**NEPAL COLLEGE OF INFORMATION TECHNOLOGY**

Balkumari, Lalitpur

*Affiliated to Pokhara University*

**LAB REPORT FOR DATABASE MANAGEMENT SYSTEM**



**LAB REPORT 6 and 7: FAMILIARIZATION WITH TCL COMMAND AND DQL COMMANDS**

Submitted by: Submitted to:

Name: Arpan Adhikari Er. Simanta Kasaju

Roll No.: 231309

BE-CE (3rd SEM)

|  |  |  |  |
| --- | --- | --- | --- |
| **S. N** | **LAB TOPIC** | **DATE** | **REMARKS** |
| **1.** | **INTRODUCTION TO SQL** | 10th Nov |  |
| **2.** | **FAMILIARIZATION WITH DDL COMMAND** | 17th Nov |  |
| **3.** | **FAMILIARIZATION WITH VARIOUS CONSTRAINTS IN SQL** | 24th Nov |  |
| **4.** | **FAMILIARIZATION WITH DML C­­OMMANDS (INSERT, UPDATE, DELETE)** | 1st Dec |  |
| **5.** | **FAMILIARIZATION WITH DCL C­­OMMANDS (GRANT, REVOKE)** | 8th Dec |  |
| **6,7.** | **FAMILIARIZATION WITH TCL COMMAND AND DQL COMMANDS** | 30th Dec |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**LAB 6,7: FAMLIARIZATION WITH TCL COMMAND AND DQL COMMANDS**

**THEORY:**

Transaction Control Language (TCL) commands are used in SQL to manage transactions within a database. A transaction in a database is a sequence of one or more SQL operations performed as a single logical unit of work. TCL commands help ensure the integrity of data by allowing users to control the changes made by their SQL queries. These commands are crucial for managing data consistency and maintaining reliability in database operations.

There are three primary TCL commands:s

1. COMMIT:  
   The COMMIT command is used to save all the changes made during the current transaction to the database. Once a COMMIT is executed, the changes become permanent and visible to other users. It marks the successful completion of a transaction and ensures that the database remains in a consistent state. For example, after updating or inserting data, using the COMMIT command will confirm those changes.

Example:

UPDATE students SET marks = 85 WHERE id = 1;

COMMIT;

1. ROLLBACK:  
   The ROLLBACK command is used to undo changes made during the current transaction before a COMMIT is issued. It reverts the database to its previous state, discarding all uncommitted changes. This command is useful when an error occurs during a transaction, and you want to undo the changes to maintain data integrity.

Example:

UPDATE students SET marks = 90 WHERE id = 2;

ROLLBACK;

In this case, the marks change will be discarded, and the original value will be retained.

1. SAVEPOINT:  
   The SAVEPOINT command is used to set a point within a transaction to which the user can later ROLLBACK. It allows partial rollback of a transaction instead of rolling back the entire transaction. SAVEPOINT is useful in complex transactions where specific changes need to be undone without affecting the entire transaction.

Example:

SAVEPOINT sp1;

UPDATE students SET address = 'Pokhara' WHERE id = 3;

ROLLBACK TO sp1;

In this case, the ROLLBACK TO sp1 command will revert the changes made after the SAVEPOINT.

**Importance of TCL commands:**  
TCL commands are essential for maintaining the integrity of data in multi-user database environments. They help prevent data corruption by ensuring that only valid transactions are permanently applied to the database. They also provide mechanisms for handling errors and managing complex transactions effectively.

In summary, TCL commands play a vital role in database management by allowing users to control transactions, ensuring that data remains consistent, reliable, and secure. Understanding these commands is crucial for database administrators and developers who need to manage data efficiently in real-world applications.

Data Query Language (DQL) is a subset of SQL commands used to retrieve data from a database. Unlike other SQL commands, DQL focuses only on querying and extracting information from the database without modifying the underlying data. DQL commands help users fetch records based on specific conditions and criteria, allowing them to analyze and manipulate the retrieved results.

The primary DQL command is:

1. **SELECT**  
   The SELECT command is used to retrieve data from one or more tables in a database. It allows users to specify the columns they want to retrieve, apply filters to narrow down the results, and sort the data as required. The SELECT command is one of the most commonly used SQL commands, as it provides flexible ways to fetch data based on various conditions.

Syntax:

SELECT column1, column2, ...

FROM table\_name

WHERE condition;

In the SELECT statement, specific columns can be mentioned to retrieve only those columns from the table. The WHERE clause is used to filter the results based on a given condition. If the WHERE clause is omitted, the query will return all records from the specified table.

Example:

SELECT name, address

FROM students

WHERE marks > 70;

In this query, the SELECT command retrieves the name and address of students who have scored more than 70 marks.

**Advanced Features of SELECT Command:**

* **DISTINCT:** This keyword is used to eliminate duplicate records from the result set.  
  Example:

SELECT DISTINCT faculty

FROM students;

* **ORDER BY:** This clause is used to sort the result set in ascending or descending order.  
  Example:

SELECT name, marks

FROM students

ORDER BY marks DESC;

* **GROUP BY:** This clause groups the records that have the same values in specified columns. It is typically used with aggregate functions like COUNT, SUM, AVG, etc.  
  Example:

SELECT faculty, COUNT(\*)

FROM students

GROUP BY faculty;

* **HAVING:** The HAVING clause is used to filter the grouped records after applying the GROUP BY clause.  
  Example:

SELECT faculty, AVG(marks)

FROM students

GROUP BY faculty

HAVING AVG(marks) > 70;

**Importance of DQL Commands:**  
DQL commands are essential in data retrieval and analysis. They allow users to query large datasets efficiently, apply filters, sort the results, and group data based on specific conditions. In real-world applications, DQL commands are frequently used to generate reports, perform data analysis, and extract meaningful insights from databases.

In summary, DQL commands like SELECT provide powerful tools for retrieving data from a database. These commands help users to fetch, filter, and present data in a structured way, making them an essential part of SQL for managing databases effectively.

**Question 1:**

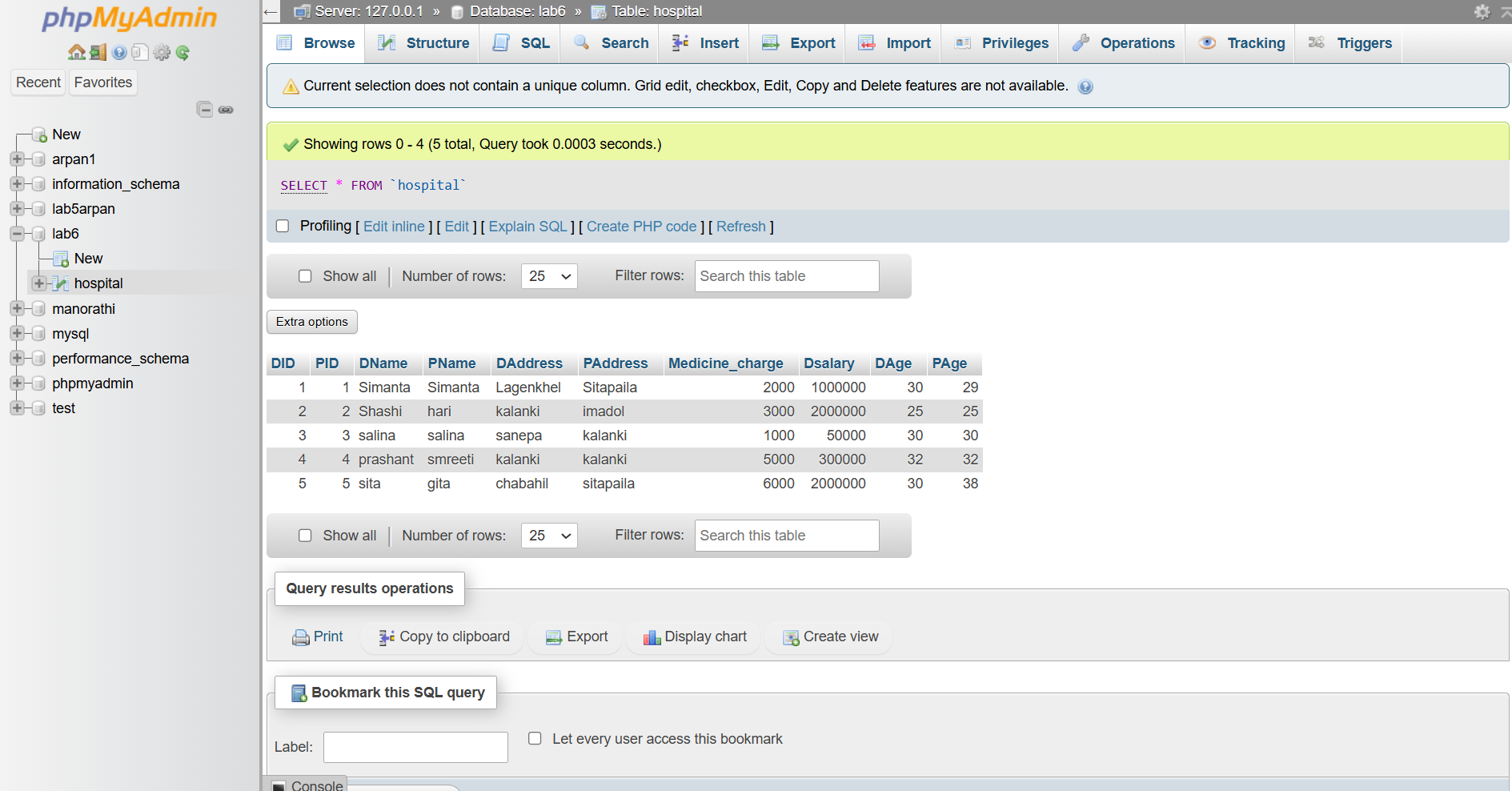
**1. Create a table Hospital having attribute. D\_Id, D\_name, p\_Id, p\_name, D\_address, p\_address,medicine charge, dr salary, p\_age, d \_age. Insert at least five data into the table.**

create TABLE Hospital( DID int, PID int, DName varchar(20), PName varchar(20), DAddress varchar(20), PAddress varchar(20), Medicine\_charge bigint, Dsalary bigint, DAge int, PAge int );

INSERT INTO hospital

VALUES

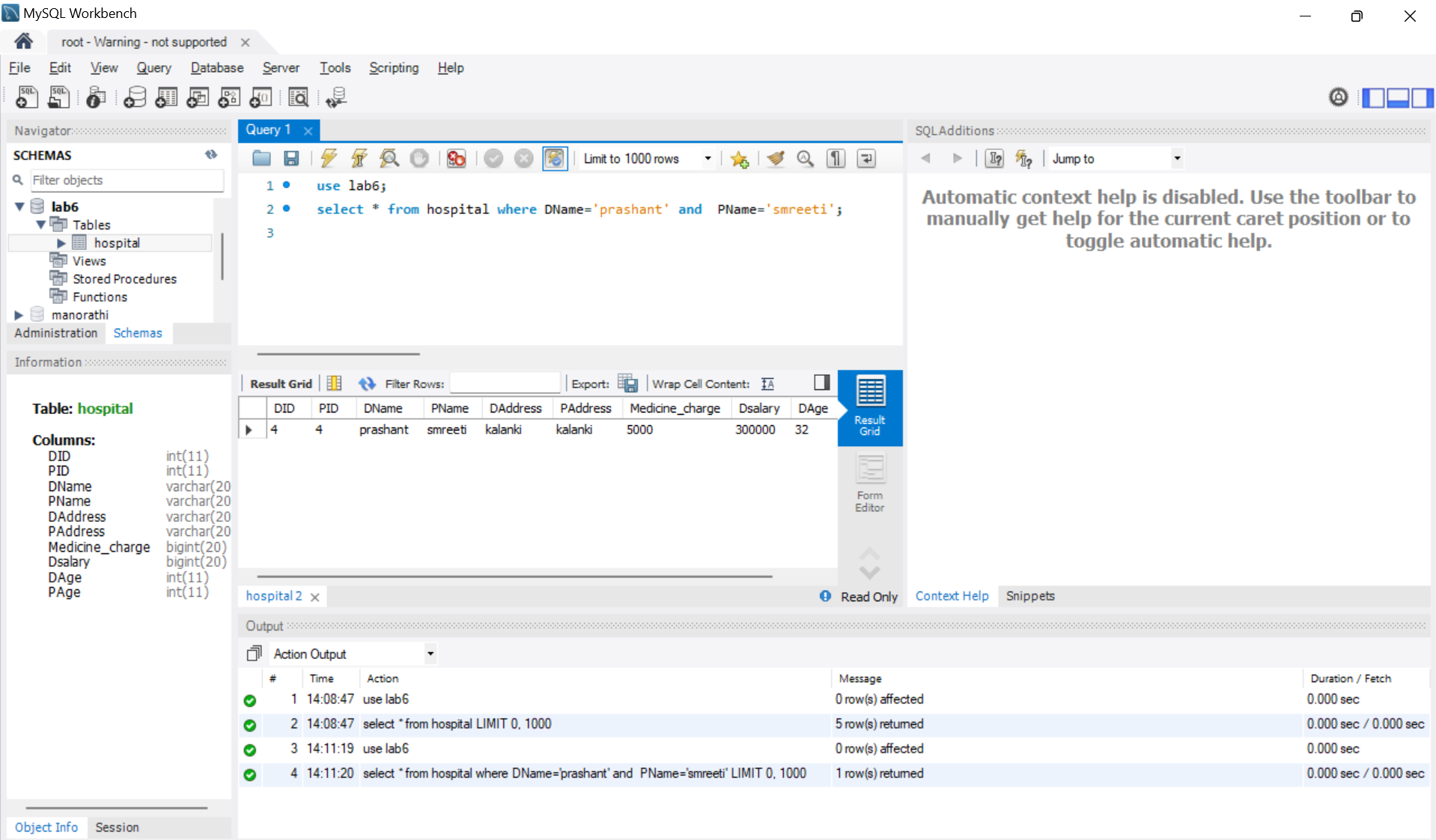
(1,1,"Simanta","Simanta","Lagenkhel","Sitapaila",2000,1000000,30,29), (2,2,"Shashi","hari","kalanki","imadol",3000,2000000,25,25 ), (3,3,"salina","salina","sanepa","kalanki",1000,50000,30,30 ), (4,4,"prashant","smreeti","kalanki","kalanki",5000,300000,32,32 ), (5,5,"sita","gita","chabahil","sitapaila",6000,2000000,30,38 );



