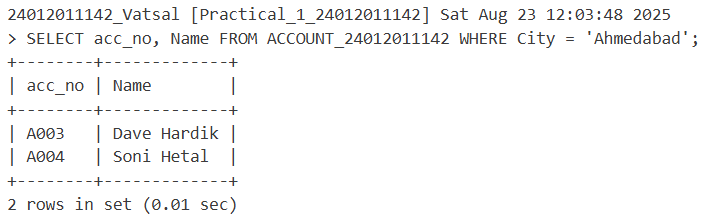
1. **Display all rows and all columns of table Installment.**

****

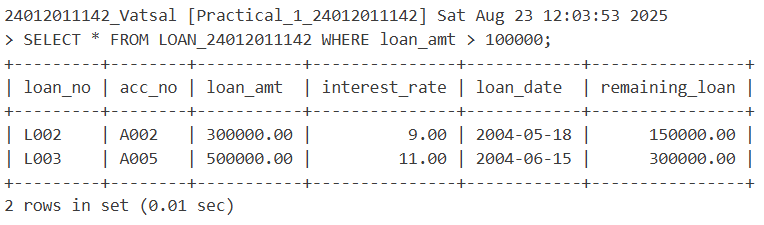
1. **Display all rows and selected columns of table Installment.**

****

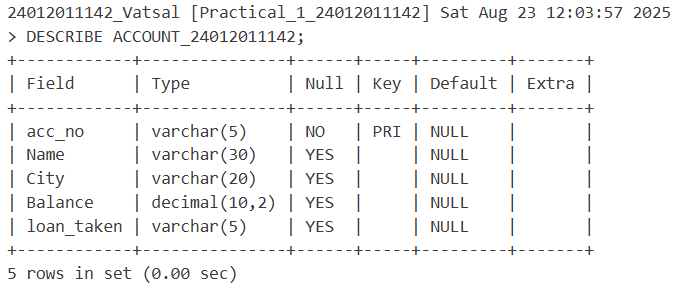
1. **Display selected rows and selected columns of table Account.**

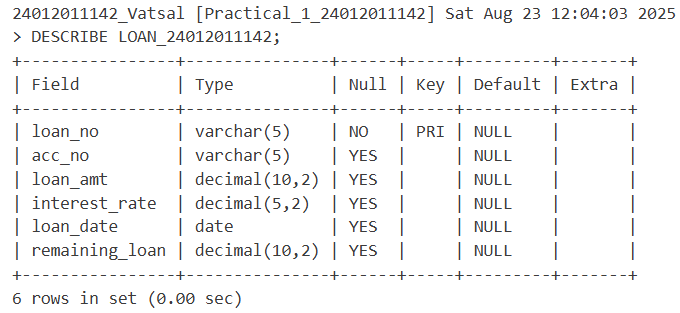
****

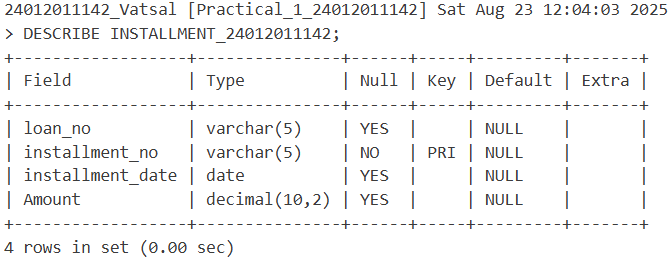
1. **Display selected rows and all columns of table loan.**

****

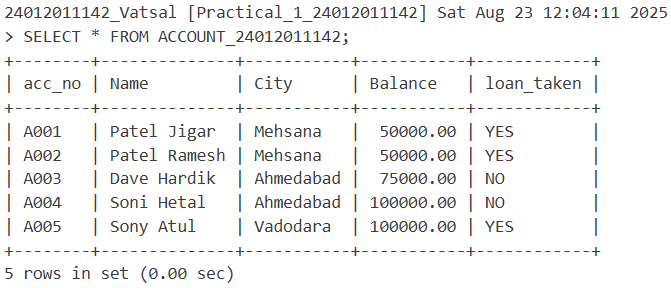
1. **Show the structure of the table loan, account and installment.**

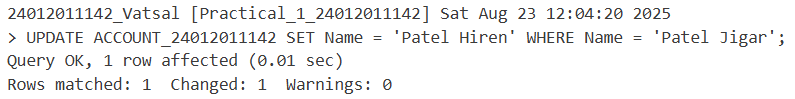
****

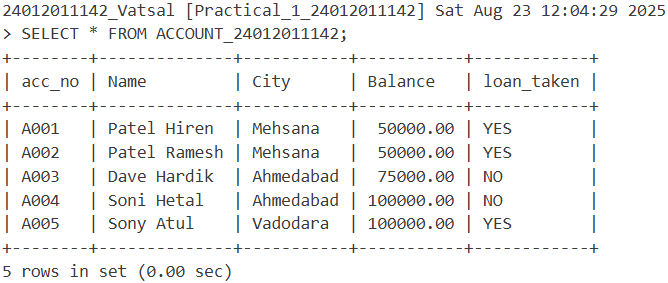
****

****

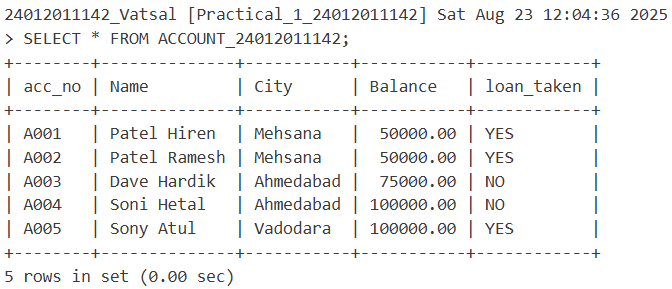
1. **Change the name ‘Patel Jigar’ to ‘Patel Hiren’ in Account Table.**

