Experiment - I

DDL Commands – Create, Alter, Drop

Aim:- To create, alter, drop, Rename, Truncate, using data definition language (DDL) statements.

Description:- Data definition language (DDL) statements are used to define the database structure or schema.

DDL commands:- Create, Alter, Drop, Rename, Truncate

Questions:-

1) Create a table name student with following structure.

Create table student (Reg No. int name varchar(15), Gender char(1), DOB Date, Mobile No int, city varchar (15));

Dear students:-

Field	Type	Null	Key	Default	Extra
Reg No	int (3)	Yes			
Name	char (15)	Yes			
Gender	char(1)	Yes			
DOB	int (10)	Yes			
Mobile No	int (10)	Yes			
City	char (10)	Yes.			

2) Create a table name Faculty with following structures.

Create table faculty (Fac-No varchar(4);
Fac_name varchar(15);
Gender char(1);
DOB date;
DOJ date,
Mobile_No int(10));

Desc Faculty:-

Field	Type	Null	Key	Default	Extra
Fac No	char(3)	Yes		Null	
Fac name	char(15)	Yes		Null	
Gender	char(1)	Yes		Null	
DOB	int(10)	Yes		Null	
DOJ	int(10)	Yes		Null	
Mobile-No	int(10)	Yes		Null.	

3) Create a table name department with the following structure.

Create table department (dept No varchar(4),
dept name varchar(15),
depart head varchar(4)),

desc department;

Field	Type	Null	Key	default	Extra.
dept No	int(10)	Yes		Null	
dept name	char(10)	Yes		Null	
dept head	char(10)	Yes		Null	

4) Create a table course with following structure.

4) Create table course (course No varchar(3),
course desc varchar(15),
course type char(1),
sem No char(1),
Hall No varchar(4),
Fac No varchar(4));

desc course;

Field	Type	Null	Key	Default	Extra.
Course No	varchar(3)	Yes		Null	
Course desc	varchar(15)	Yes		Null	
course type	char(1)	Yes		Null	
sem No	char(4)	Yes		Null	
Hall No	char(4)	Yes		Null	
Fac No	char(4)	Yes		Null	

5) Modify the faculty by adding a column name dept of data type varchar(10).

Alter table faculty,
Add column dept varchar(10),

desc faculty:-

Field	Type	Null	Key	Default	Extra
Fac No	int (3)	No	PRI	Null	
Fac name	char (15)	Yes		Null	
Gender	char (1)	Yes		Null	
DOB	int (10)	Yes		Null	
Mobile No	int (10)	Yes		Null	
DOJ	int (10)	Yes		Null	
dept	char (10)	Yes		Null.	

Result:-

Tables are created, altered and modified using DDL commands.

Experiment - 2

DDL commands with constriants - Primary foreign key · Unique check.

Aim:- To add the constraints like primary key unique key and check using DDL commands.

1) Alter the table students with following structure

Alter table student add primary key (Reg No);

desc student :-

Field	Type	Null	key	Default	Extra.
Reg No	int	No	PRI	Null	
S name	varchar(15)	Yes		Null	
Age	char(2)	Yes		Null	
Mobile No	int	Yes		Null	
address	varchar(15)	Yes		Null.	

2) After the table faculty with the following structure.

Alter table faculty add primary key (FacNo),

Alter table faculty add check Gender = 'M' or 'F'

desc Faculty.

Field	Type	Null	Key	Default	Extra
Fac NO	int(3)	No	PRI	Null	
Fac name	char(15)	Yes		Null	
Gender	char(1)	yes		Null	
DOB	int(10)	yes		Null	
Mobile no	int(10)	Yes		Null	
DOJ.	int(10)	Yes.		Null.	

3) Alter the tables name department with the following structure.

Alter table department add primary key (dept No)

Field	Type	Null	Key	Default	Extra.
dept No	int(10)	No	PRI	Null	
dept name	char(10)	yes		Null	
dept head	char(10)	Yes		Null.	