**2. Select the info having name prashant and smreeti as Dr and patient name.**

use lab6;

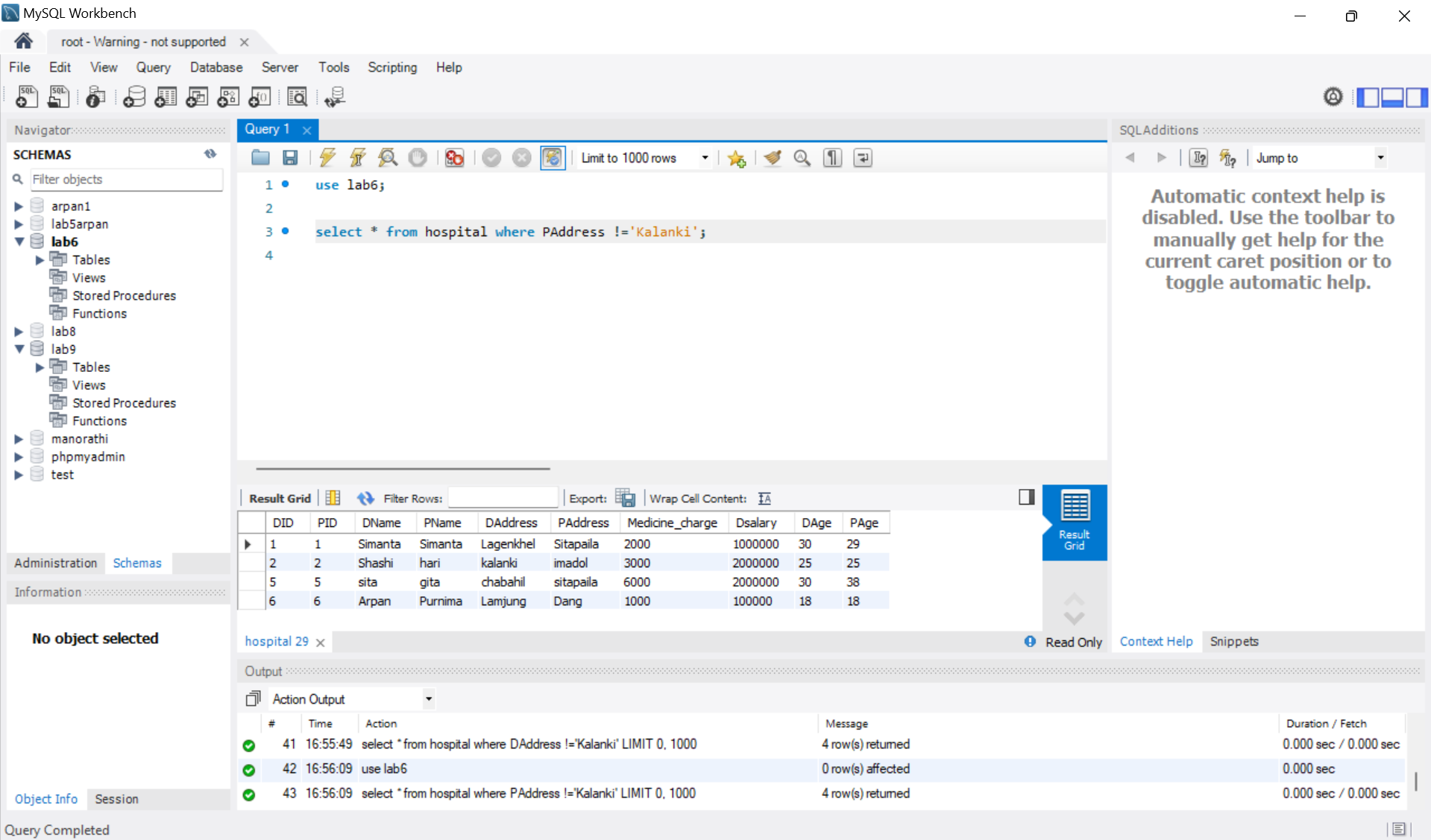
select \* from hospital where DName='prashant' and PName='smreeti';



1. **Select the info of patient != who does not belong to kalanki.**

use lab6;

select \* from hospital where PAddress !='Kalanki';



1. **Select dr\_salary which age<=30**

use lab6;

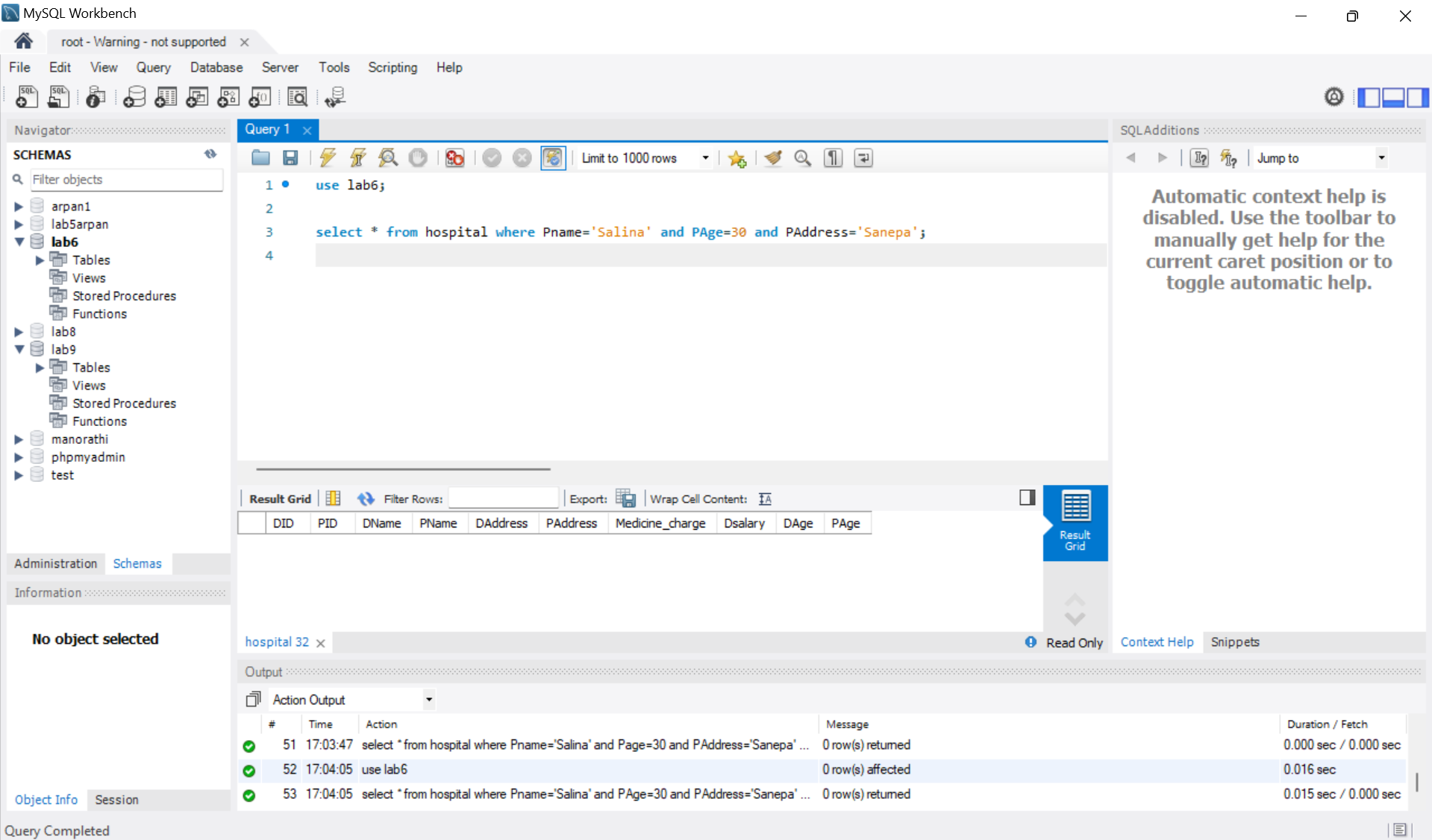
select Dsalary from hospital where DAge<=30;

****

1. **Select all the information of patient who is salina from ‘sanepa’ and age is 30.**

use lab6;

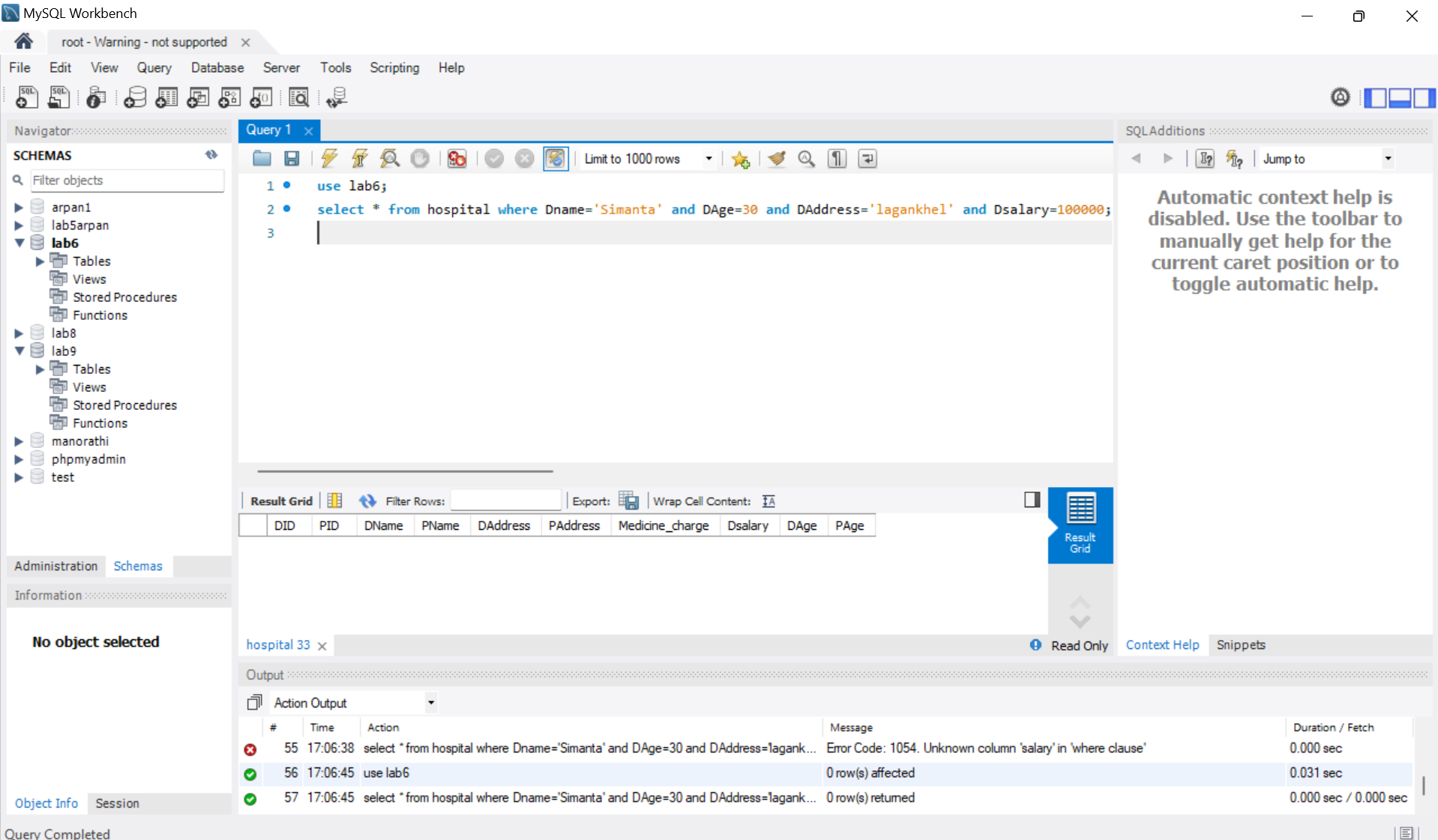
select \* from hospital where Pname='Salina' and PAge=30 and PAddress='Sanepa';