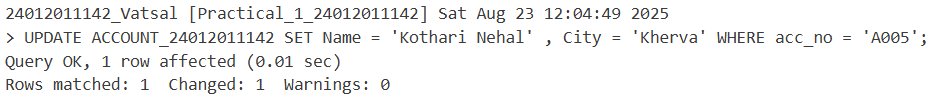
****

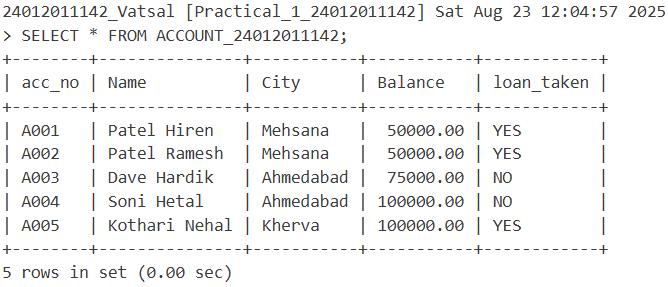
****

****

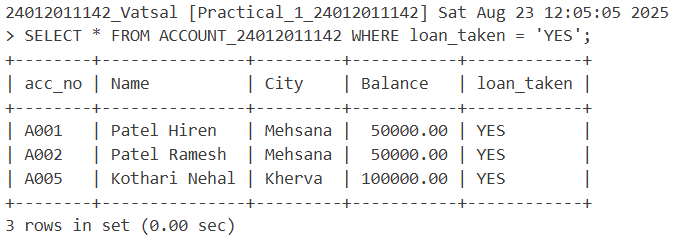
1. **Change the name and city where account number is A005. (new name = ‘Kothari Nehal’ and new city = ‘Kherva’).**

****

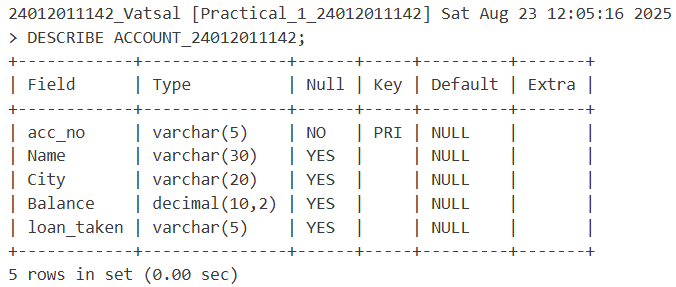
****

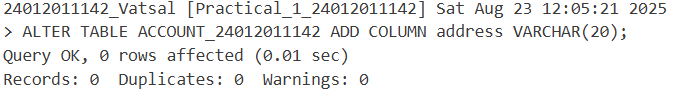
****

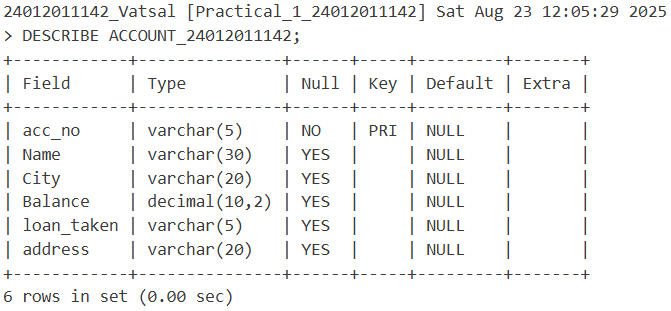
1. **Display only those records where loan taken status is ‘YES’.**

****

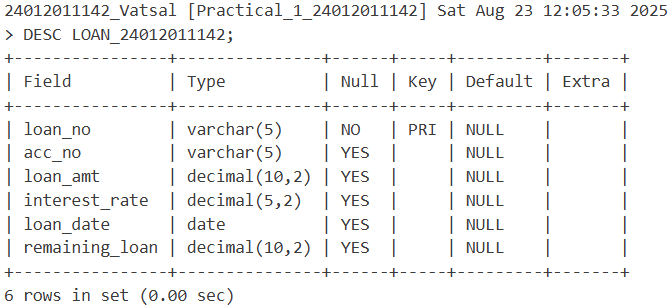
1. **Add the new column (address varchar2 (20)) into table ACCOUNT.**

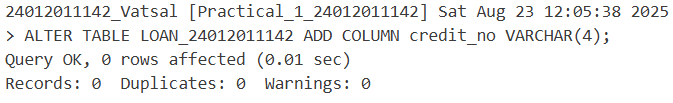
****

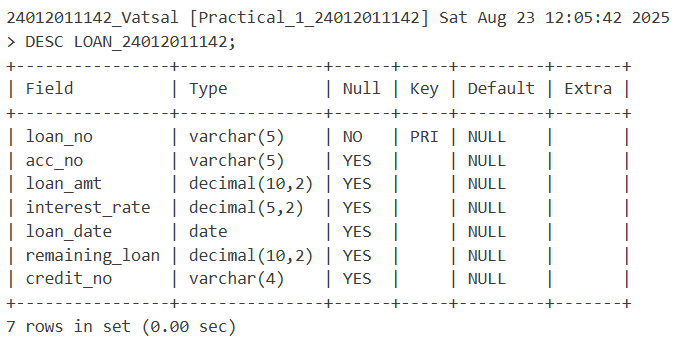
****

****

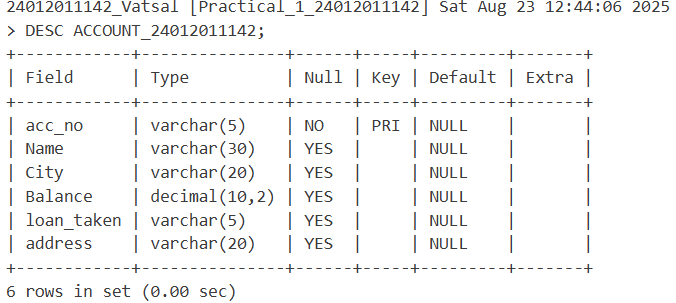
1. **Modify the structure of table LOAN by adding one column credit\_no varchar2 (4) (Loan table).**

