Experiment-5: To Implement The Structure Of The Table

Student Name:- Neha Sharma **UID:- 20BCS4576**

Branch:- CSE-IOT **Section/Group:- A**

Semester:- 3rd **Date of Performance:- 23/09/2021**

Subject Name:- DBMS LAB **Subject Code:- 20CSP-233**

1. Aim/Overview of the practical:- To implement the structure of the table.

2. Task to be done:- Implementation of DDL and DML commands of SQL with proper Input queries syntax and the output.

3. Contraints in SQL:

1. Adding NOT NULL Constraint

Command:- NOT NULL

Purpose:- If you want to ensure that a column must always have a value i.e., it should not be left blank, then define a not null constraint on it. Null value can be inserted into the columns of any data type. Not null constraint can only be applied at column level.

Syntax:-

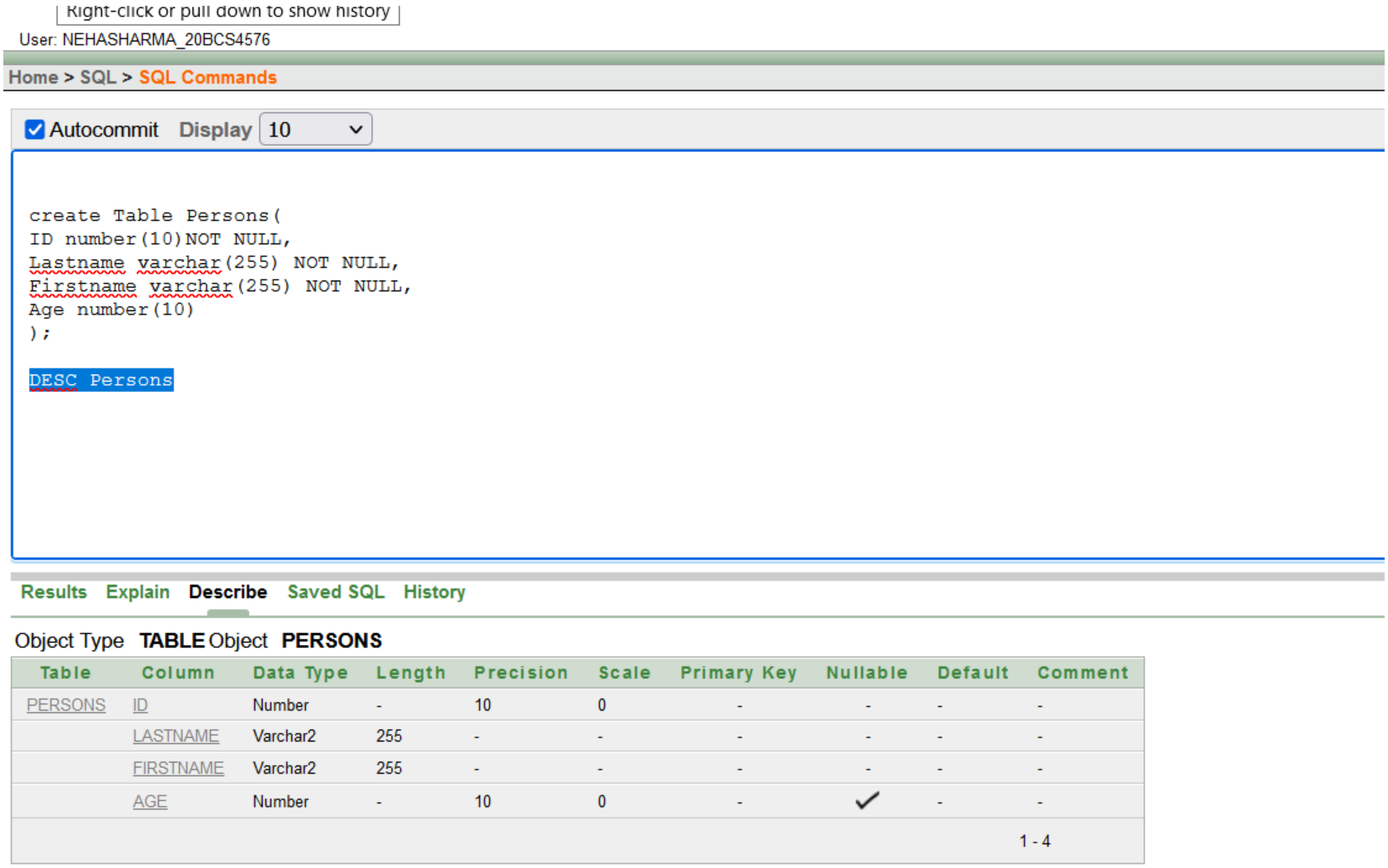
Column_name data_type(size) **NOTNULL**

Example:

create Table Persons(ID number(10)NOT NULL, Lastname varchar(255) NOT NULL, Firstname varchar(255) NOT NULL, Age number(10));

DESC Persons;

Output:-



Right-click or pull down to show history
Home > SQL > SQL Commands
Results
Explain
Describe
Saved SQL
History

User: NEHASHARMA_20BCS4576

☑ Autocommit Display 10

```
create Table Persons(
ID number(10)NOT NULL,
Lastname varchar(255) NOT NULL,
Firstname varchar(255) NOT NULL,
Age number(10)
);

DESC Persons
```

Object Type **TABLE** Object **PERSONS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PERSONS	ID	Number	-	10	0	-	-	-	-
	LASTNAME	Varchar2	255	-	-	-	-	-	-
	FIRSTNAME	Varchar2	255	-	-	-	-	-	-
	AGE	Number	-	10	0	-	✓	-	-