****

1. **SELECT ALL THE information of doctor who is simanta age=30 and address=lagankhel and salary=100000.**

use lab6;

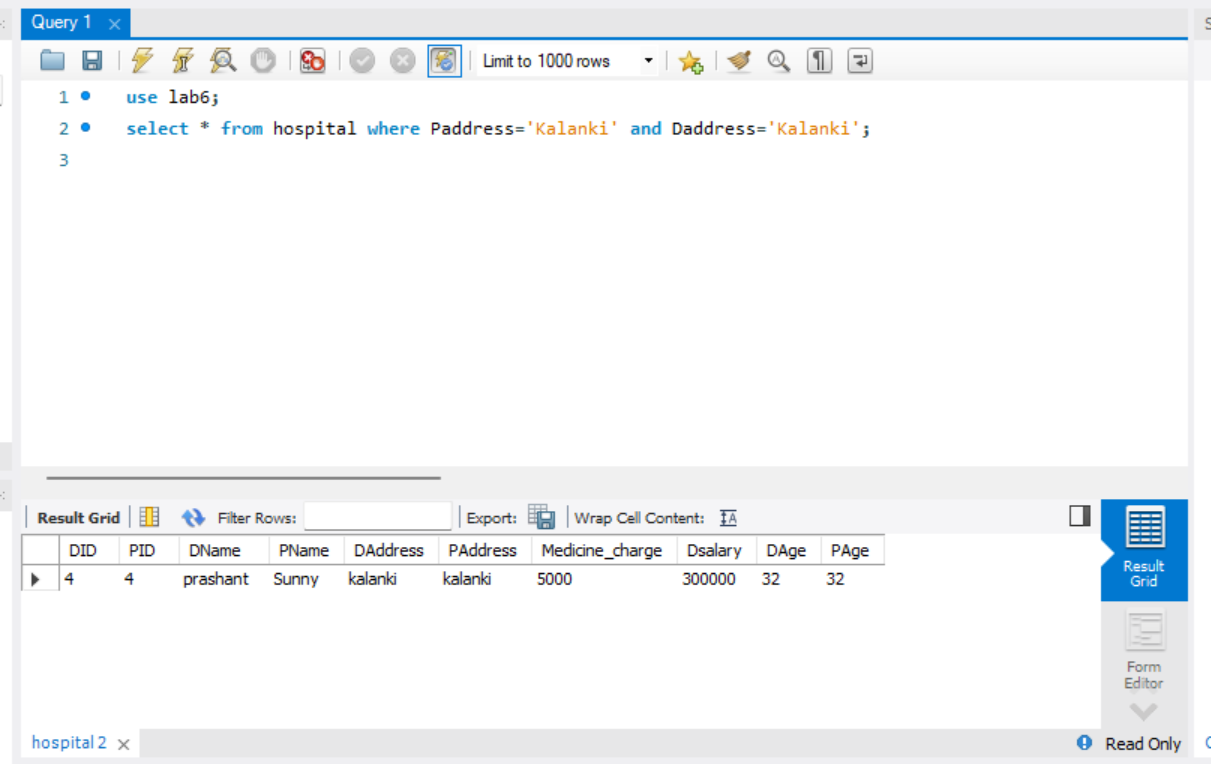
select \* from hospital where Dname='Simanta' and DAge=30 and DAddress='lagankhel' and Dsalary=100000;

****

1. **Select the info of patient and doctor having address kalanki of both.**

use lab6;

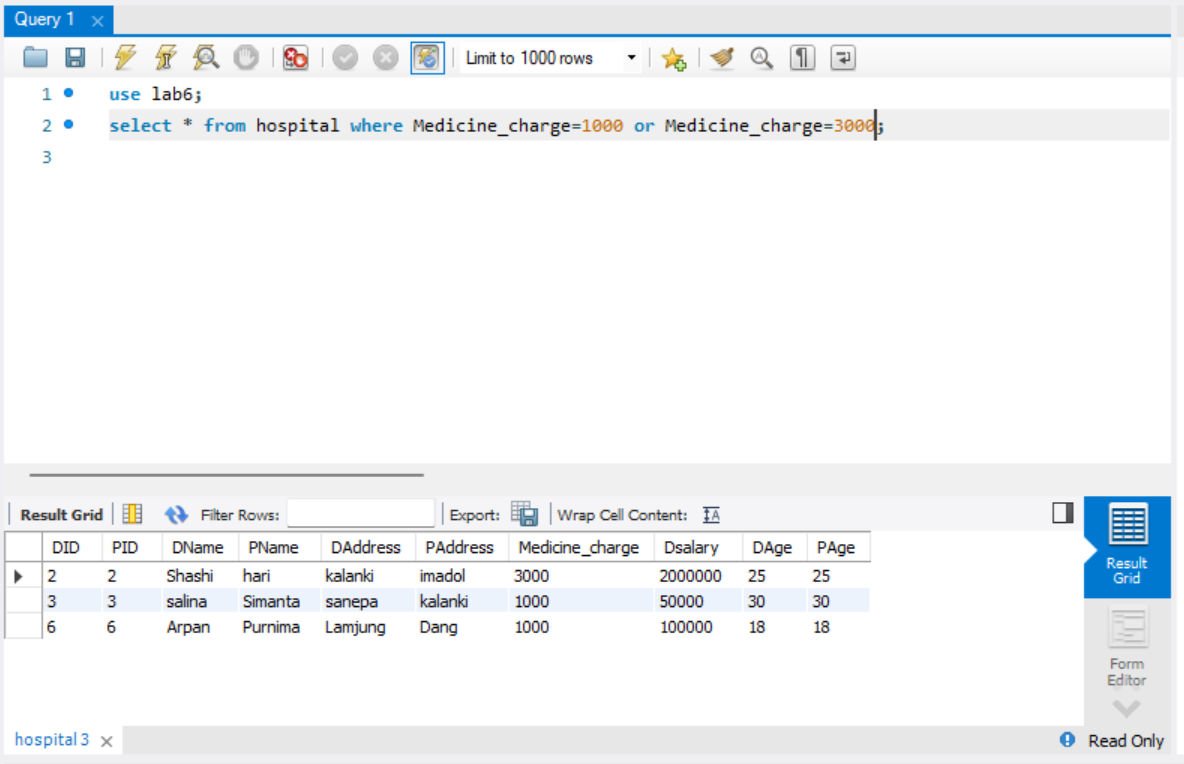
select \* from hospital where Paddress='Kalanki' and Daddress='Kalanki';

****

1. **Select the info which medical charge is 1000 or 3000.**

**use lab6;**

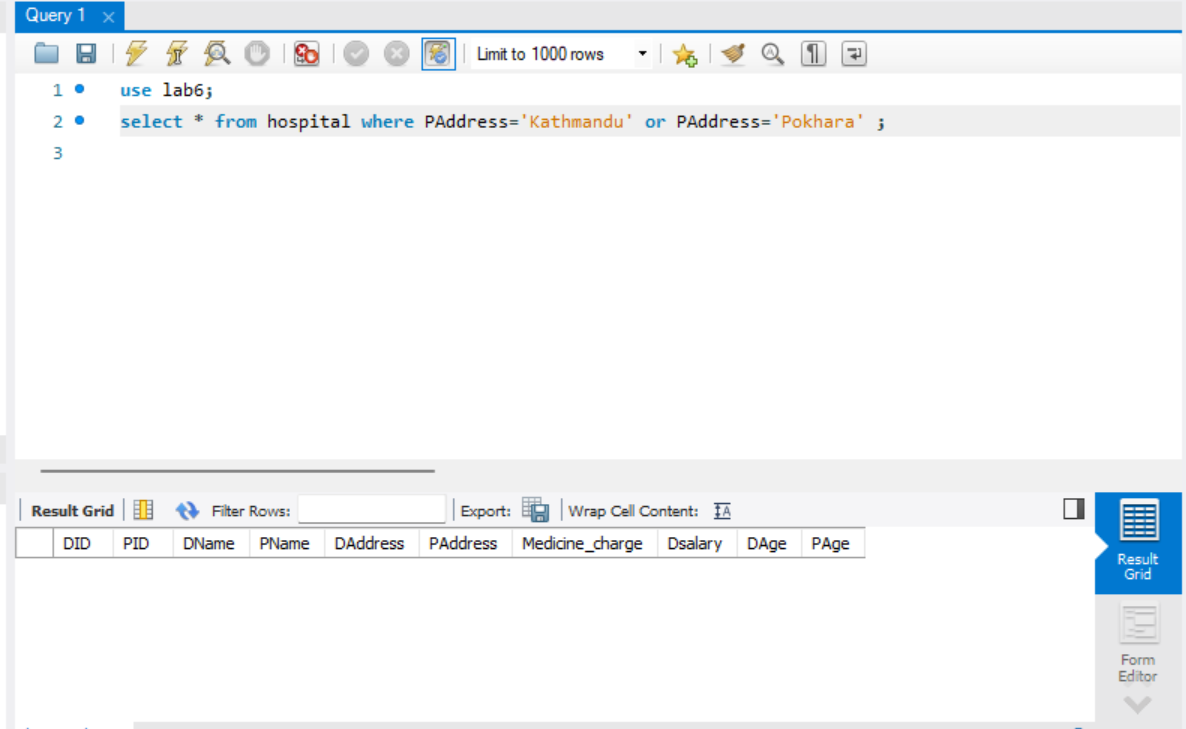
**select \* from hospital where Medicine\_charge=1000 or Medicine\_charge=3000;**

****

1. **Select info of the patient having address is ‘ktm’ or ‘pkr’.**

**use lab6;**

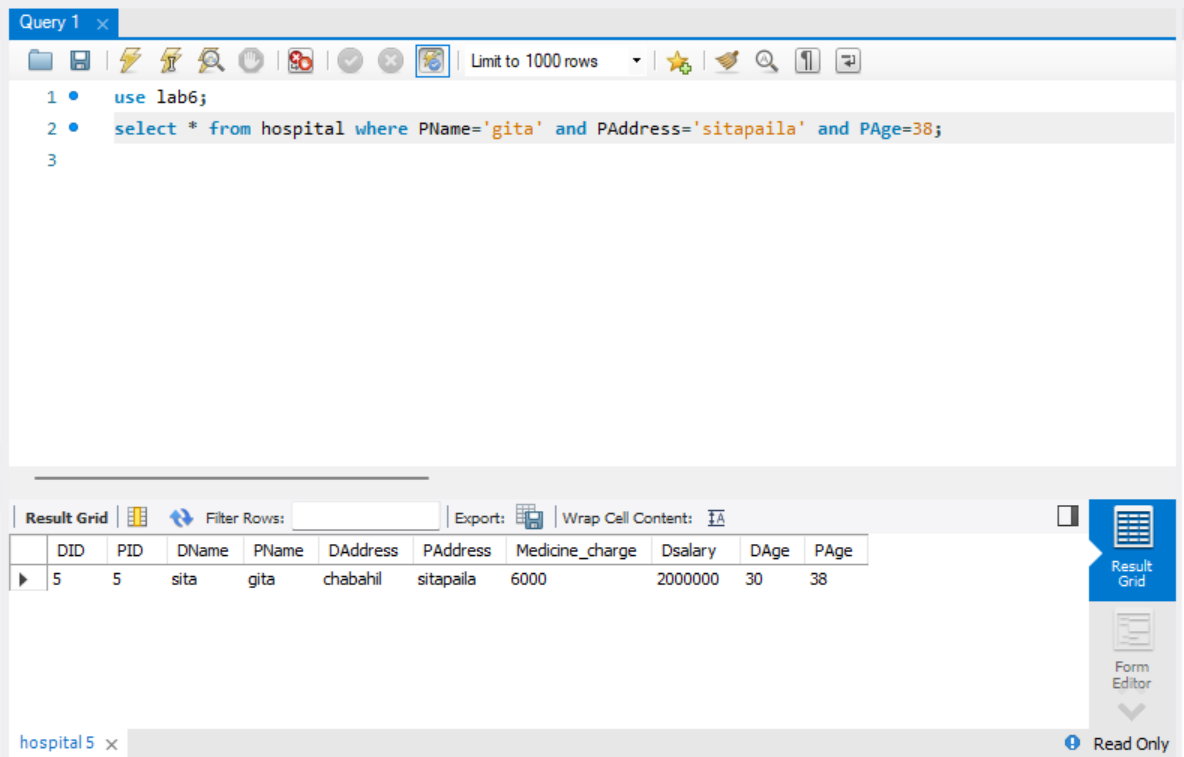
**select \* from hospital where PAddress='Kathmandu' or PAddress='Pokhara';**