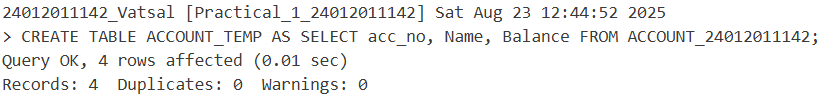
****

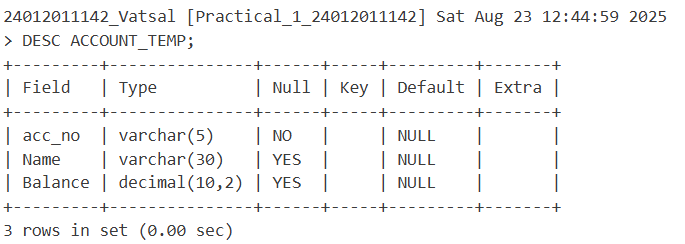
****

****

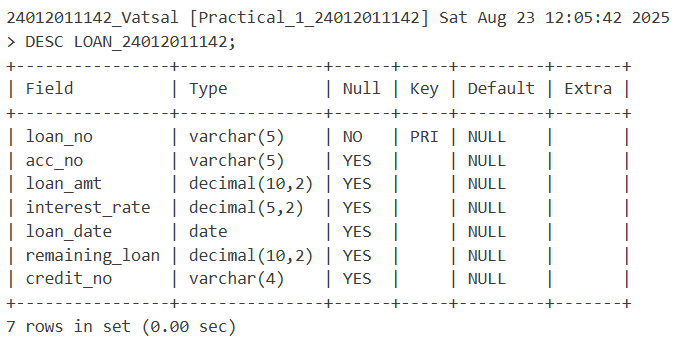
1. **Create another table ACCOUNT\_TEMP having columns (acc\_no, name, balance) from table ACCOUNT.**

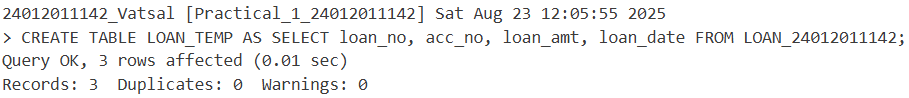
****

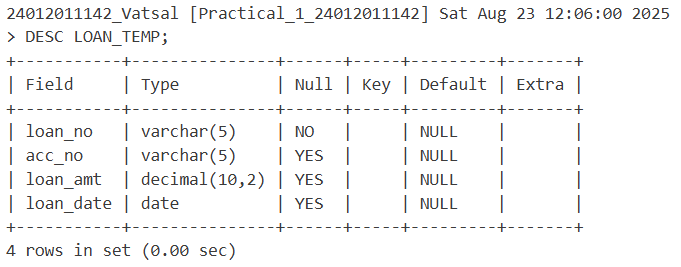
****

****

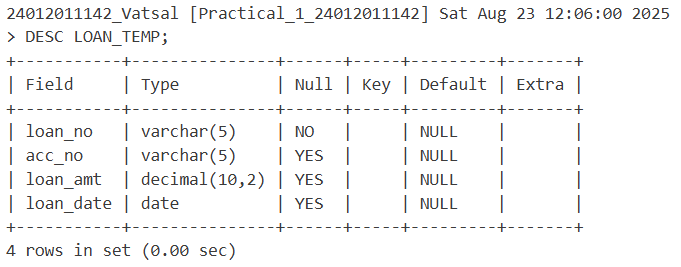
1. **Create another table LOAN\_TEMP (loan\_no, Acc\_no, loan\_amt, loan\_date) from The table LOAN.**

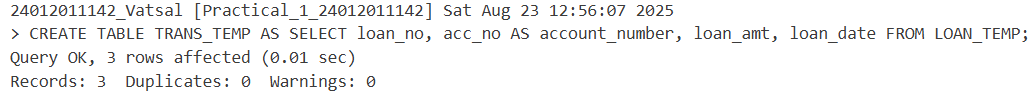
****

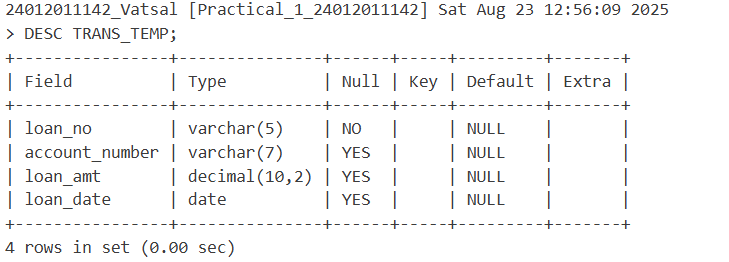
****

****

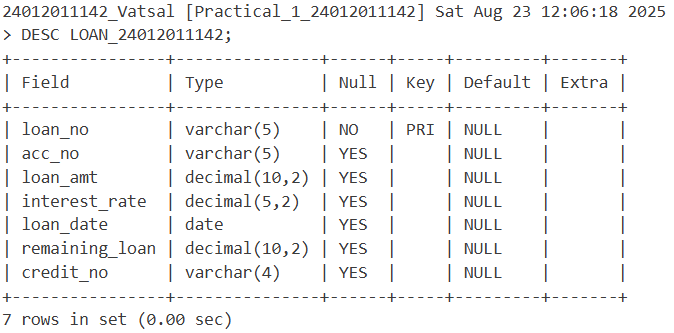
1. **Create another table TRANS\_TEMP by change the column name acc\_no to account\_no from LOAN\_TEMP.**