1 - 4

2. Adding Unique Constraint

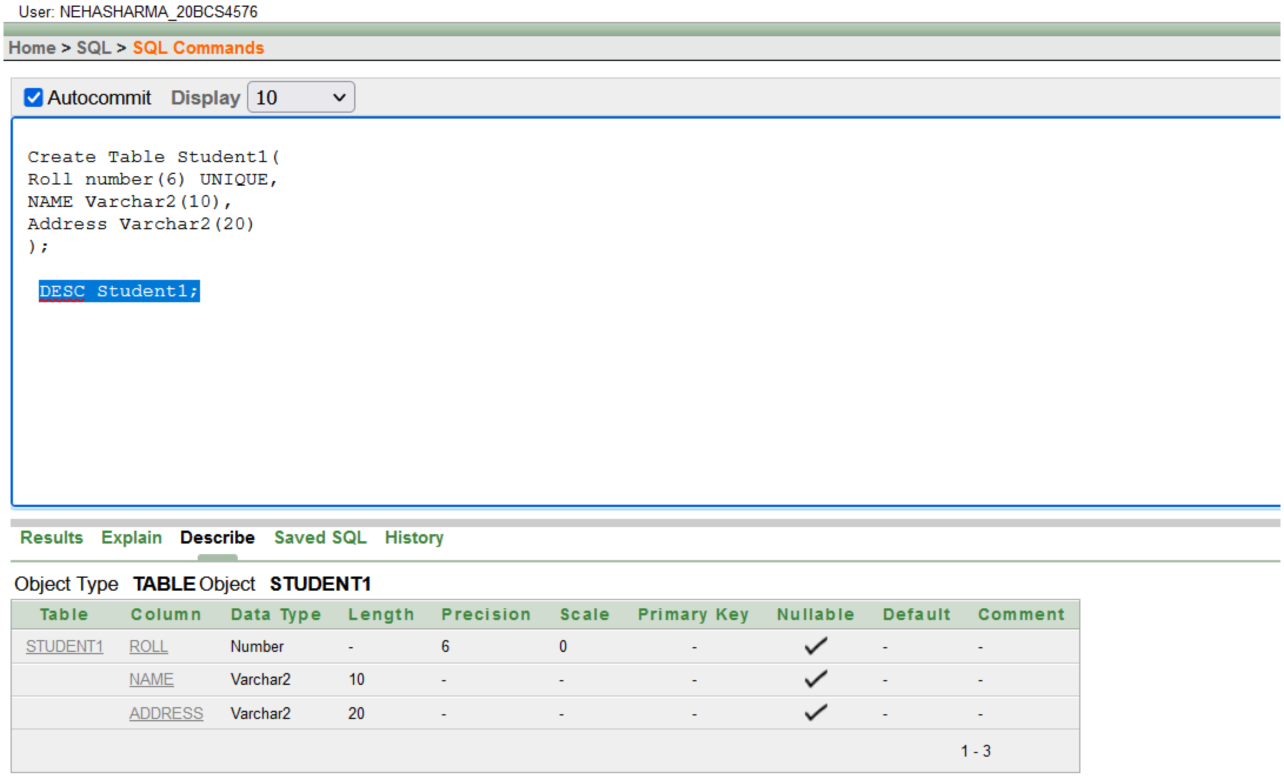
Command:- UNIQUE

Purpose:- The UNIQUE constraint ensures that all values in a column are different. you can have many UNIQUE constraints per table.

Syntax: - CREATE TABLE table_name (...column_name data_type UNIQUE ...);

Example:- Create Table Student1(Roll number(6) UNIQUE, NAME Varchar2(10), Address Varchar2(20));

DESC Student1;



User: NEHASHARMA_20BCS4576

Home > SQL > SQL Commands
☑ Autocommit Display 10

```
Create Table Student1(
Roll number(6) UNIQUE,
NAME Varchar2(10),
Address Varchar2(20)
);

 DESC Student1;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **STUDENT1**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
STUDENT1	ROLL	Number	-	6	0	-	✓	-	-
	NAME	Varchar2	10	-	-	-	✓	-	-
	ADDRESS	Varchar2	20	-	-	-	✓	-	-

1 - 3

3. Adding a Primary key

Command- Primary Key

Purpose: - A primary key column in a table has special attributes: It defines the column as a mandatory column i.e., the column cannot be left blank.

The NOT NULL attribute is active. The data held across the column must be UNIQUE.

Syntax: - Defined at column level: Column name datatype(size) Primary Key Defined at table level

Primary Key (column name1, column name2,….., column name n)

Example: -

Defined at column level:

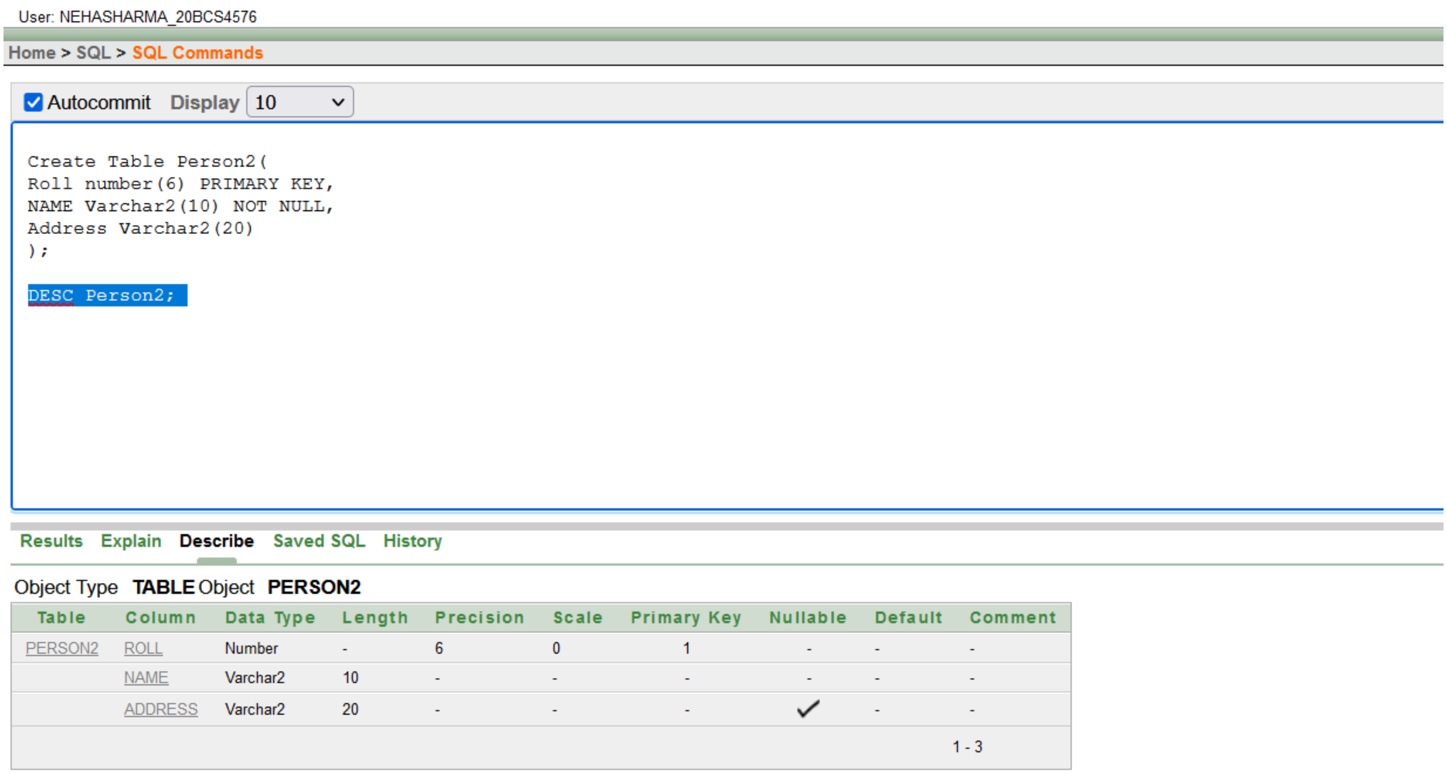
```
Create Table Person2(Roll number(6) PRIMARY KEY,NAME Varchar2(10) NOT NULL, Address Varchar2(20));