4) Alter the table name course with following structure.

Alter table course add primary key (courseno)

Alter table course add check (semno >= 1 and semno = 6).

desc course;

Field	Type	Null	Key	Default	Extra
course No	int(3)	No	PRI	Null	
course desc	char(15)	Yes		Null	
course type	char(1)	Yes		Null	
sem no	int(10)	Yes		Null	
Hall no	int(10)	Yes		Null	
Fac no	int(10)	Yes		Null.	

5) Alter the table name student with following structure.

Alter table student add unique key (s name);

desc student:.

Field	Type	Null	Key	Default	Extra.
Reg no	int	No	PRI	Null	
S name	varchar(15)	Yes	UNI	Null	
Age	char(2)	Yes		Null	
Mobile no	int	Yes		Null	
Address	varchar(15)	Yes		Null	

6) After the student1 & student2 table are successfully created, test if you can add constriants Foreign key to the Reg no of this table.

select * from department

Defin	Deptname	Depthead
1	wind	vay
2	war	eir
3	looper	lut
4	fappa	fayal

Result: Data Manipulation Language (DML) commands such as INSERT, SELECT are performed in the five tables.

Experiment - 4

DML Commands with Constraints - UPDATE, DELETE

Aim:- To perform DML Commands such as UPDATE, DELETE in the table.

Description:

UPDATE - update existing data within a table.

DELETE - deletes all records from a tables, the space for the records remain.

Update:-

```
update table_name
  set column1 = value1, column2 = value2,
  where some_column = some_value;
```

Delete:-

```
delete from table_name
where some_colum = some_value;
```

Questions:-

1) Update the value of student name register number as '191711342'.

```
update student set_name = 'ram', where regno =
  '191711342';
```

Regno	name	Gender	DOB	Mobile no	city
191711342	Ram	M	2005-05-12	9908269?3	chennai
191347293	Mohan	M	2007-10-05	9898970	mumbai

2) Delete the record in the table Faculty, who resigned her job?

```
Delete from faculty where resigned = 'n';
  select * from faculty;
```

Facno	Fac name	Gender	DOB	DOJ	mobileno	deptno
1923SD4108	Sanjay	M	2005-01-12	2014-12-20	9926275b	Mtr
1923232??	saheel	M	2004-01-12	2015-06-14	9966139	Mtr

3) Modify the age the faculty where name is 'Sanjay' with the value '59'.

update table set name age = 59 where name = 'mohan'.

select * from faculty;

Fac No	Fac name	Gender	DOB	DOJ	Mobileno	dept	regr	age
192S2488	Sanjay	M	2005-03-04	2016-12-20	93902	MNr	Y	45
19192S168	saheel	M	1992-05-17	2016-12-30	9390373(	Mtr	Y	68

4) Remove all faculty who are having over 65 years.

Delect from faculty where age = 65;

select * from Faculty ;

Facno	Facname	Gender	DOB	DOS	Mobno	dept	regr	age
19192S24	sanjay	M	2005-03-20	2004-06-15	993031	CSE	Y	45
1923243?	saheel	M	2004-01-12	2014-05-03	ESE	Y	59	

Result: Data manipulation language (DML) commonds such as UPDATE, DELETE are performed in the live tables.

Experiment-5

select with various clause - WHERE, pattern matching

Aim:- To view the records from the tables using select commands with where clause & pattern matching.

Description:- The select statement allows you to get the data from tables. A tables consits of rows & colums like a spreadsheet. Often, you want to see a subset rows, a subset of columns, or a combination of bo.

select:-

select column-1, column 2,..
From table-1
[Inner | left | Right] Join table-2 on conditions where conditions.
Group by column-1
Having group-conditions
orded by column-1
Limit offset, length;

1) The student counselor wanted to display the registration number, student name and date of birth for all the students.

select reg.no, name, DOB from student

Reg.no	name	DOB
19221151	Sanjay	2004-01-12
19152148	saheel	2005-02-13

Reg No	Name	Gender	DOB	Mobile No	city
192126108	Sivi	F	2000-01-12	8919243776	chittor.