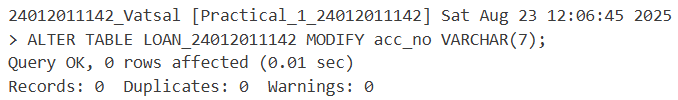
****

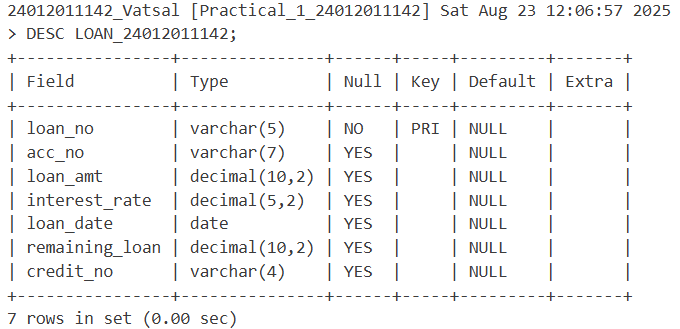
****

****

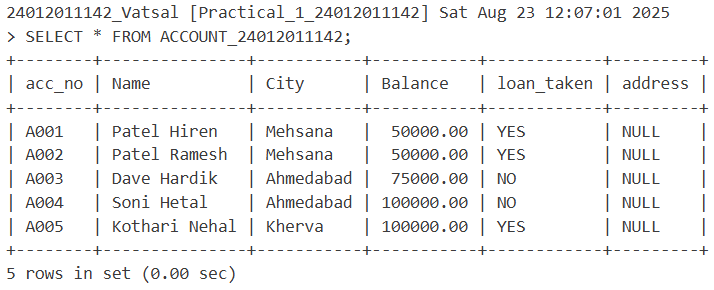
1. **Increase the size 5 to 7 of column acc\_no (Loan table).**

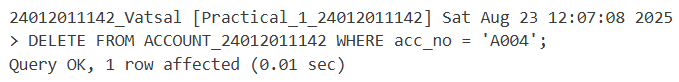
****

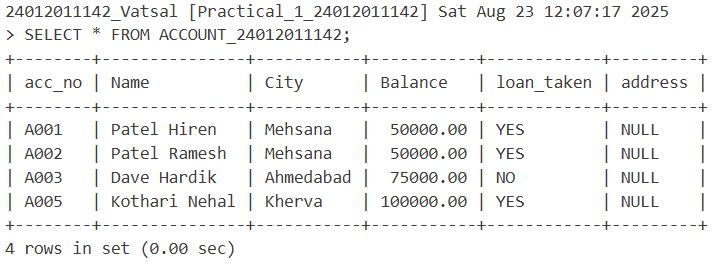
****

****

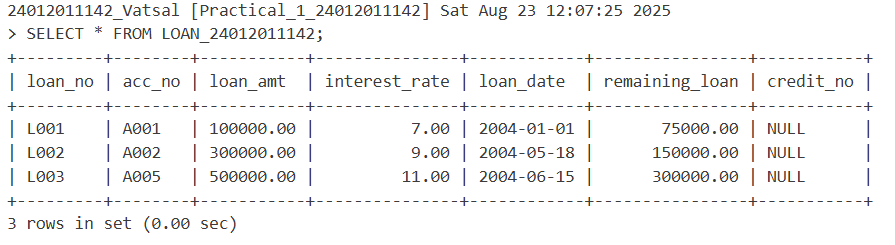
1. **Delete the records whose account no is A004.**

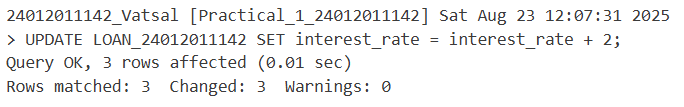
****

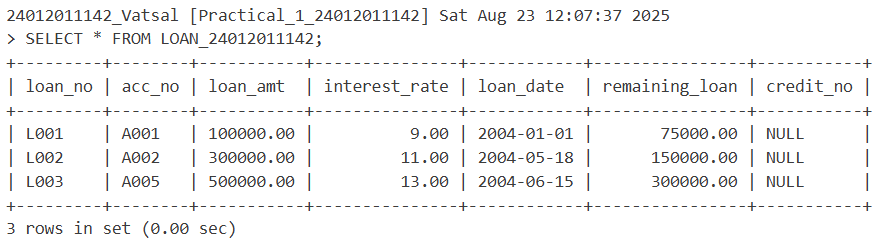
****

****

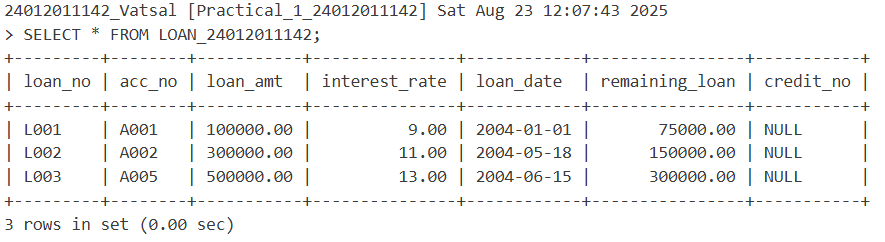
1. **For each loan holders Increase the interest rate by 2% (Loan table).**