DESC Person2;
```

Defined at table level:

```
create table stu101(ID number(10),rollno number(10) , Name varchar2(20), Branch varchar2(5), Primary Key(ID, rollno));

insert into stu101 values(1,101,'A','CSE');

insert into stu101 values(1,101,'B','CSE');

insert into stu101 values(,102,'C','CSE');
```

Output

Defined at column level



User: NEHASHARMA_20BCS4576

Home > SQL > SQL Commands

☑ Autocommit Display 10

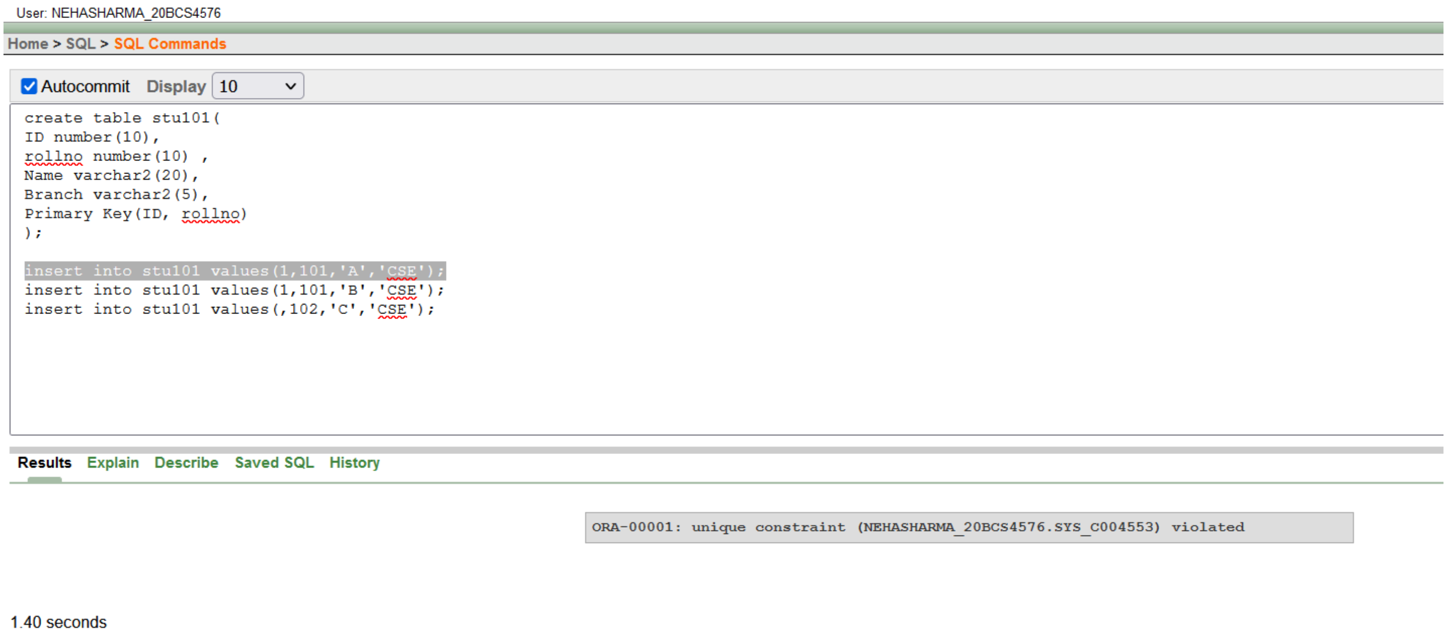
```
Create Table Person2(
Roll number(6) PRIMARY KEY,
NAME Varchar2(10) NOT NULL,
Address Varchar2(20)
);

DESC Person2;
```

Object Type TABLE Object PERSON2

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PERSON2	ROLL	Number	-	6	0	1	-	-	-
	NAME	Varchar2	10	-	-	-	-	-	-
	ADDRESS	Varchar2	20	-	-	-	✓	-	-

1 - 3

Defined at table level:



User: NEHASHARMA_20BCS4576

Home > SQL > SQL Commands
☑ Autocommit Display 10
Results Explain Describe Saved SQL History

```
create table stu101(
ID number(10),
rollno number(10) ,
Name varchar2(20),
Branch varchar2(5),
Primary Key(ID, rollno)
);

insert into stu101 values(1,101,'A','CSE');
insert into stu101 values(1,101,'B','CSE');
insert into stu101 values(,102,'C','CSE');
```