****

1. **Select patient name who is gita from ‘sitapaila’ and age is 38.**

**use lab6;**

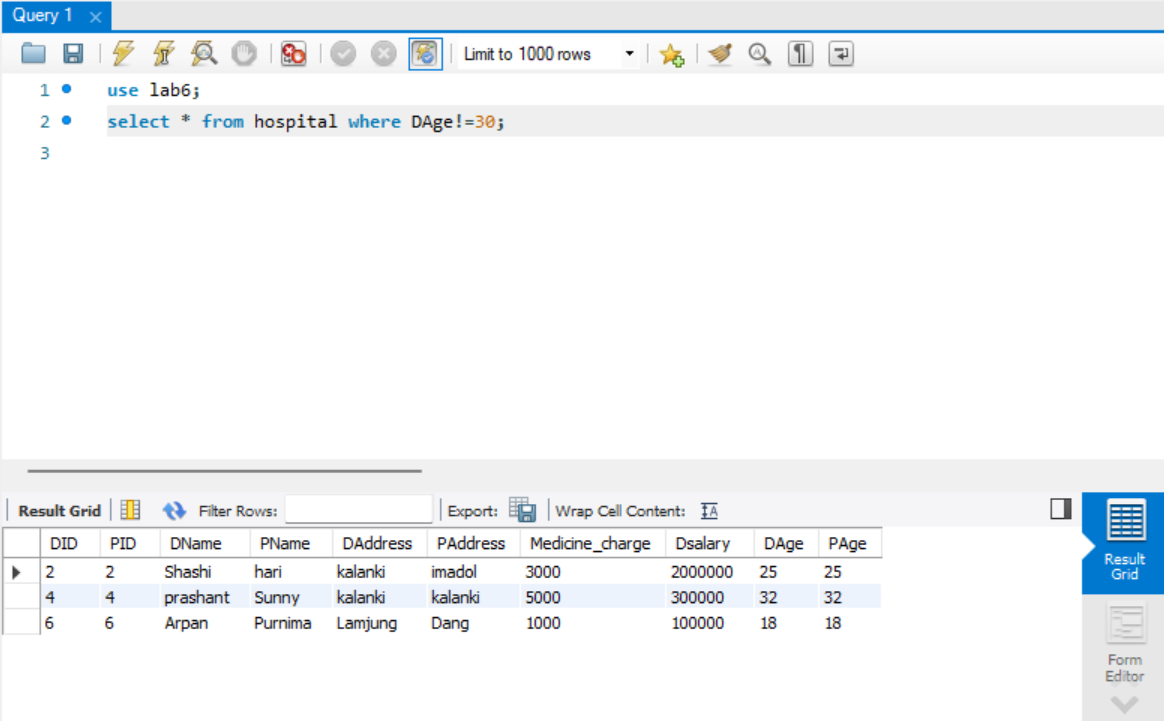
**select \* from hospital where PName='gita' and PAddress='sitapaila' and PAge=38;**

****

1. **Select the info dr whose age is not is 30.**

**use lab6;**

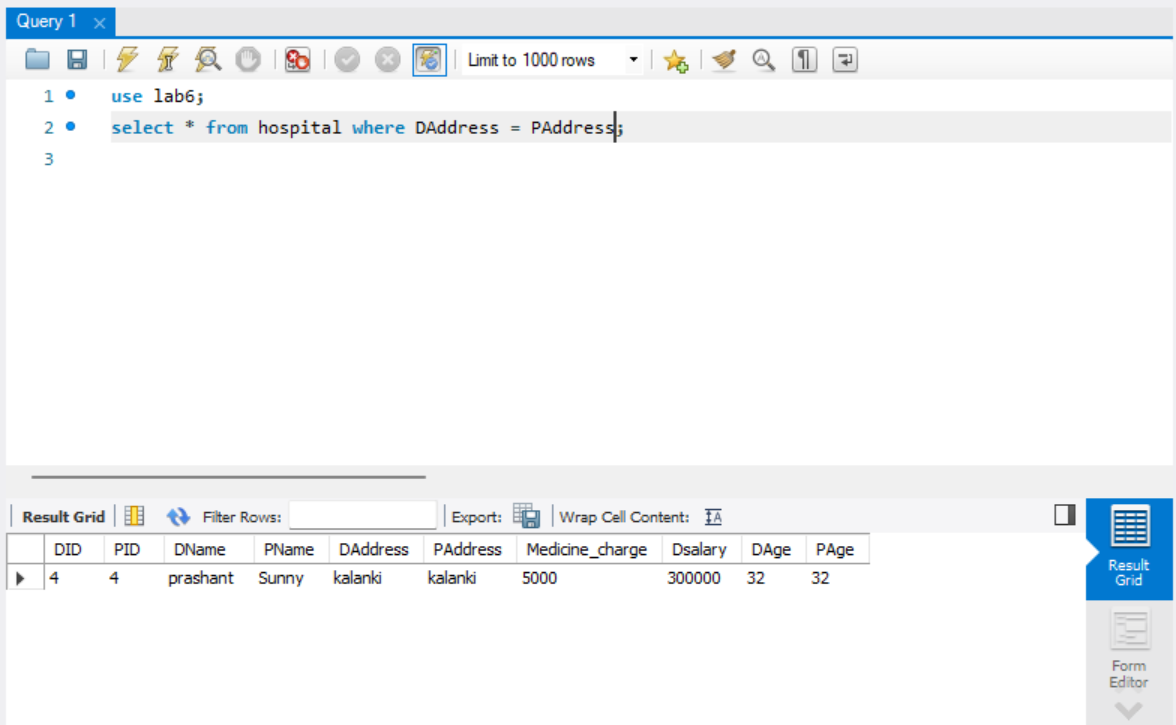
**select \* from hospital where DAge!=30;**

****

1. **Select the info of patient and doctor having same address.**

**use lab6;**

**select \* from hospital where DAddress = PAddress;**

****

1. **Select the info of patient and doctor having different address.**

**use lab6;**

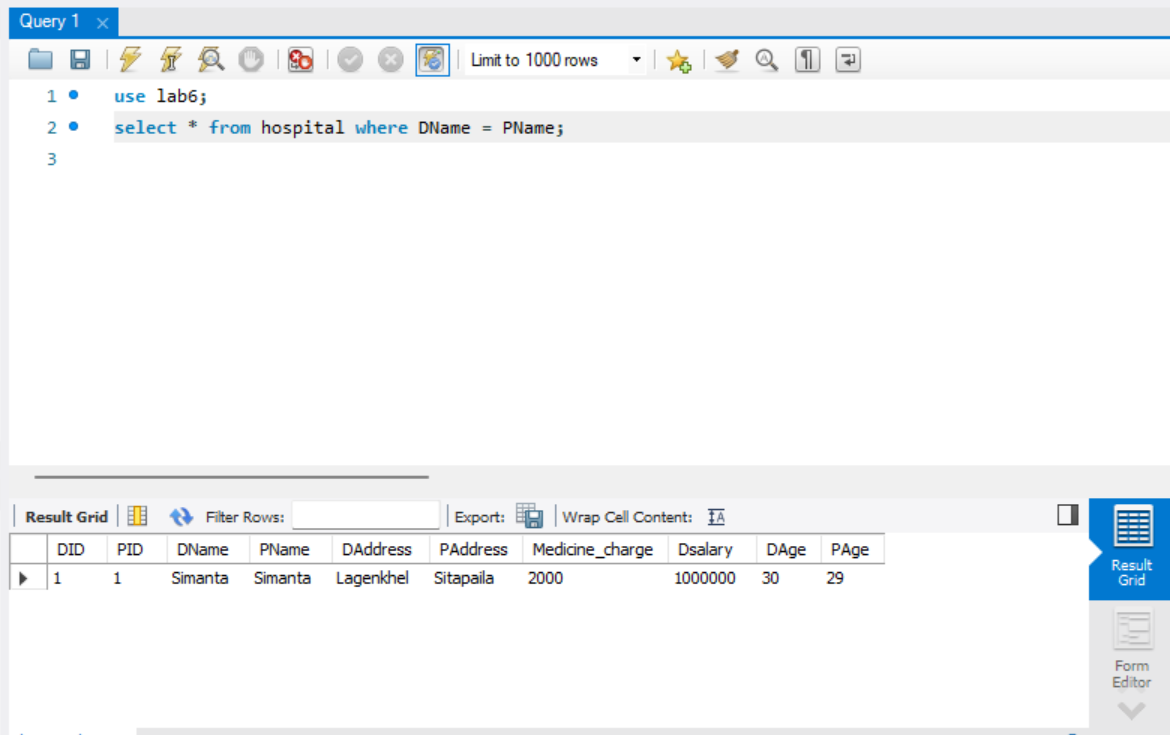
**select \* from hospital where DAddress != PAddress;**

****

1. **Select the info of patient and doctor having same name.**

**use lab6;**

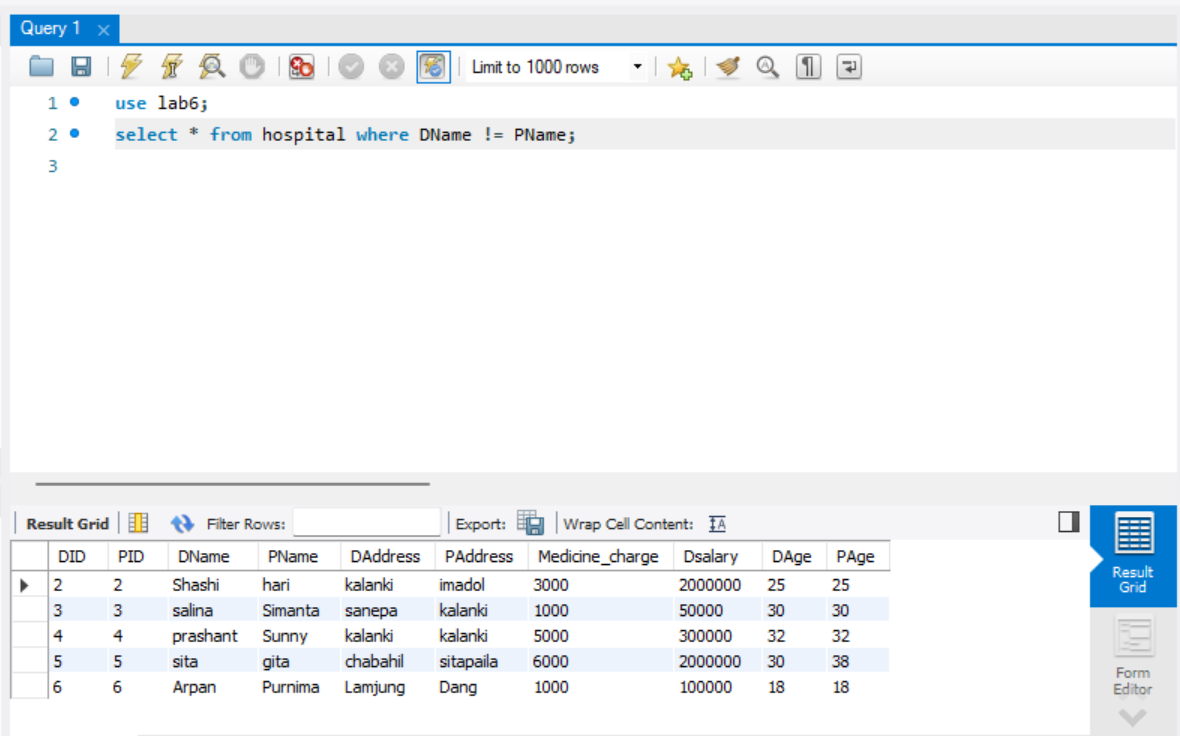
**select \* from hospital where DName = PName;**

****

1. **Select the info of patient and doctor having different name.**

**use lab6;**

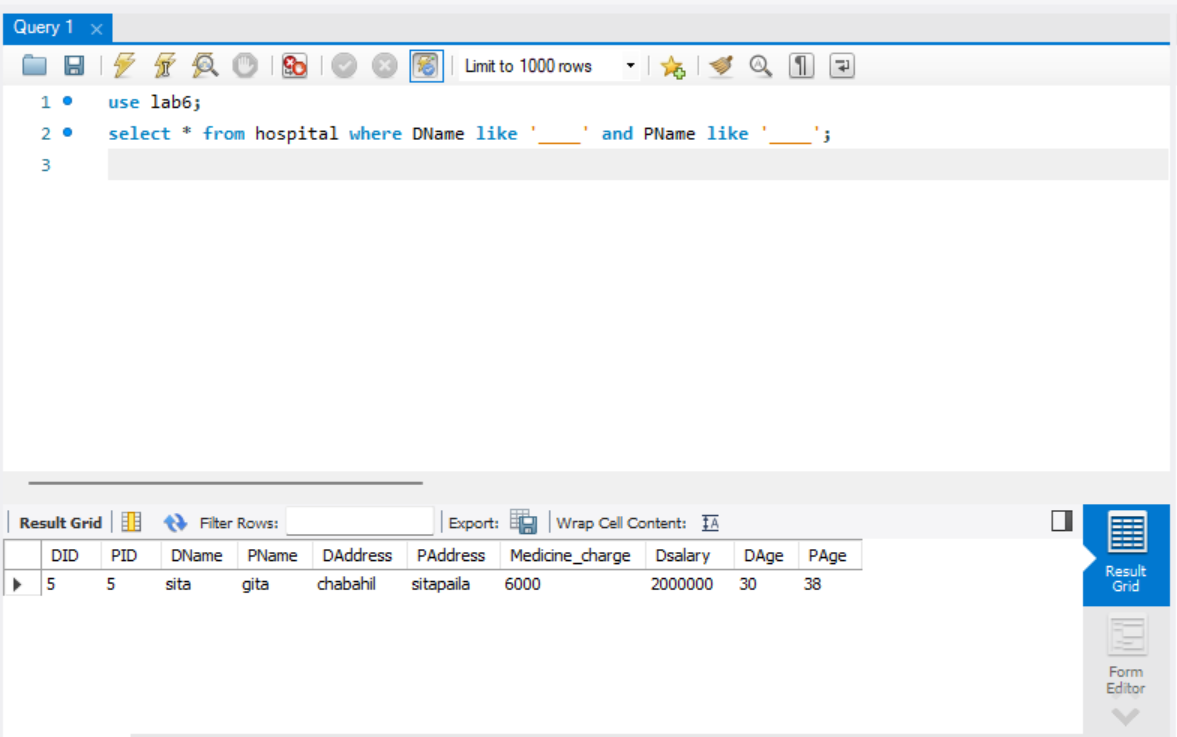
**select \* from hospital where DName != PName;**