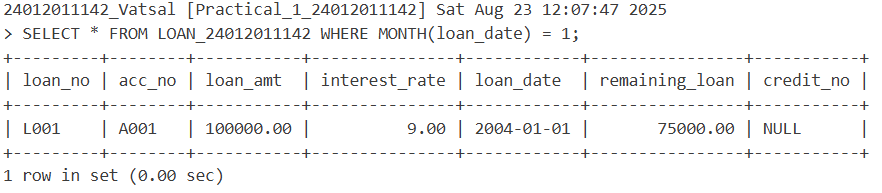
****

****

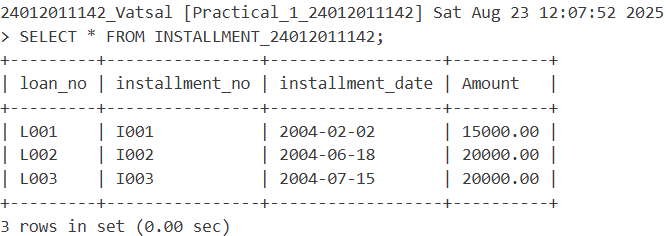
****

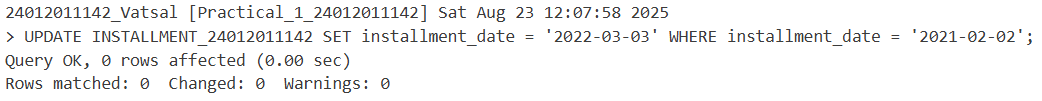
1. **Display only those records where loan holder taken a loan in month of January (Loan table).**

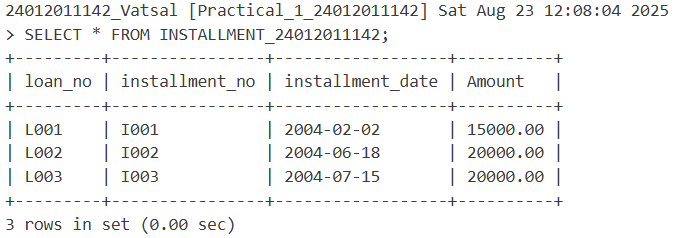
****

****

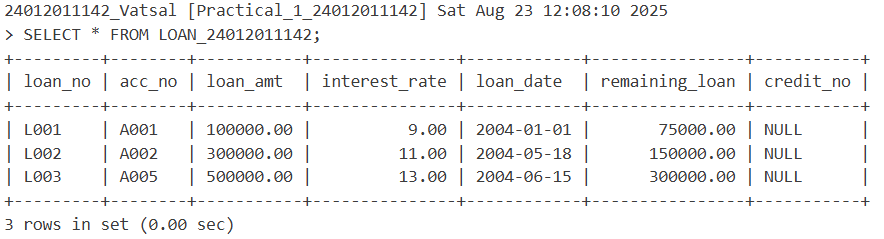
1. **Change the Inst\_Date ‘2-Feb-21’ to ’3-Mar-22’.**