```
ORA-00001: unique constraint (NEHASHARMA_20BCS4576.SYS_C004553) violated
```

1.40 seconds

4. Adding a Foreign Key-:

Command: - Foreign Key

Purpose: - A foreign key is a column or group of columns whose values are derived from the primary key or unique key of some other table. If the name of the column is not specified, by default oracle uses primary key in the master table.

Syntax: - Defined at column level:

Column name datatype(size) References table name [(column name)] [On Delete Cascade]

Defined at table level: Foreign Key(column name[, column name]) References table name [(column name [,column name])

Example: -

Defined at column level:

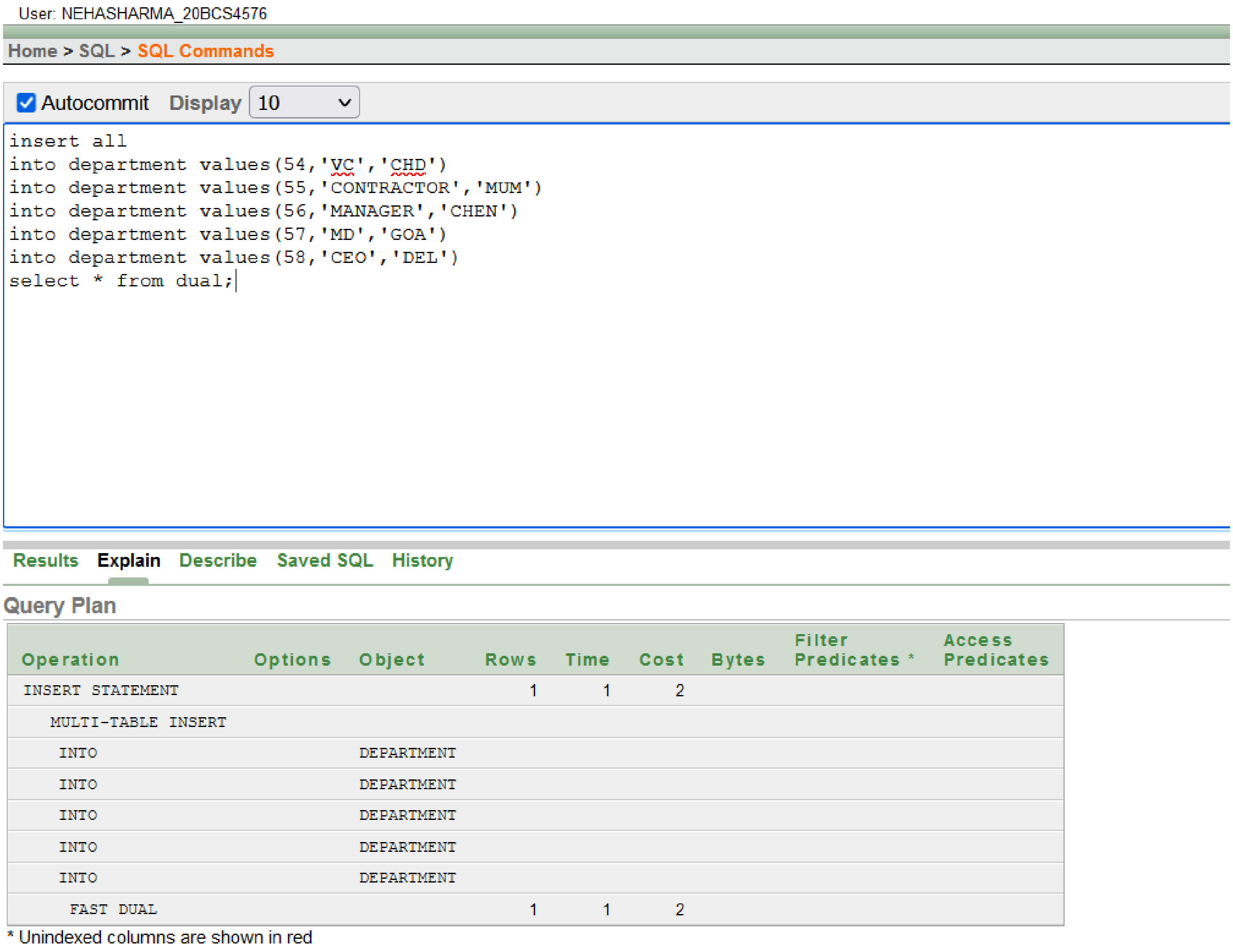
```
 Create Table DEPARTMENT ( Deptno Number(2) Primary key, Dname Varchar2 (15) Not Null,
Dlocate Varchar2(20) );
```

```
create table employees( EMPNO1 int Primary Key, ENAME Varchar2(20) Not Null, JOB
Varchar2(20), MGR Number(4), HIREDATE Date, SAL number(7), COMM number(10),

DEPTNO number(2),
REFERENCES DEPARTMENT(DEPTNO)
);
```

```
insert all

into department values(54,'VC','CHD')
into department values(55,'CONTRACTOR','MUM')
into department values(56,'MANAGER','CHEN')
into department values(57,'MD','GOA')
into department values(58,'CEO','DEL')
select * from dual;
```

```
INSERT all
into employees values(108,'NIMS','manager',45,sysdate+10,60000,10000,58)
into employees values(107,'RIYA','jk',45,'01-jun-19',10000,500,57)
into employees values(106,'SAM','guard',45,'02-jun-21',10000,500,56)
into employees values(105,'DHRUV','gr',45,'04-jun-02',120000,5000,55)
into employees values(104,'KIMI','rep',45,'01-jun-09',10000,500,54)
select * from dual;
```

Output:-



User: NEHASHARMA_20BCS4576

☑ Autocommit Display 10

```
insert all
into department values(54,'VC','CHD')
into department values(55,'CONTRACTOR','MUM')
into department values(56,'MANAGER','CHEN')
into department values(57,'MD','GOA')
into department values(58,'CEO','DEL')
select * from dual;
```

Query Plan

Operation	Options	Object	Rows	Time	Cost	Bytes	Filter Predicates *	Access Predicates
INSERT STATEMENT			1	1	2			
MULTI-TABLE INSERT								
INTO		DEPARTMENT						
INTO		DEPARTMENT						
INTO		DEPARTMENT						
INTO		DEPARTMENT						
INTO		DEPARTMENT						
FAST DUAL			1	1	2			

* Unindexed columns are shown in red

Defined at table level:

Create Table DEPARTMENT (

Deptno Number(2) Primary key,

Dname Varchar2 (15) Not Null,

```
Dlocate Varchar2(20)
);

create table employees(
EMPNO1 int Primary Key,
ENAME Varchar2(20) Not Null,
JOB Varchar2(20),
MGR Number(4),

HIREDATE Date,
SAL number(7),
COMM number(10),
DEPTNO number(2),
foreign key(DEPTNO) REFERENCES DEPARTMENT(DEPTNO)