****

1. **Select all the information where dname and pname having 4 characters in their name.**

**use lab6;**

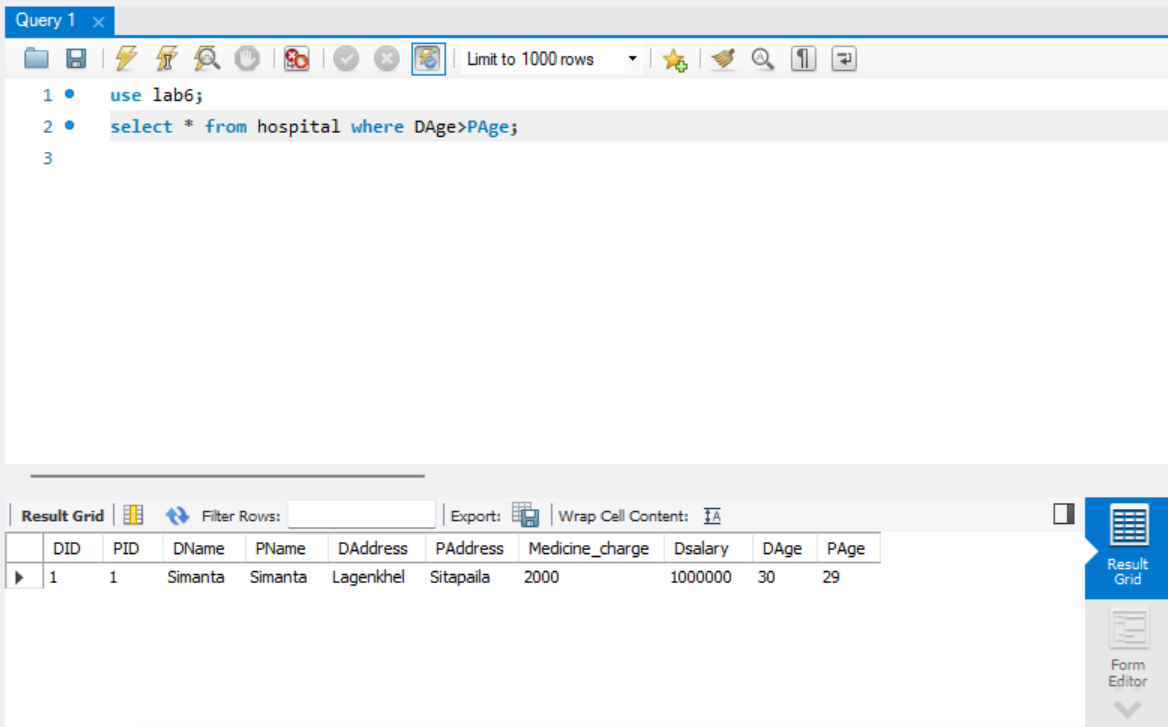
**select \* from hospital where DName like '\_\_\_\_' and PName like '\_\_\_\_';**

****

1. **Select the information of doctor and patient where doctor age is greater than patient age.**

**use lab6;**

**select \* from hospital where DAge>PAge;**

****

1. **After reading 5 info use commit command to SAVE IT. Delete the record of the patient whose id 3 and rollback.**

**use lab6;**

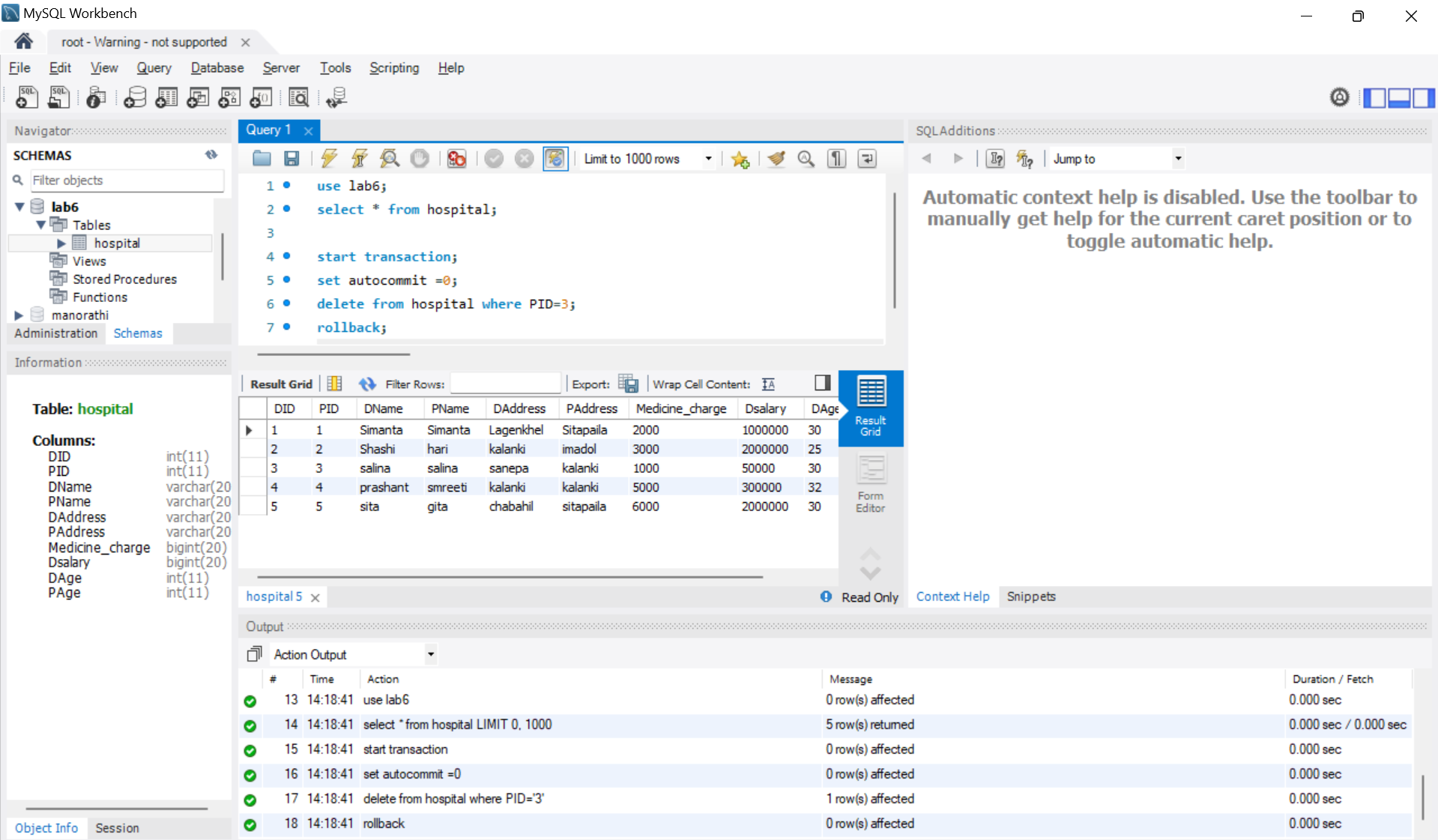
**select \* from hospital;**

**start transaction;**

**set autocommit=0;**

**delete from hospital where ID=3;**

**rollback;**



**19. Update the p\_name to simanta who is from kalanki and commit,update d\_name to hari whose id is 4 and rollback and find the result.**

**use lab6;**

**start transaction;**

**set autocommit =0;**

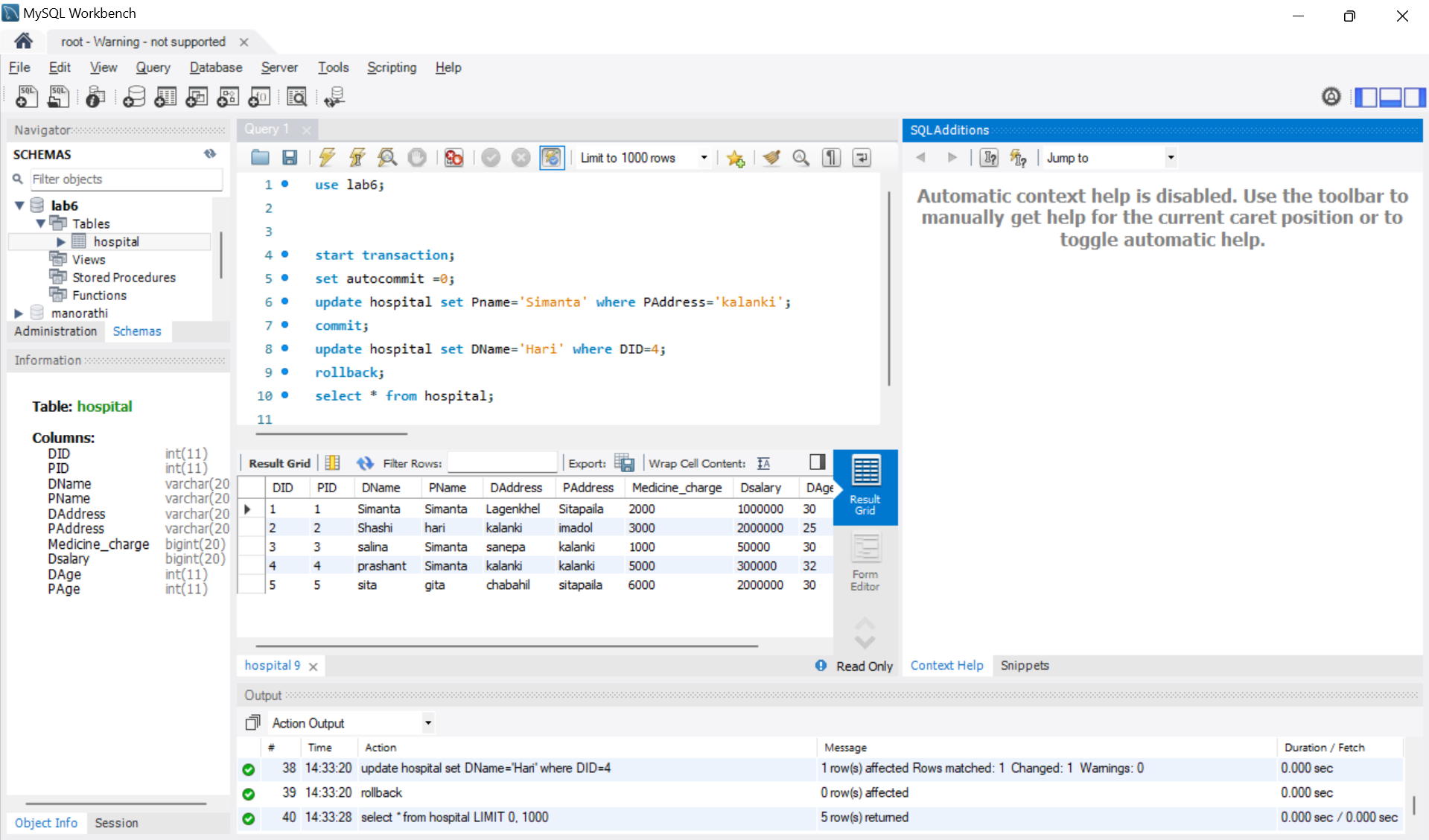
**update hospital set Pname='Simanta' where PAddress='kalanki';**