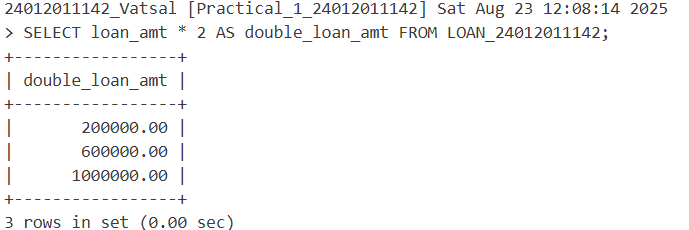
****

****

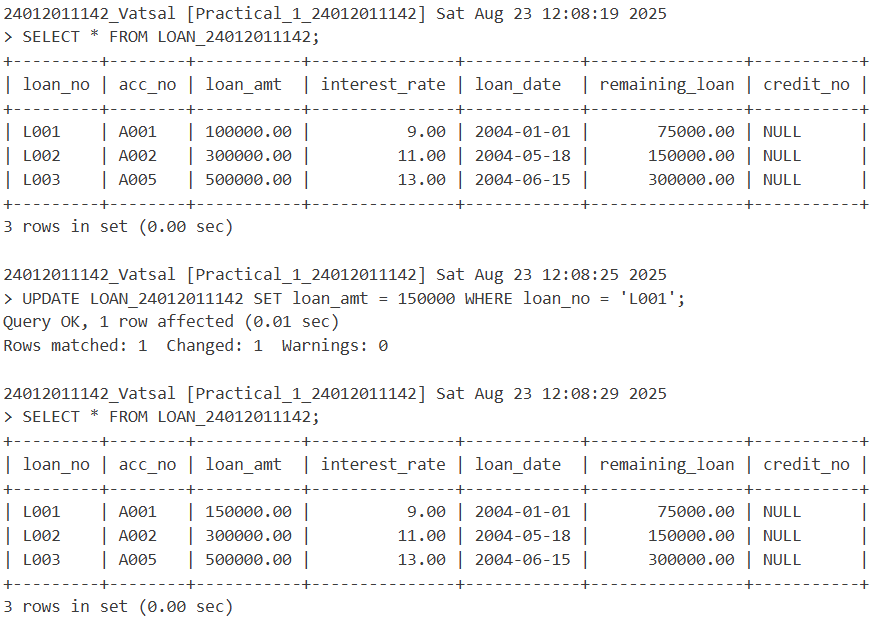
****

1. **Display the Loan amount\*2 of table LOAN.**

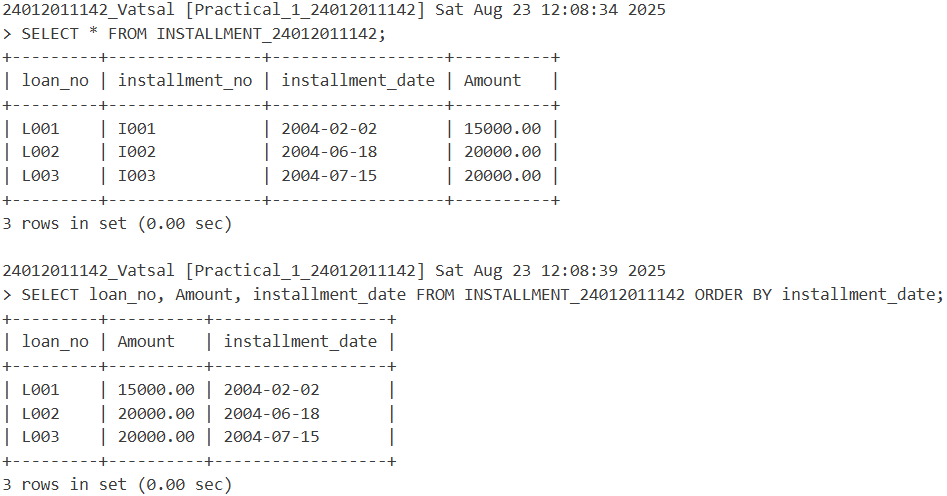
****

****

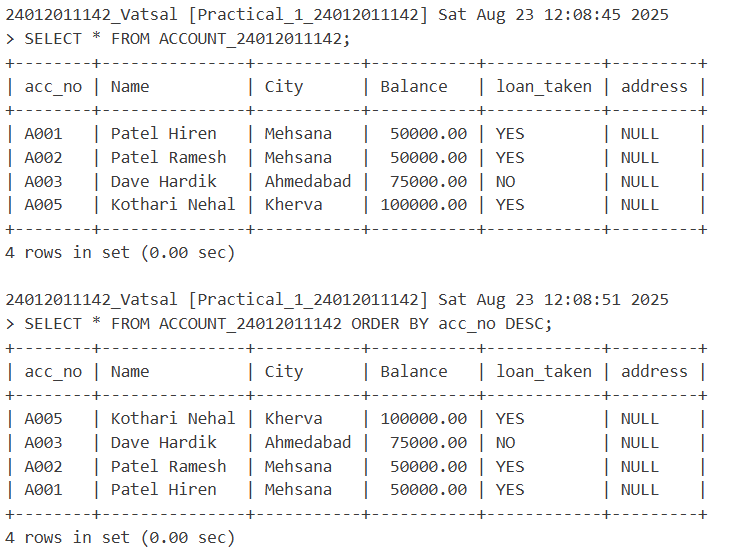
1. **Change the loan\_amt 100000 to 150000 where loan number is L001. (Loan table).**

****

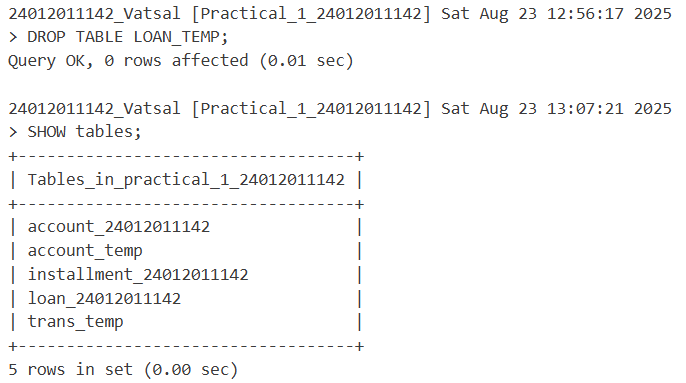
1. **Display loan\_no, amount of Installment table by date wise.**

****

1. **Select all the records of account table in descending order (account number wise).**

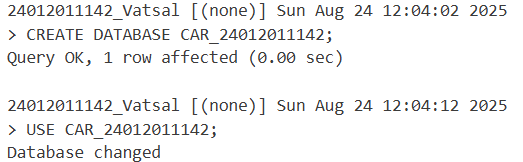
****

1. **Delete a table LOAN\_TEMP.**

****

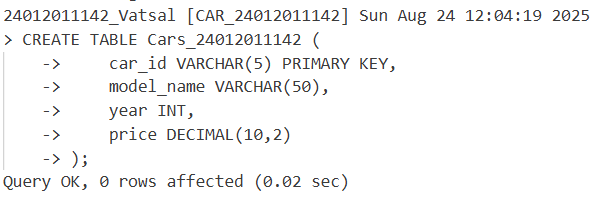
**DIY Task:**

* **Create a Relational Database for a Car Manufacturing Company. Think of at least 3 schemas of the same. Write Queries to perform CRUD Operation.**

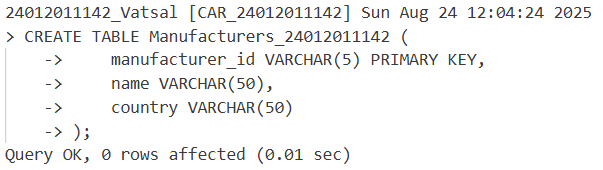


Create three related tables:

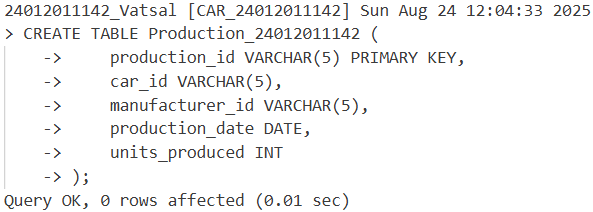
1. Cars\_yourENno (store car ID, model name, year, and price)



1. Manufacturers\_yourENno (store manufacturer ID, name, and country)



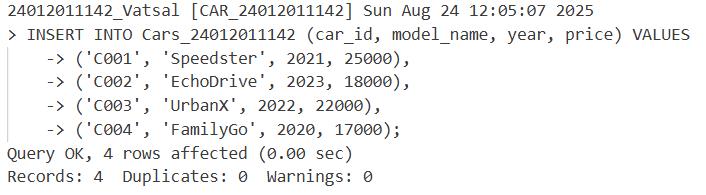
1. Production\_yourENno (store production ID, car ID, manufacturer ID, quantity, and production date)

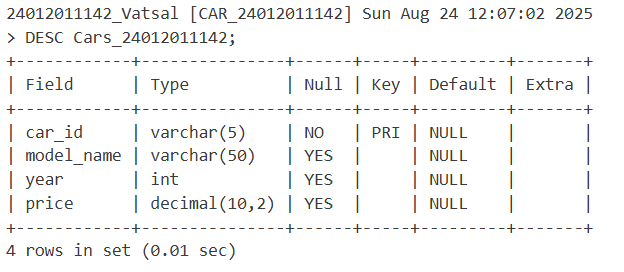


Write Queries to perform CRUD Operation.