 );



 insert all
into department values(54,'VC','CHD')
into department values(55,'CONTRACTOR','MUM' ')
into department values(56,'MANAGER','CHEN')
into department values(57,'MD','GOA')
into department values(58,'CEO','DEL')
select * from dual;

INSERT all

into employees values(108,'NIMS','manager',45,sysdate+10,60000,10000,58)
into employees values(107,'RIYA','jk',45,'01-jun-19',10000,500,57)
into employees values(106,'SAM','guard',45,'02-jun-21',10000,500,56)
into employees values(105,'DHRUV','gr',45,'04-jun-02',120000,5000,55)
into employees values(104,'KIMI','rep',45,'01-jun-09',10000,500,54)
select * from dual;
```

Output: Both level of definition of foreign key has the same output

5. JOINS-:

Command: - INNER JOIN

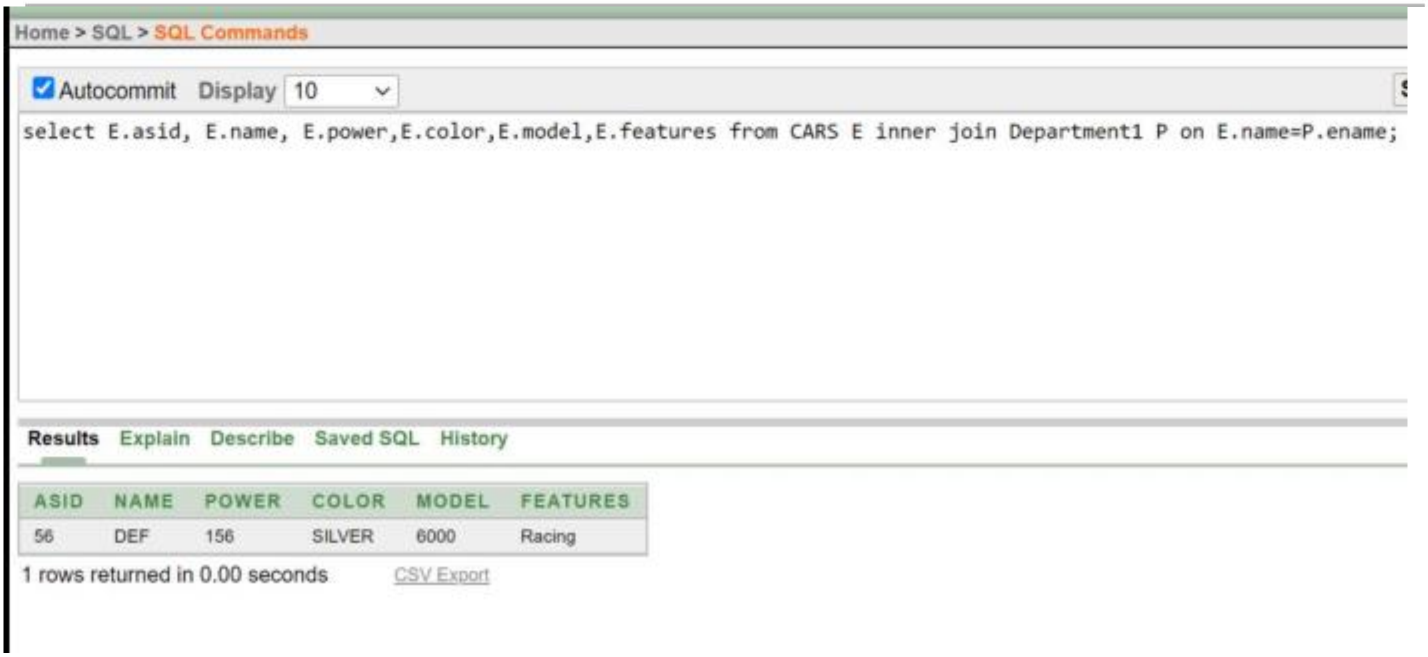
Purpose: - Inner joins are also known as Equi join. Sometimes it is necessary to work with multiple tables as though they were a single entity. Then a single SQL sentence can manipulate data from all the tables. Joins are used to achieve this.

select <column name 1>,….,< column name N> from <Table name1> **Inner join <**table name2> O**N** <Table name1>**.**<column name> = <Table name2>**.**<column name> **Where** <condition> **Order By** column name 1>,….,< column name N>;

Example:-

select E.eid, E.ename, P.Pid, P.Pname, P.Proj_duration from INSTRUCTOR E **inner join** Project101 P on E.Eid=P.Eid;

Output:-


Home > SQL > SQL Commands
☑ Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
select E.asid, E.name, E.power,E.color,E.model,E.features from CARS E inner join Department1 P on E.name=P.ename;
```