**commit;**

**update hospital set DName='Hari' where DID=4;**

**rollback;**

**select \* from hospital;**



**20. Do insert, update, delete whatever you want and commit and rollback the last transaction.**

**use lab6;**

**start transaction;**

**set autocommit =0;**

**insert into hospital values (6, 6, 'Arpan', 'Purnima', 'Lamjung', 'Dang', 1000, 100000, 18, 18);**

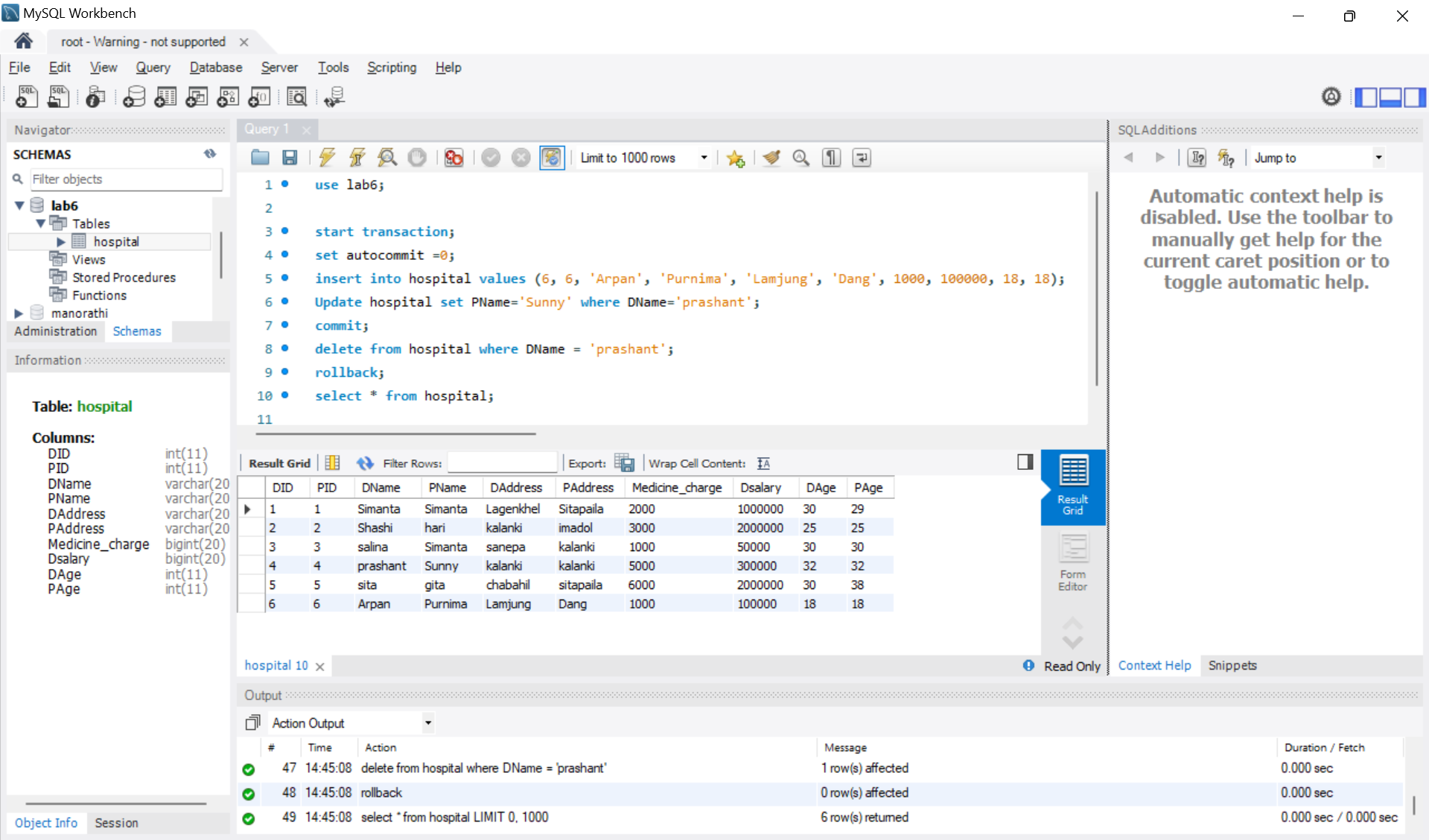
**Update hospital set PName='Sunny' where DName='prashant';**

**commit;**

**delete from hospital where DName = 'prashant';**

**rollback;**

**select \* from hospital;**



**21. Update p\_age to 45 whose id =2 and make it save point A, update d\_age 90 whose id=4 make it save point B and finally insert new record and rollback to b and see the result.**

**use lab6;**

**start transaction;**

**set autocommit=0;**

**update hospital set PAge = 45 where PID = 2;**

**savepoint a;**

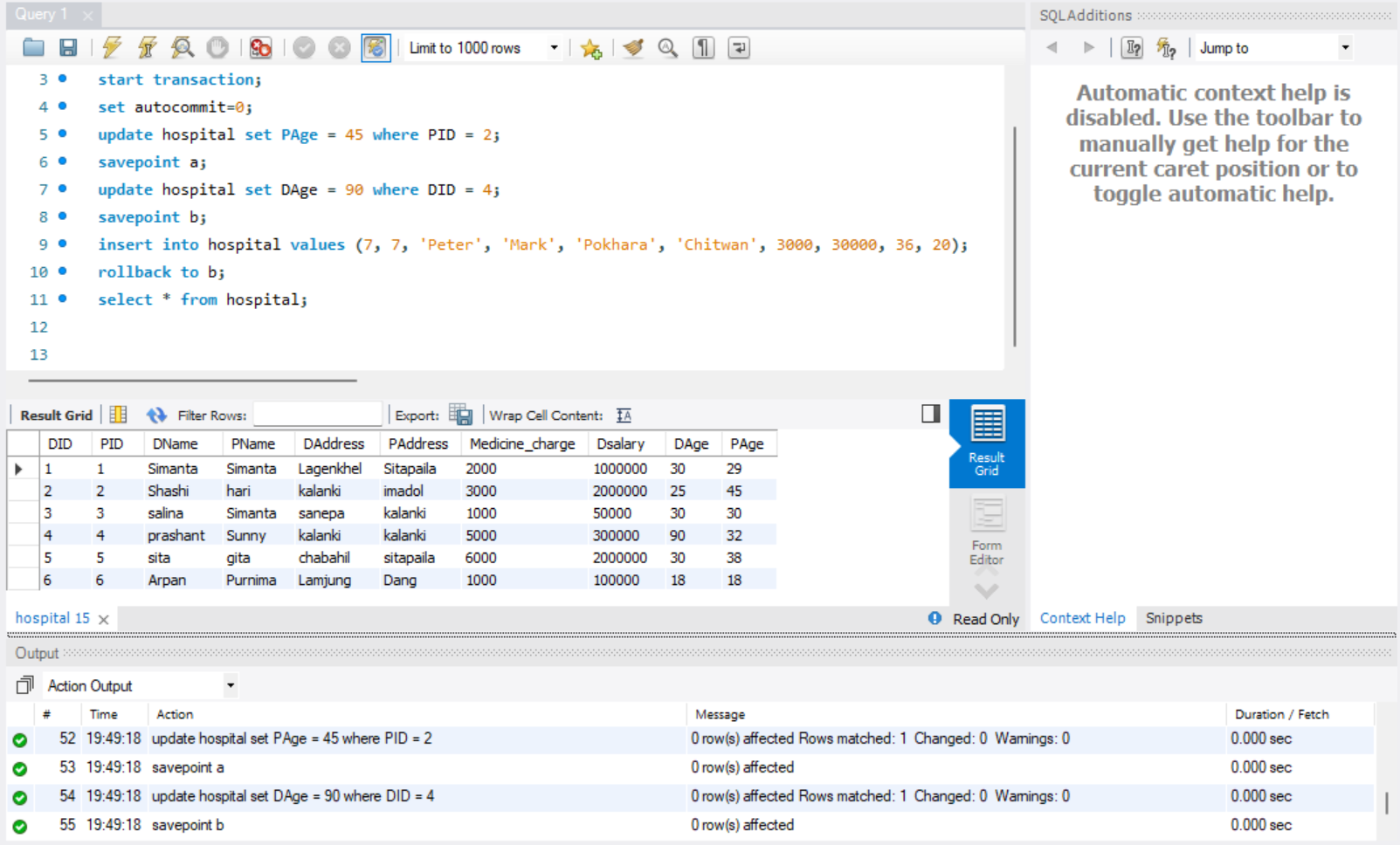
**update hospital set DAge = 90 where DID = 4;**

**savepoint b;**

**insert into hospital values (7, 7, 'Peter', 'Mark', 'Pokhara', 'Chitwan', 3000, 30000, 36, 20);**

**rollback to b;**

**select \* from hospital;**

****

**22. Create 3 save points A, B, C. insert new record and use savepoint A. Again insert new record and use save point B and lastly insert 1 new record and use save point C. Now try to roll back at the point you can’t access C.**