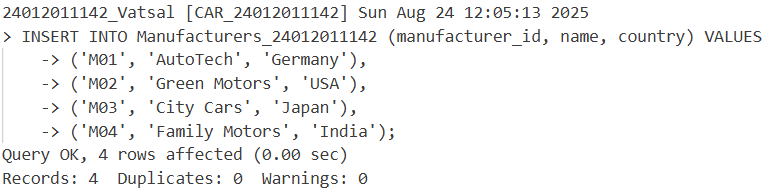
1. Insert a new car with model name 'EcoDrive', year 2023, and price

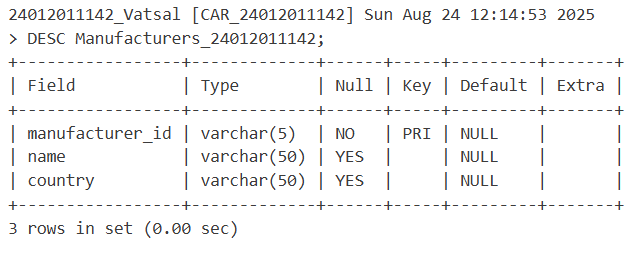
18,000 into the Cars table.



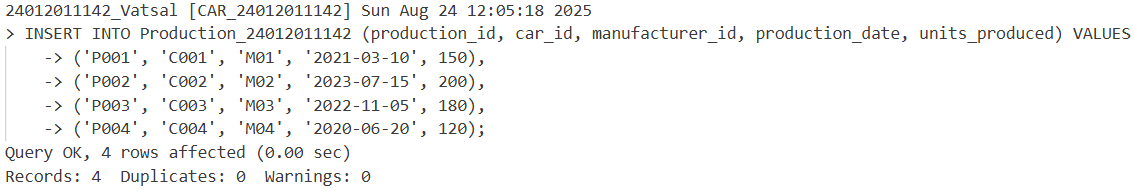


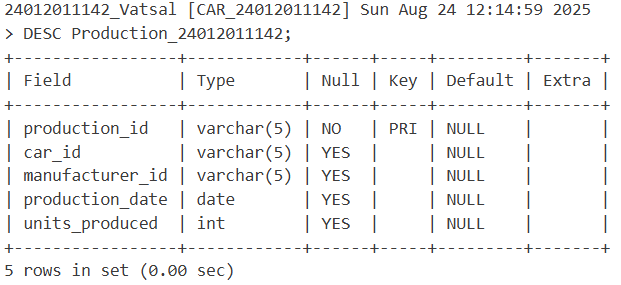
1. Insert a new manufacturer named 'GreenMotors' from 'USA' into the Manufacturers table.