ASID	NAME	POWER	COLOR	MODEL	FEATURES
56	DEF	156	SILVER	6000	Racing

1 rows returned in 0.00 seconds

6. Create SIMPLE index:

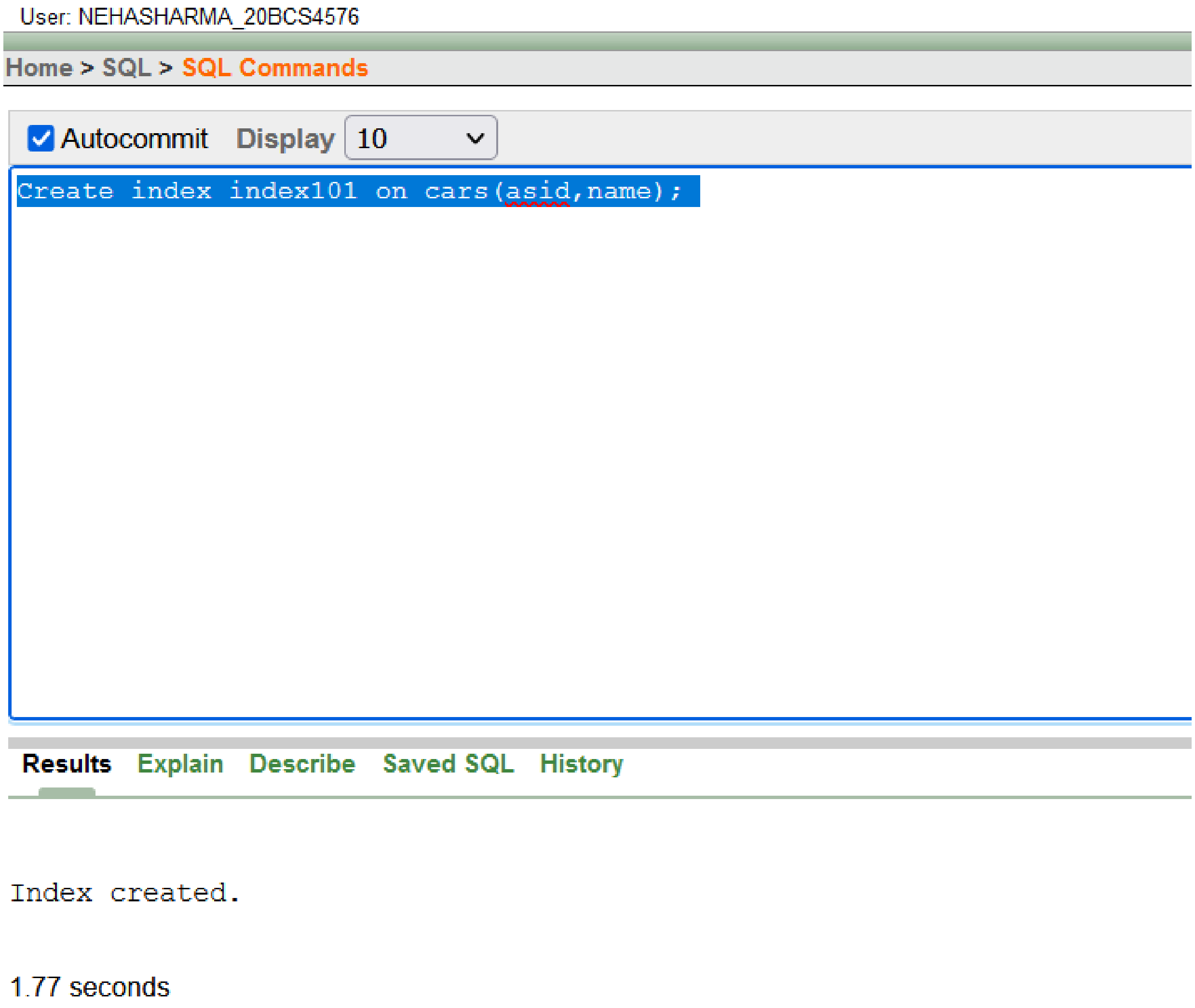
Command: - create index

Purpose: - Indexing a table is an access strategy, i.e., a way to sort and search records in the table. Indexes are essential to improve the speed with which record(s) can be located and retrieved from a table.

Syntax: - Create Index < index name > On
();

Example: create index index101 on CARS(asid,name);

Output:-



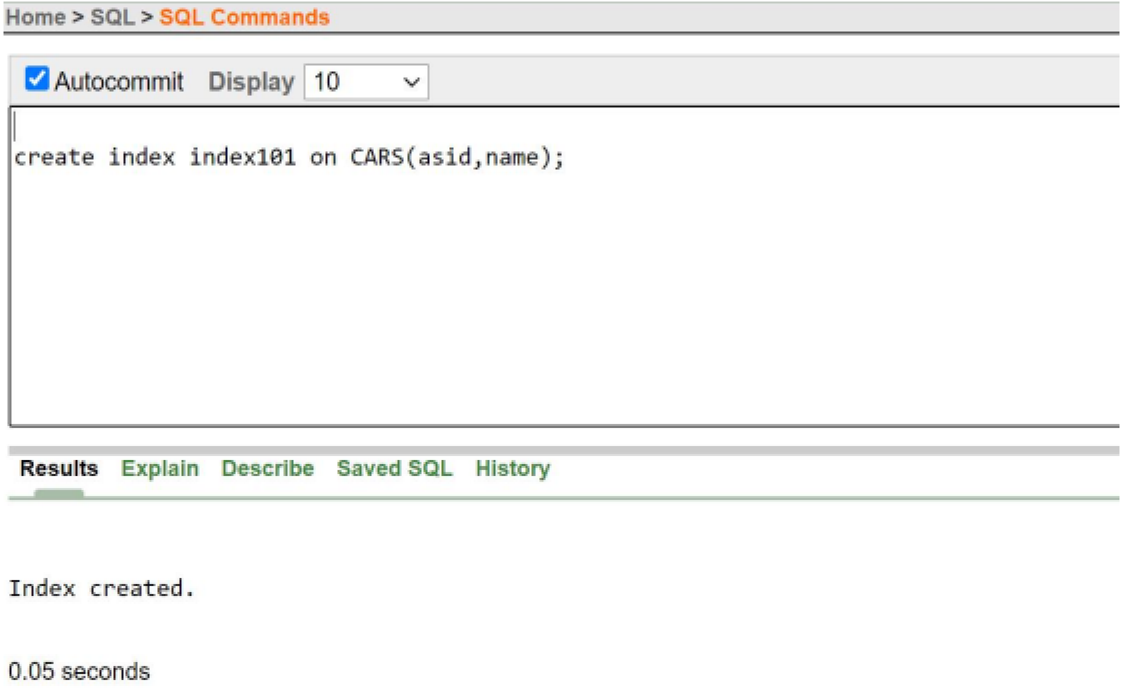
```
Create index index101 on cars(asid,name);
```

```
Index created.
```

1.77 seconds

7. **Create UNIQUE index:**
   Command: - create UNIQUE index
   Purpose: - Creates a unique index on a table. Duplicate values are not allowed.