**USE lab6;**

**START TRANSACTION;**

**SET AUTOCOMMIT = 0;**

**INSERT INTO hospital**

**VALUES (8, 8, 'John', 'Doe', 'Kathmandu', 'Lalitpur', 5000, 40000, 40, 35);**

**SAVEPOINT A;**

**INSERT INTO hospital**

**VALUES (9, 9, 'Jane', 'Smith', 'Bhaktapur', 'Pokhara', 6000, 45000, 45, 30);**

**SAVEPOINT B;**

**INSERT INTO hospital**

**VALUES (10, 10, 'Alice', 'Brown', 'Chitwan', 'Dharan', 5500, 50000, 38, 28);**

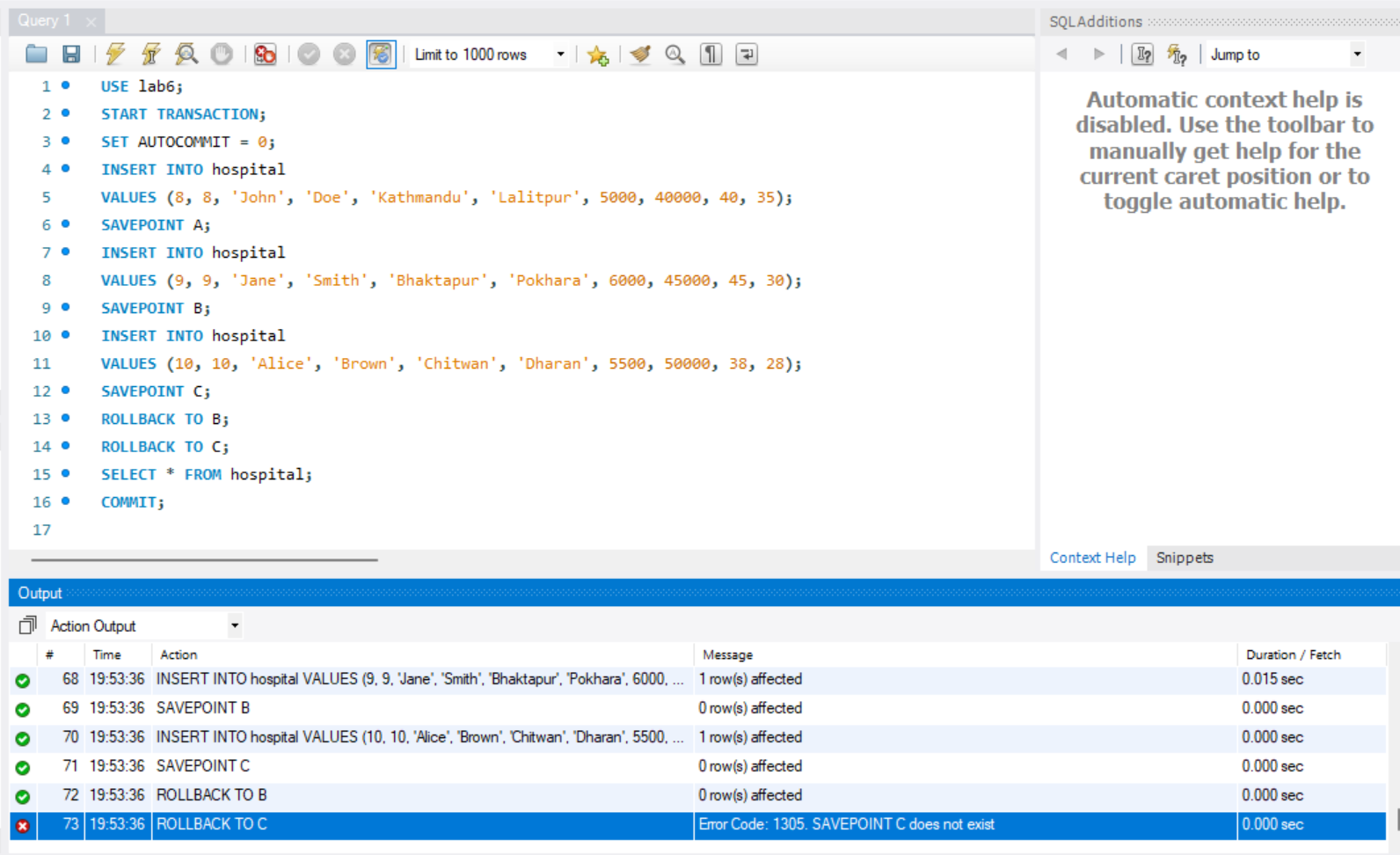
**SAVEPOINT C;**

**ROLLBACK TO B;**

**ROLLBACK TO C;**

**SELECT \* FROM hospital;**

**COMMIT;**

****

**23. Create a new table for student with id, name, gender, marks, and age.**

**create table student(**

**id int primary key,**

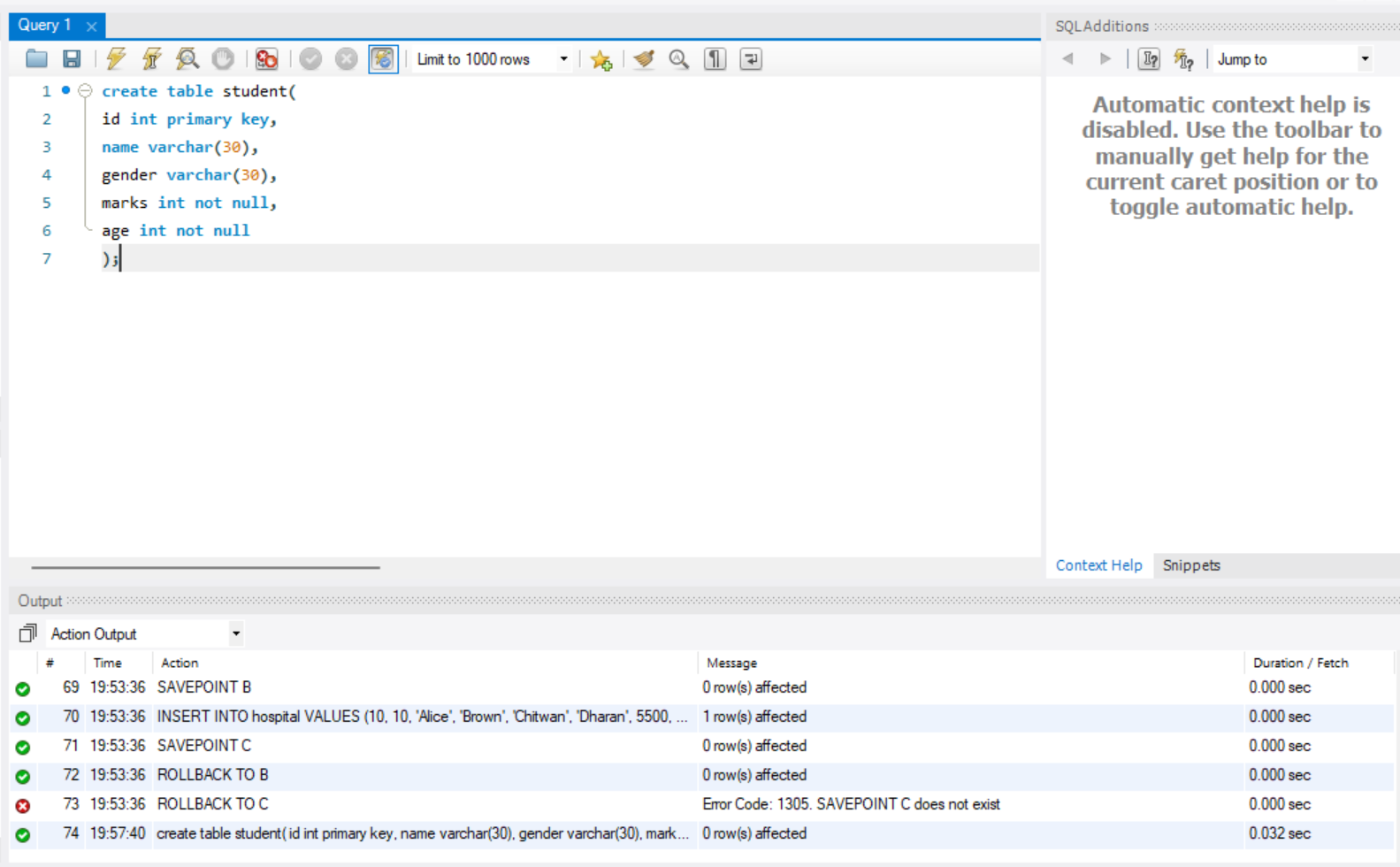
**name varchar(30),**

**gender varchar(30),**

**marks int not null,**

**age int not null**

**);**

****

**24. Insert any 5 records and commit.**

**INSERT INTO student (id, name, gender, marks, age)**

**VALUES**

**(1, 'Alice', 'Female', 85, 20),**

**(2, 'Bob', 'Male', 78, 22),**

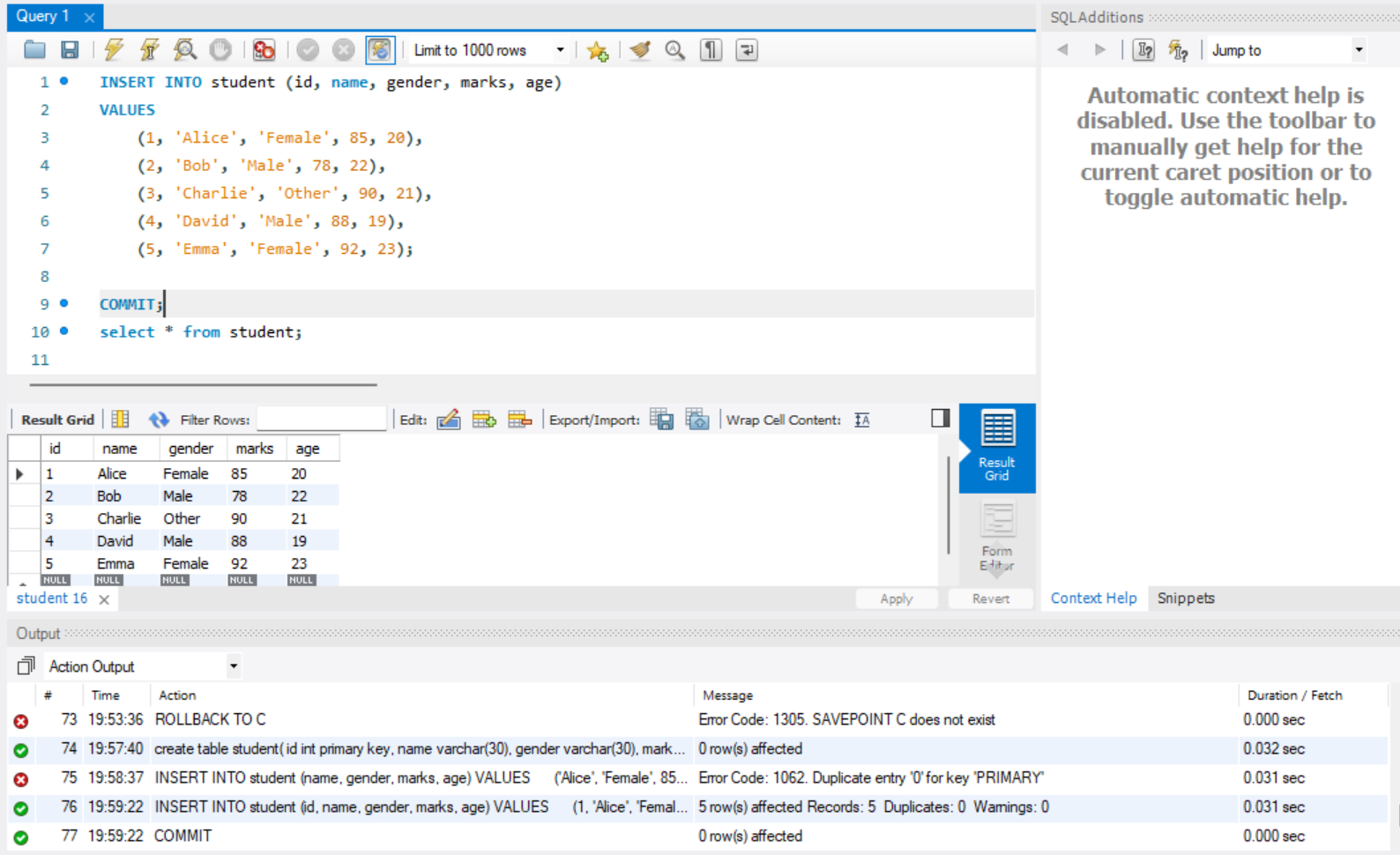
**(3, 'Charlie', 'Other', 90, 21),**

**(4, 'David', 'Male', 88, 19),**

**(5, 'Emma', 'Female', 92, 23);**

**COMMIT;**

**select \* from student;**

****

**25. Update the name of the student who is 19 years old to Hari and roll back it.**

**use lab6;**

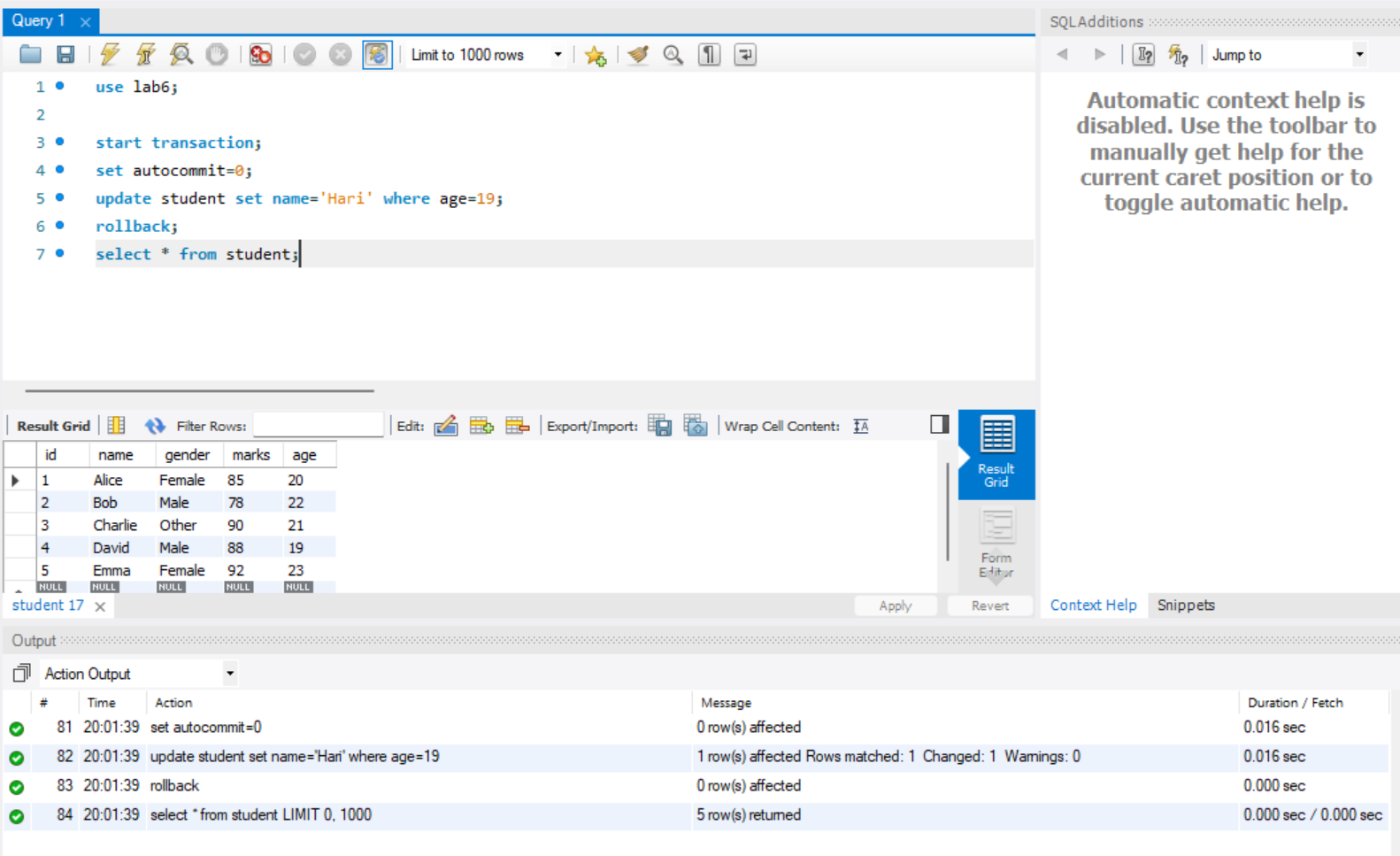
**start transaction;**

**set autocommit=0;**

**update student set name='Hari' where age=19;**

**rollback;**

**select \* from student;**

****

**26. Delete information of student who is male and rollback it.**

**use lab6;**

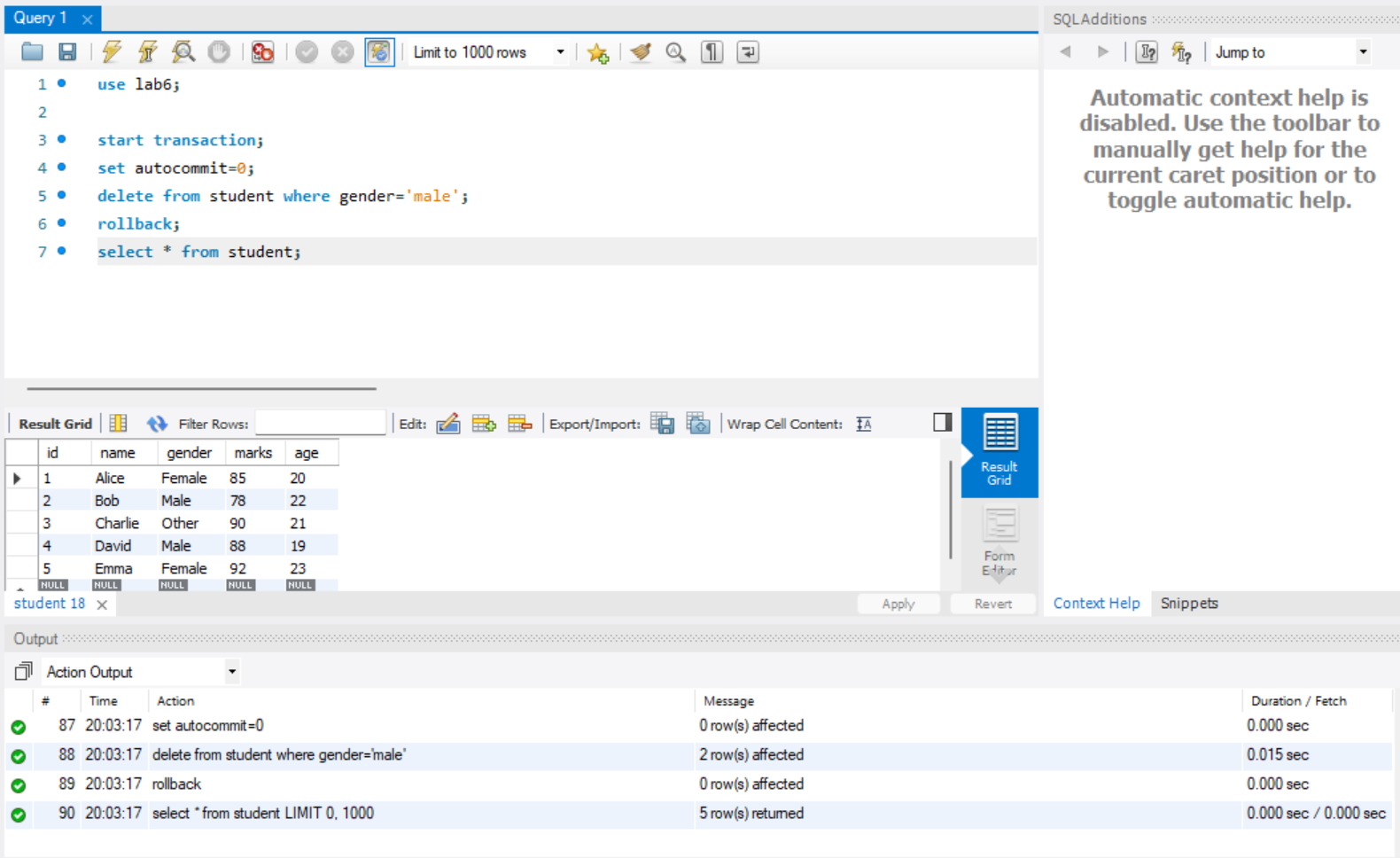