3) List the students who registered for the course "cool".

select * from student where course = "cool";

Regno.	name	Gender	DOB	Mobile	city	course
192526108	sanjay	M	2000-12-13	9876012	Tirupathi	cool
192222125	Rosy	F	2000-08-22	9324761	Nellore	cool

4) Display all faculty details joined before November '2014'.

select * from faculty where DOB2 '2014-11-01',

Fac no	Facname	Gender	DOB	DOJ	Mobile	dept	age
197157	sanjay	M	2000-01-12	2018-05-13	9357	MTV	54

5) Display all the courses not allotted to hall 'Hool'

Select * from course where hall no = 'Hool'

Course no	Computer desc	Course type	sem no	Hall no	Fac no.
CS201	Computer science	L	2	Hool	F001

6) List the student whose name ends with the substring "sh"

Select * from student where name like "sh"

Reg no	name	Gender	DOB	mobile no	city
192214726	Rowesh	M	2005-03-14	9827316411	MMR

7) Display all the students where name contain the substring "sh"

select * from student where name like "ramesh";

RegNo	name	Gender	DoB	mobileNo	city
19221153	Ramesh	M	2005-03-14	9957621492	MNR

8) Find all the students who are located in cities having "sal" as substring.

select * from student where name like "%sal%";

RegNo	Name	Gender	DoB	mobileNo	city	courseNo
19221156	DUMA	M	2004-12-15	9854328526	salt	601

9) Display the students whose name do not contain six letters.

RegNo	name	Gender	DoB	mobileNo	city	courseNo
19221567	Puma	M	2004-12-15	9876541129	salt	601

10) find all the students whose name contains "am".

select * from student where name like "%am%";

RegNo	name	Gender	DoB	mobileNo	city
19221153	ramesh	M	2005-03-04	9957614318	MNR

Result:

The records from the table are displayed uses SELECT commands with WHERE clause and pattern matching.

Experiment-6

Select with various clause - Between, in, Aggregate function.

Aim:- To view the records from the table using select commands with between, in, aggregate functions.

Between operator:

```
select column1, column2,...
from table_name
where expre [NOT] between begin-expr and end-expr;
```

In operator:-

```
select column1, column2,
from table-name
where (expre (column-1) in ('value1', 'value2' ...)
```

1) List the type of the course "statistic" and "Programming"

A) select courtype from course where coursedesc in ('computer science 101', Mathematic 201');

courseType
mandatory
elective

2) The instructor wants to known the courseNo whose scores are in the range 50 to 80.

A) select courseNo from studentscore where score between 50 & 80;

CourseNO
c001
c001
c002
c002

3) Find the average mark of "c002".

A) select avg(score) from studentscores where

CourseNo = 'C002';

avg(score)

75.000

2) List the max, mini mark for "C002".

A) select max(score), min(score) from studentscores where courseNo = 'C002';

Max(score)	min(score)
85	65

3) List the name of the courses & average mark of each course.

A) select max(score), min(score), avg(score) from student scores group by coursename;

max(score)	min(score)	avg(score)
90	70	80.0000
85	65	75.0000

5) Calculate the sum of all the scores.

select sum(score) from student scores;

sum(score)

465

6) How many student are registered for each course. Display the course description and the no. of students registered in each course.

select coursedesc, count(studentNo) from studentscores group by coursedesc;

coursedesc	count(studentNo)
cse	3
mat	3

Result: The records from the table are displayed using (select) commands with (where) clause and pattern matching

Experiment - 7

select with various clause - Group by, Having, order by.