   Syntax: - Create Unique Index < index name > On
   ();
   Example:
   create UNIQUE index index104 on department1(eaddress,age);



```
create index index101 on CARS(asid,name);
```

```
Index created.

0.05 seconds
```

8. Remove Index From INSTRUCTOR table:

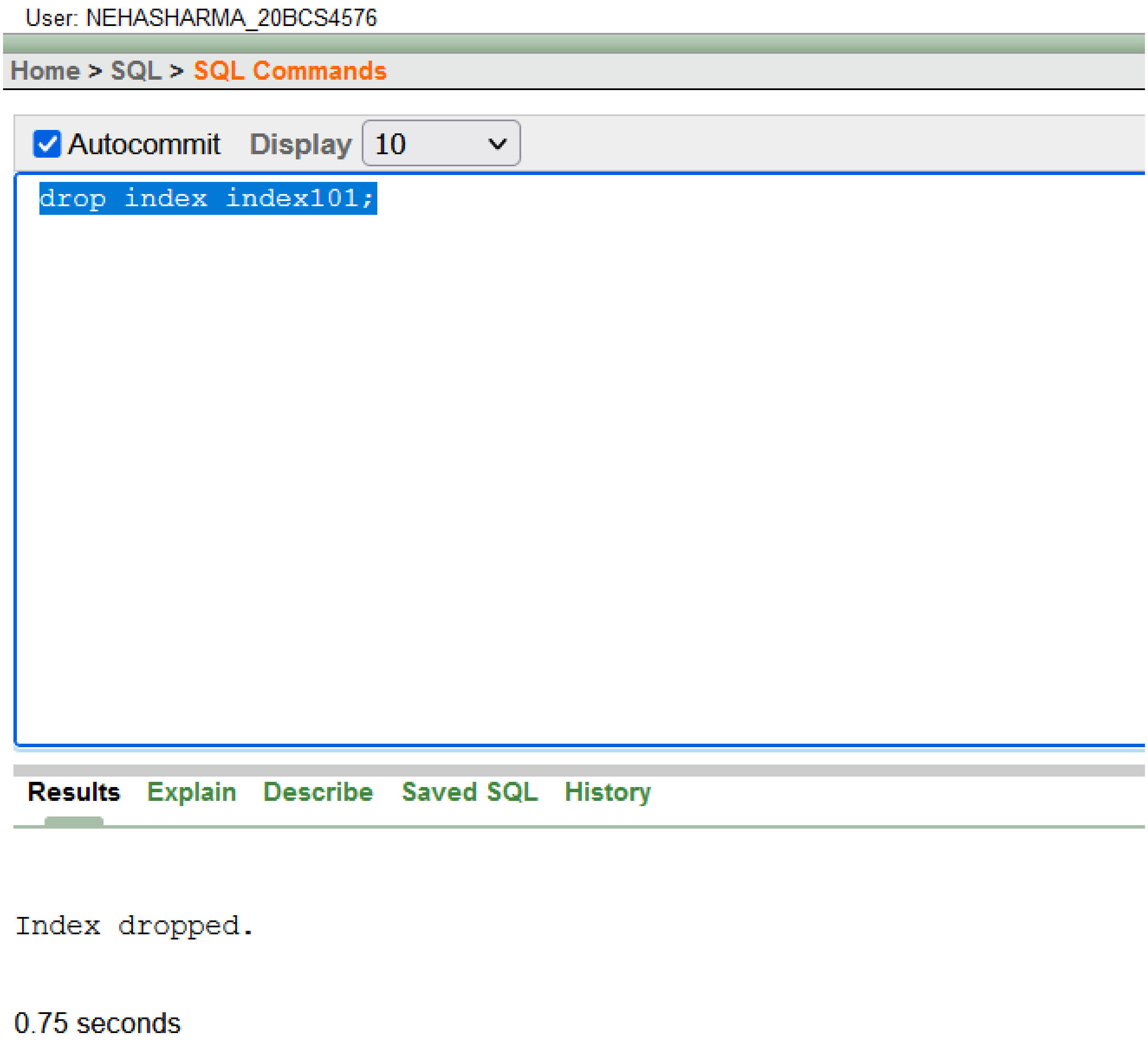
Command: - Drop Index

Purpose: - To drop all the indexes create don INSTRUCTOR table.

Syntax: - Drop index ;

Example: drop index index101;

Output:



```
drop index index101;
```

```
Index dropped.

0.75 seconds
```

9. <u>Create a view:</u>

Command: - Create view

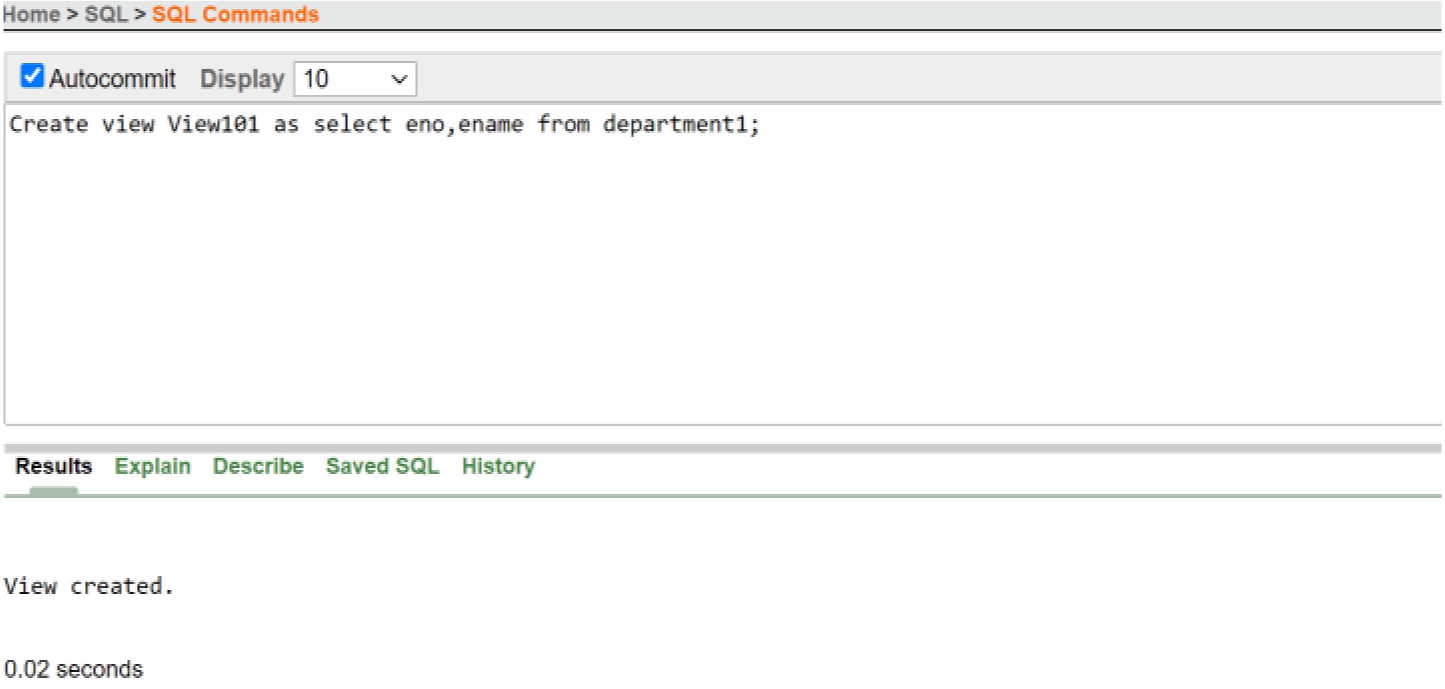
Purpose: - If a user wants to see only relevant

data then he/she can use views.