**start transaction;**

**set autocommit=0;**

**delete from student where gender='male';**

**rollback;**

**select \* from student;**

****

**27. Delete information who is >15 years old and make it save point P, update information of student who is 19 years old to 21 years old and make it save point Q again delete information of student who is male now rollback to P see the output. Q and R will rollback.**

**use lab6;**

**start transaction;**

**set autocommit=0;**

**delete from student where age>15;**

**savepoint P;**

**update student set age=21 where age=19;**

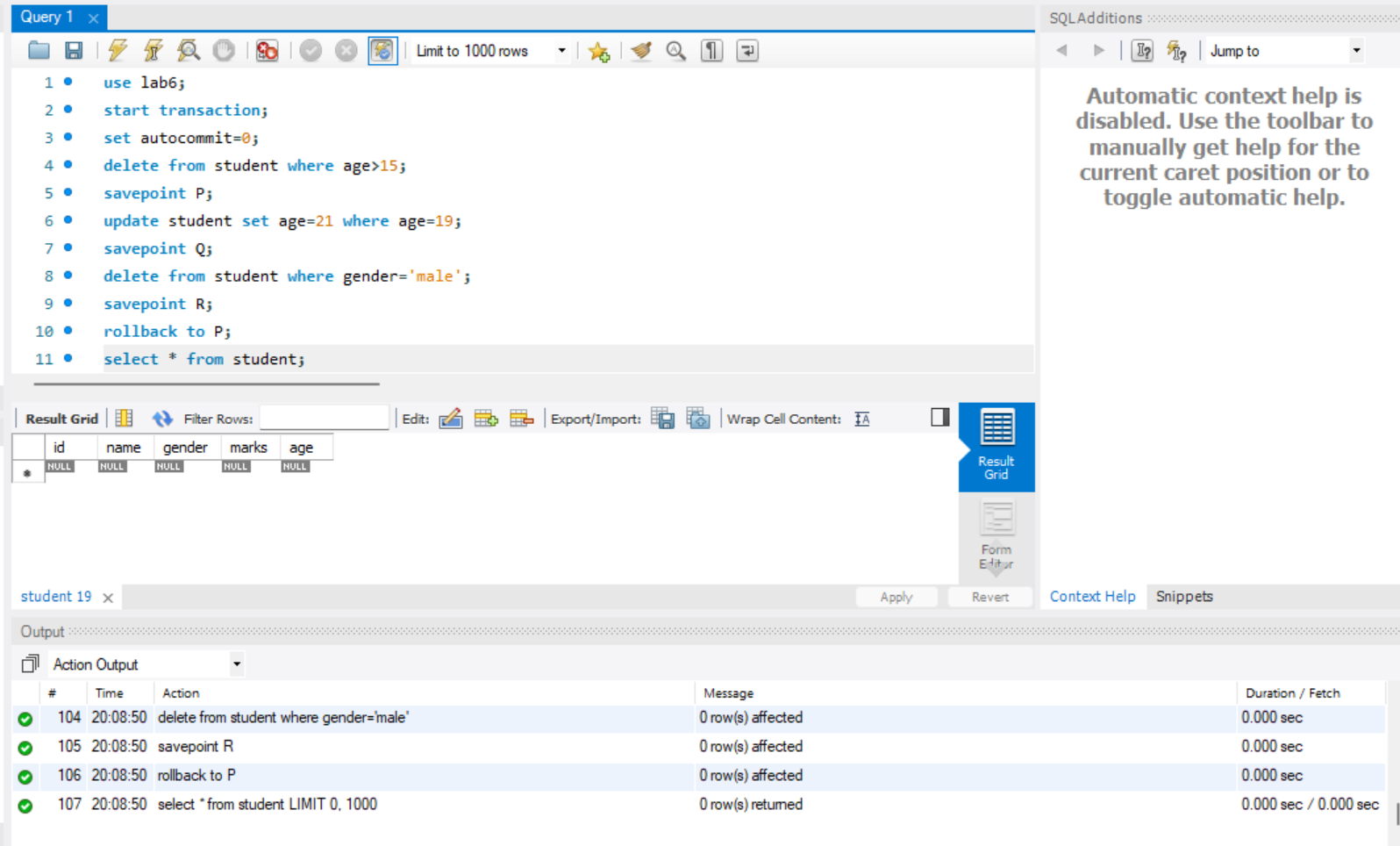
**savepoint Q;**

**delete from student where gender='male';**

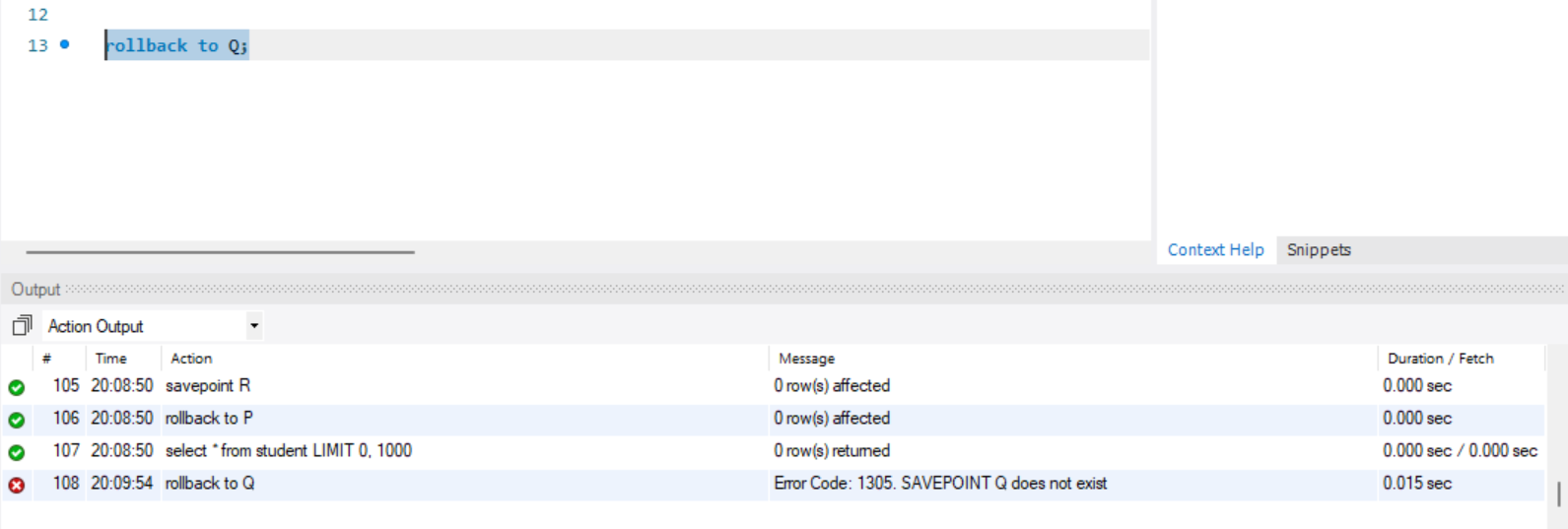
**savepoint R;**

**rollback to P;**

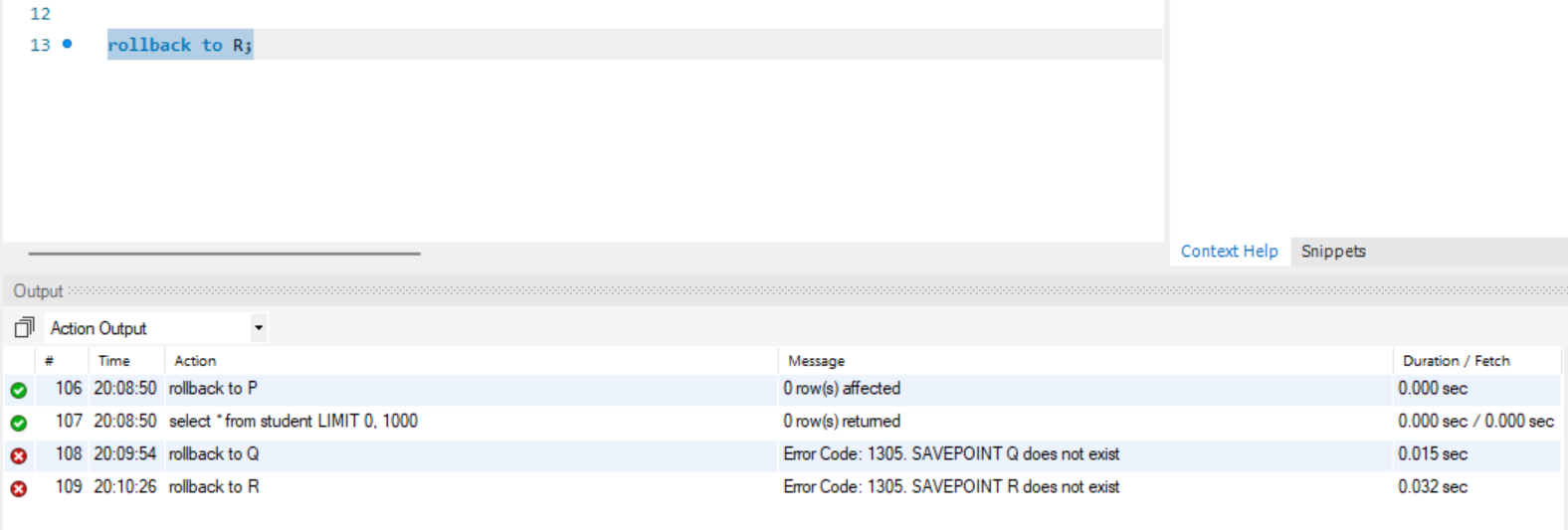
**select \* from student;**



**Rollback to Q:**

****

**Rollback to R:**



**CONCLUSION:**

In this practical, we learned how to use **Transaction Control Language (TCL)** and **Data Query Language (DQL)** commands in SQL. TCL commands like **COMMIT**, **ROLLBACK**, and **SAVEPOINT** help manage database transactions, ensuring data integrity and allowing us to undo changes when necessary. **DQL commands**, especially **SELECT**, are used to retrieve data from the database.

This practical enhanced our ability to control transactions and efficiently query data, both essential skills for working with databases in real-world applications.