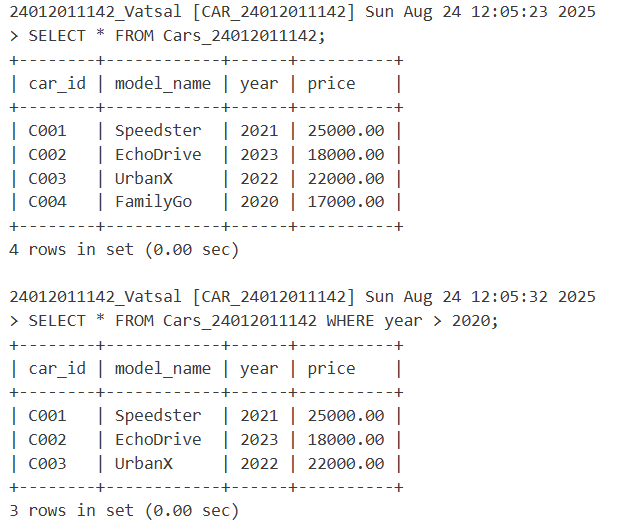


1. Insert a production record for 200 units of car ID 2 made by manufacturer ID 2 on '2023-07-15'.

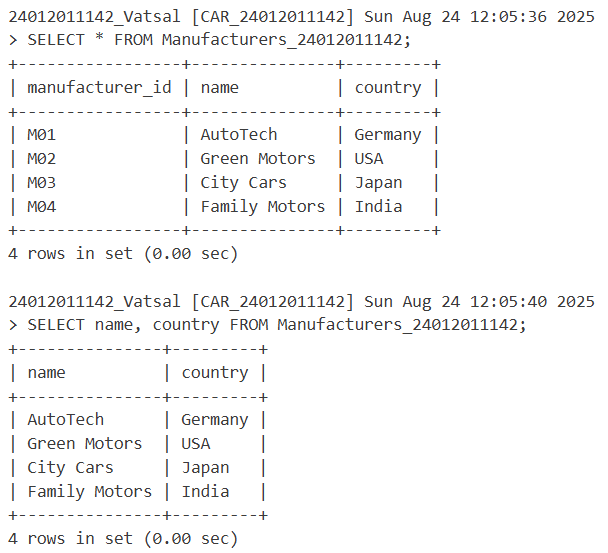




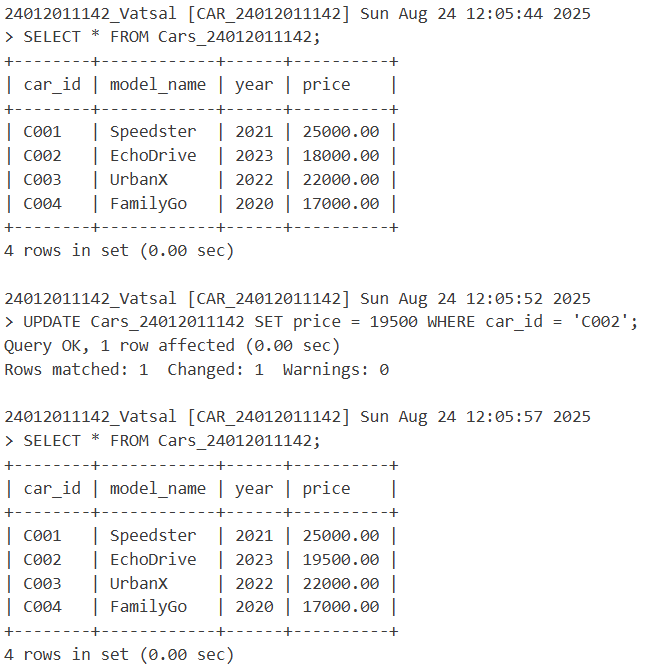
1. Select all cars that were manufactured after the year 2020.



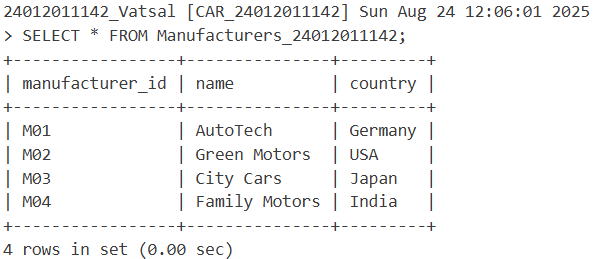
1. Select the names and countries of all manufacturers.

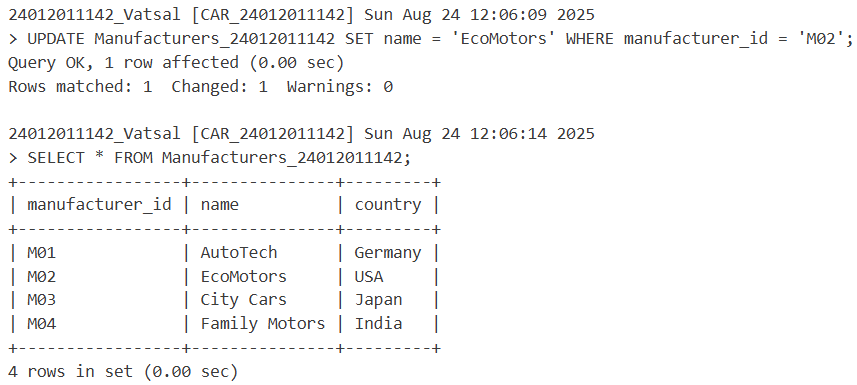


1. Update the price of the car with ID 2 to 19,500.

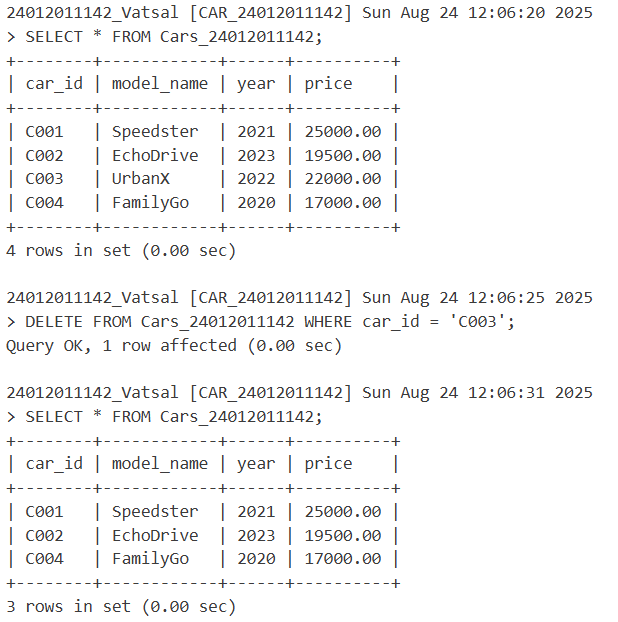


1. Update the name of the manufacturer with ID 2 to 'EcoMotors'.

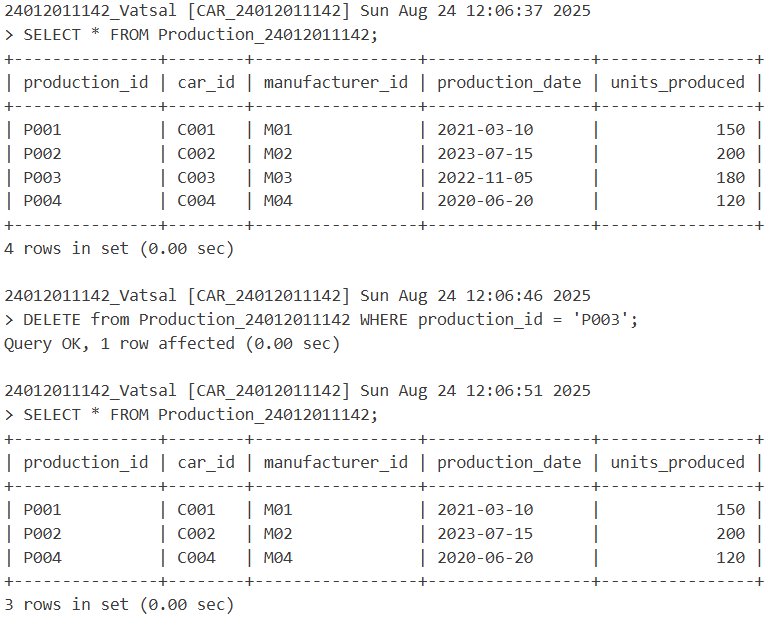




1. Delete the car with ID 3 from the Cars table.



1. Delete the production record with ID 1002 from the Production table.



1. Select all production records and show the car ID, manufacturer ID, and quantity produced.