Syntax: - Create view AS Select ;

Example: Create view View101 as select eno,ename from department1;

Output:-



```
Create view View101 as select eno,ename from department1;
```

```
View created.

0.02 seconds
```

10. Creating a sequence

Command:- create sequence

Purpose:- Oracle provides an object called a sequence that can generate numeric values. A sequence can be defined to generate numbers in ascending or descending order, provide intervals bteween numbers, caching of sequence of numbers in memory to sped up their availability.

Syntax:-

CREATE SEQUENCE <sequence name>

[INCREMENT BY <integer value>

START WITH <integer value>

MAXVALUE <integer value>/NOMAXVALUE

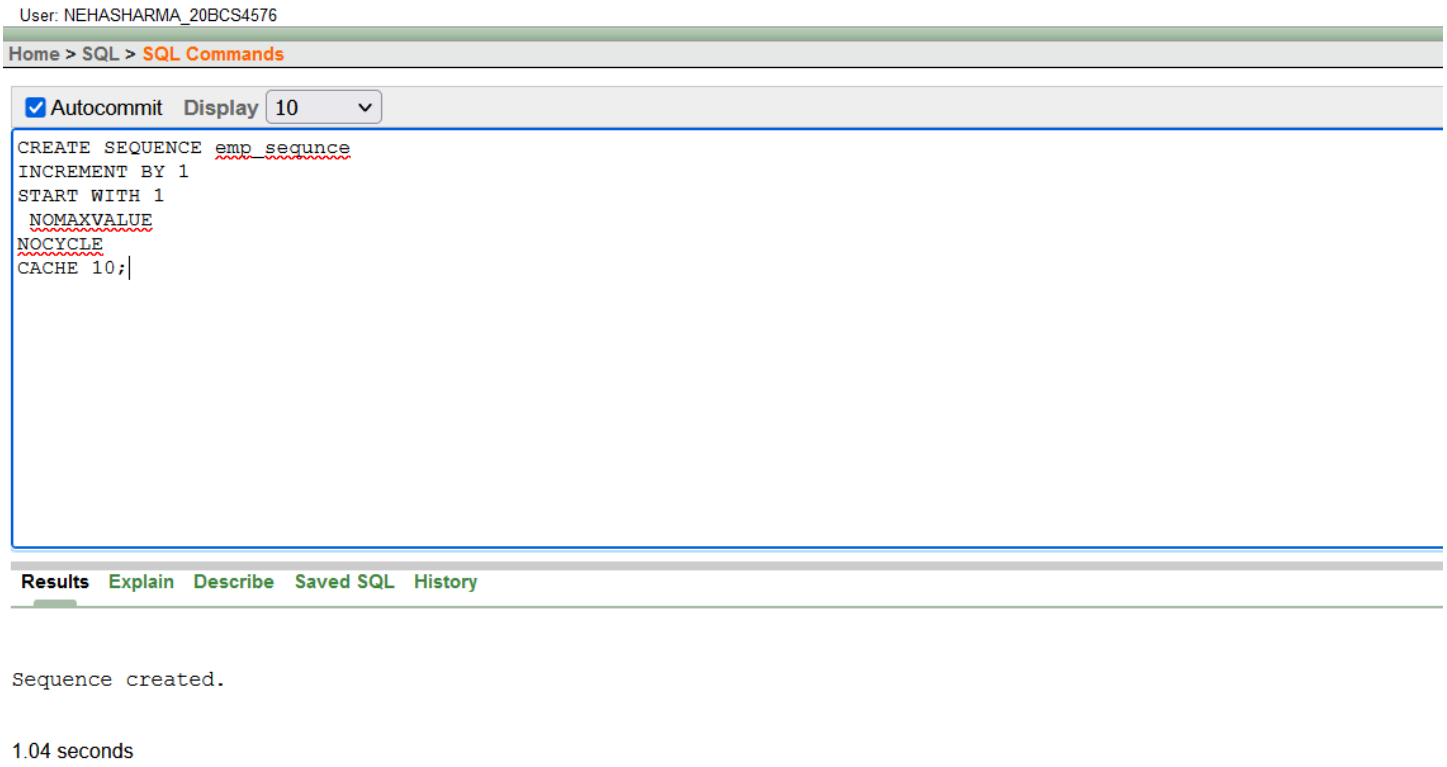
MINVALUE <integer value>/NOMINVALUE

CYCLE / NOCYCLE CACHE<integer value> / NOCACHE ORDER/ NOORDER];

Example:- create sequence seq101 increment by 1 maxvalue 20 nocycle order start with 1;

```
CREATE SEQUENCE emp_sequence

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOCYCLE

CACHE 10;
```

Output:-



☑ Autocommit Display 10

```
CREATE SEQUENCE emp_sequnce
INCREMENT BY 1
START WITH 1
 NOMAXVALUE
NOCYCLE
CACHE 10;
```

```
Sequence created.
```

1.04 seconds

4. Observations/Discussions:- We observed the different syntax of implementing constraints on created table in SQL.

5. Result/Output/Writing Summary:- After completing this experiment we get to know constraints types in SQL and its implementation on table like JOINS, SEQUENCE, VIEW CREATION and FOREIGN KEY.

6. Graphs (If Any): Image/Soft copy of graph paper to be attached here:- NA

Learning outcomes (What I have learnt):

1. **We learnt different syntax of implementing constraints on created table in SQL.**
2. **After completing this experiment we get to know constraints types in SQL and its implementation on table like JOINS, SEQUENCE, VIEW CREATION and FOREIGN KEY.**

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			