Aim: To view the records from the tables using select commands with Group by, having, order by

Syntax:

Group By-Having

select c1, c2 ..., cn, aggregate-function (ci)
from table
where where_conditions
Group By c1, c2, ... cn
having conditions

Order by:

select column1, column2, ..
from tbl
order by column [ASC [DESC], column2 [ASC [DESC]--]

1) How many students are registered for each course. Display the course description & the no. of ?

select coursedesc, count (studentNo) from studentscores group by coursedesc;

coursedesc	count (studentNo);
cse	3
mat	3

2) How many courses did each student register.

A) select studentNO, count (coursename) from studentscores group by studentNO;

studentNo	count (coursname)
s001	2
s002	2
s003	2

D) Retrieve Name, Gender, Mobile No of all student in ascending order of Reg No.

A) select name, gender, Mobile_no from student order by Reg_no;

Name	Gender	Mobile_No
mohan	m	9000932613
raju	m	9886589 4
ramesh	m	9982635248
san	m	9982692 45
siri	f	8919385 34

Result:- The records from the tables are displag using select commands with Group By, Having and order by.

Experiment - 8

Query with sub query & correlated Query

Aim: To perform subquery and correlated on the given relations.

description: A mysql subquery is a query need within another such as select, insert, update, delect in addition, a mysql subquery can be nested inside another subquery.

Syntax:-

subquery:

```
select c1, c2, ----, cn
from table
where c1 in (select c1, c2, ---cn
             from table
             where where_condition);
```

correlated query:-

```
select *
from table_name
whereexists (subquery);
```

1) which of the student's score is greater than the average score?

A) select * from stud where marks > (select avg (marks) from stud);

regno	name	courseno	marks	facultyid	assement
1234	nani	c001	50	2001	5
1235	ramu	c001	50	2002	4

2) which of the student's have written more than one assessment test?

A) select name from stud where assessmk > 1;

name
nani
ramu
ravi

b) which faculty has joined recently and when?

A) select * from faculty order by doj limit 1;

facno	facultyname	gender	DOB	DOJ	mobileno	degree
f01	sanjay	M	0000-00-00	000-00-00	2147483647	CSE

b) List the course & course score of assessments that have the value more than average score each course.

A) select courseno, marks from stud where marks>(select avg(marks) from stud) order by courseno;

courseno	marks
C001	50
C001	50

Result: The records from the tables are displayed using sub-query over correlated sub-query.

Experiment - 9

Query with Joins - EquiJoin, Innerjoin, outerjoin

Aim: To perform join using equi join, inner join, outer join on the given relation.

description: Mysql supports the following types of joins

1) Cross join
2) Inner join
3) Left join
4) Right join

Syntax:

Cross Join:-

```
select t1.id, t2.id
from t1 cross join t2;
```

Inner join:-

```
select t1.id, t2.id
from t1 inner join t2 on
t1.pattern = t2.pattern
```

Left join:-

```
select t1.id, t2.id
from t1 left join on t1.pattern = t2.pattern
order by t1.id;
```

Right join

```
select t1.id, t2.id
from t1 Right join t2 on.
t1.pattern = t2.pattern
order by t2.id;
```

Questions:-

1) List the departments where the faculty members are working.

A) select faculty.facno, faculty.facname, department.deptno, department.deptname from faculty cross join department;

facno	facname	deptno	deptname
802	sanjay	11	sales

3) Find the student who has no score in any of the courses. List student name and course number.

A) select student.name, student.marks, course.courseno from student inner join course on student.name = course.courseno;

name	marks	courseno
sanjay	0	C00
rame	0	C00
Geetha	0	C00
Pooja	0	C00

3) The office clerk needs the names of the course taken by the faculty belonging to 'sales' whose name is "sanjay".

A) select faculty.facno, faculty.facname, department.deptno, department.deptname from faculty cross join department;

facno	facname	deptno	deptname
802	sanjay	11	sales

Result:- The records from the tables are displayed using join using equijoin, innerjoin, outerjoin.

Experiment-10

Database with view and Index

Aim: To create view and index on the given relation.

Description: Mysql has supported database views since version 5+. In Mysql almost features of views conform to the sql.

Syntax - view:

```
create [Algorithm = {merge | temptable | undefined}]
view [database_name].[view_name]
as [select statement]
```

Index:-

```
create [unique | fulltext | spatial] Index index_name
using [Btree | hash | Rtree]
on table_name (column_name [(length)] [ASC | DESC],...)
```

1) Create a view with name 'V1' using employees tables which holds the values of employee_id and salary of employee.

A)

employee_id	first_name	last_name	device_serial	salary
1	John	Smith	ABC123	60000
2	Jane	Doe	DEF456	65000
3	Bob	Johnson	GHI789	70000
4	Sally	Fields	JKL012	75000
5	Michael	switch	MNO345	80000
6	Emily	william	PQR678	85000

desc employee1;

Field	Type	Null	Key	Default	Extra
employee_id	int	Yes	Mul	Null	
first_name	varchar(50)	Yes		Null	
last_name	varchar(50)	Yes		Null	
device_serial	varchar(15)	Yes		Null	
salary	int	Yes		Null	

create view v1 as select employee_id, salary from employee1;

Field	Type	Null	Key	Default	Extra
employeeid	int	Yes		Null	
salary	int	Yes		Null	

Index:

1) Create index i for 'salary' athibute from employee relation & list the first name of the employee who salary is above 145000 & explain the working principle of indexing and drop the table.

A) select first_name from employee81 where salary > 145000;

first_name
sanjay
John
Jane.

drop index index1 on employee1;

Results: The records from the tables are displayed using view and index on the given relation.

Experiment - 11

Database with Auto-increment sequence

Aim: To create Auto increment sequence on the given relation.

Syntax:

Create table table_name (
col-name1 Auto_increment primar key, col-name2, col-names...);

Questions:

1) Populate register number using auto increment in DBMS_stud
2) Manually populate register number
3) Drop the auto increment

A) Create table DBMS_stud (
Regno key (Reg-no),
name varchar(30) Not null,
department varchar(30) Not null,
Mark int(30) Not null);

Field	Type	Null	Key	Default	Extra
Regno	int	No	Pri	Null	auto-increment
Name	varchar(30)	No		Null	
Department	varchar(30)	No		Null	
Mark	int	No		Null	

Insert into DBMS_stud (name, department, marks) values.
('Raj', 'CSE', 89), ('Ramesh', 'ECE', 87),
('Sanjay', 'CSE', 88), ('Rajan', 'ECE', 90);

select * from DBMS_student;

Regno	Name	Department	Mark
1	Raj	CSE	89
2	Sanjay	CSE	88
3	Ramesh	ECE	87
4	Rajan	ECE	90

Result: The records from the tables are displayed using auto_invenst sequence on the given relation.

Experiment - 12

Simple Programming using Repeat, while

Aim: To learn how to use various MySql loop statement including while, repeat to run a block of code repeatedly based as condition.

while loop syntax:-

while expression
Do statements
end while,

Procedure: 1) The while loop checks the expression at the beginning of each iteration.
2) If the expression values to true, mysql will execute between while & end while until the expression false.

Programm 1:- write a function that uses while statements to build a string repeatedly until the select statement.

Programm 2:- write a function that uses Repeat statement which would repeat the loop until income is greater than (or) equal to 4000, at which Repeat loop.

Program - 1:-

```
create Procedure test_mysql_while_loop()
    Begin
    declare x int;
    declare str varchare(255);
    set x = 1;
    set str = '';
     while x <= 5 DO
    set str = concat(str, x, ',');
    set x = x + 1;
    end whilie;
    select str;
    end
```

call test_mysql_while_loop()

str

1, 2, 3, 4, 5

Program-2:-

Create procedure dorepeat (p, int) begin set @x=0; Repeat

set @x = @x+1;

select @x;

@x

4002

Result:- Thus the simple programming exercise using (repeat, while) executed successfully.

Experiment- 13

Aim :- To learn how to use various mysql loop statements case & loop to run a block.

Case syntax:-

```
case case_value
when when_value then statement_list
[when when_value then statement_list]
[else statement_list]
  end case
```

Program 1:- write a function that use CASE statement where if monthly_value is equal to or less than 4000, then income_level will be set to 'Low income'.

Program 2:- write a function that will use Iterate statements which would cause the loop to repeat while income is less than 4000. Once income is greater than or equal to 4000, would terminate the loop.

1) create function incomlevel (monthly_value int)

```
Returns varchar (20)
begin
  declare income_level varchar(20);
case monthy_value
  when 4000 then
    set income_level = 'low income';
  when 5000 then
    set income_level = 'Avg income';
  else set income_level = 'high income';
end case;
  return income_level;
end; //
select incomelevel(5300);
```

incomelevel(5300)
high income

6) create function calcincome2 (starting_value int)
returns int

```
begin
  declare income int;
  set income = 0;
  label1:-loop:
  set loop= income + starting_value;
  if income < 4000 then;
     iterate label;
  end if;
     leave label1:
  end loop, label1:
  return income;
end;
```

calcincome2 (2100)
4200

Result:- Thus the simple programming exercise using CASE & LOOP executed successfully.

Experiment - 14

TCL commands – Commit, savepoint, Rollback

Aim: To learn how to use various TCL commands commit, savepoint and Rollback SQL commands

Syntax:

Commit:

Commit;

RollBack:

Rollback;

Rollback to savepoint_name;

Problem 1:

Create a following table class & insert values into it in the following order & create savepoints in bits them. Try to rollback the save point & check your output by giving select commands

Update class set name = 'bravo' where id = 5';

savepoint A;

insert into class values ('uppal', 6);

savepoint B;

insert into class values ('balu', 7);

savepoint C;

A) create table class (name varchar(10), id int(5));

insert into class values ("df", 5);

> commit;

update class sqt name = "bravo" where id = "5";

savepoint A;

insert into class value ("uppal", 6);

savepoint B;

insert into class values("baku", 7);

savepoint C;

select * from class;

name	id
dj	5
uppal	6
baku	7

→ Rollback to B;

select * from class;

name	id
dj	5
uppal	6

Result:- The commands sql, commit, Rollback, & savepoint are implemented & result in verified.

Experiment - 15

DCL Commands - Grant, Revoke

Aim:-

To learn how to use various DCL commands such as GRANT and REVOKE

Syntax:-

Data control language (DCL) is used to control privileges in database.

System:

This includes permissions for creating session, table, etc. and all types of other system privileges.

object:

This includes permissions for any command or query to perform any operation on the database tables.

In DCL we have two Commands,

GRANT : used to provide any user access privileges or other privileges for the database.

Revoke: used to take back permissions from any user.

-> Allow a user to create session.

-> when we create a user in sql, it is not even allowed to login & create a session until & unless proper permissions/privileges are granted to the user.

-> following command can be used to grant the session creating privileges

```
GRANT CREATE SESSION TO username;
```

Allow a user to create table

To allow a user to create tables in the database we can use the below command,

GRANT CREATE TABLE TO username;

provides user with space on tablespace to store table

Allowing a user to create table is not enough to store Storing data in that table. we also must provide the use with privileges to use the available tablespace for their table & data.

ALTER USER username QUOTA UNLIMITED ON SYSTEM;

The above Command will alter the user details & will provide it access to unlimited tablespace on System

NOTE: Generally unlimited quota is provided to Admin user Grant all privilege to a user

Result:-

Thus the DCL Commands GRANT & REVOKE SQL executed successfully.