MySQL commands :- Part-II

 # select, from and where clauses

```
select name
from instructor;

select dept_name
from instructor;

select distinct dept_name
from instructor;

select all dept_name
from instructor;

select ID, name, dept_name
from instructor;

# use of select, from and where clause
select ID, name, salary
from instructor
where dept_name = ’ETC’;
```

 # Show ID, name, dept_name of all instructors with salary increased by 1000

```
select ID, name, dept_name, salary + 1000
from instructor;
```

 # Find the names of all instructors in the ETC department having salary greater than 100000

```
select name
from instructor
where dept_name='ETC' and salary > 100000;
```

 # Find the names of all instructors in the Computer department having salary greater than or equal to 950000

```
select name
from instructor
where dept_name='Computer' and salary >= 95000;
```

 # Find the names of all instructors who either belong to Mechanical department or having salary greater than 102000

```
select name
from instructor
where dept_name='Mechanical' or salary >= 102000;
```

 # create the table department

```
create table department (dept_name varchar (15), building varchar (15) not null, budget numeric (10,2), primary key (dept_name));
show tables;
desc department;
```

 # Inserting records in the table department

```
insert into department values ('Computer', 'Building No. 1', 1000000);
select * from department;
insert into department values ('IT', 'Building No. 2', 500000);
insert into department values ('ETC', 'Building No. 1', 800000);
insert into department values ('Mechanical', 'Building No. 3', 900000);
insert into department values ('Electrical', 'Building No. 2', 600000);
```

 # Retrieve the names of all instructors, along with their department names and department building....(using instructor and department tables)

```
select name, instructor.dept_name, building
from instructor, department
where instructor.dept_name = department.dept_name;
```

 # create table instructor_r with less number of records

```
create table instructor_r (ID int, name varchar(25) not null, dept_name varchar(15)
not null, salary numeric (10,3), primary key (ID));
show tables;
desc instructor_r;
```

 # Inserting three records in instructor_r

```
insert into instructor_r values (100, 'George', 'Computer', 110000);
insert into instructor_r values (101, 'Krishna', 'Computer', 95000);
insert into instructor_r values (102, 'Uday', 'IT', 105000);
select * from instructor_r;
```

 # create table department_r with less number of records

```
create table department_r (dept_name varchar (15), building varchar (15) not null,
budget numeric (10,2), primary key (dept_name));
```

 # Inserting three records in department_r

```
insert into department_r values ('Computer', 'Building No. 1', 1000000);
insert into department_r values ('IT', 'Building No. 2', 500000);
insert into department_r values ('ETC', 'Building No. 1', 800000);

select * from department_r;
```

 # Cartesian Product

```
select * from instructor_r, department_r;
select * from instructor_r cross join department_r;
select * from instructor, department;
select * from instructor cross join department;
```

 # Retrieve the names of all instructors, along with their salary and department building....(using instructor_r and department_r tables)

```
select name, salary, building
from instructor_r, department_r
where instructor_r.dept_name = department_r.dept_name;
```

```
# Try using  *
select *
from instructor_r, department_r
where instructor_r.dept_name = department_r.dept_name;
```

 # Natural Join and Equi join

```
select name, salary, budget
from instructor_r, department_r
where instructor_r.dept_name = department_r.dept_name;   # Equi join
```

```
select name, salary, budget
from instructor_r natural join department_r;    # Natural join
```

 # Natural Join and Equi join :- Using select *

```
select *          # here 7 columns will be displayed
from instructor_r, department_r
where instructor_r.dept_name = department_r.dept_name;
```

```
select *          # here 6 columns will be displayed
from instructor_r
natural join department_r;
```

 # To rename attribute using as clause

```
select dept_name from department_r;
select dept_name as dname from department_r;    # to rename using as clause
```

 # Changing one attribute name, but keeping others as same

```
select dept_name, building, budget from department_r;
select dept_name as dname, building, budget from department_r;
```

 # One of the reasons to use rename operation is - to replace a long relation name with a short version that is more convenient to use.

```
select name, instructor_r.dept_name, building
from instructor_r, department_r
where instructor_r.dept_name = department_r.dept_name;

select name, I.dept_name, building
from instructor_r as I, department_r as D
where I.dept_name = D.dept_name;
```

 # String operations - Pattern matching

 # Find out names of instructor whose department names start from Com

```
select name from instructor_r
where dept_name like 'Com%';    # Use of like operator

select name, dept_name from instructor_r
where dept_name like "com%";
```

 # Find out names of instructor whose department names do not start from Com

```
select name, dept_name from instructor_r
where dept_name not like "com%";     # Use of not like operator
```

 # Attribute specification in select clause - use of *

```
select * from instructor_r;
```

 # Display all attributes from one table only, but operating on two tables

```
select instructor_r.*
from instructor_r, department_r
where instructor_r.dept_name = department_r.dept_name
and department_r.building = 'Building No. 1';
```

 # use of * in following query

```
select *
from instructor_r, department_r
where instructor_r.dept_name = department_r.dept_name
and department_r.building = 'Building No. 1';
```

 # upper and lower functions

```
select upper('k.p.adhiya');
select lower('COET');
```

 # trim function

 # TRIM([{BOTH | LEADING | TRAILING} [remstr] FROM] str)

```
select trim('Computer        ');
select trim('    Computer    ');
select trim(leading 'my'from 'mytext');
select trim(trailing 'text'from 'mytext');
```

 # order by clause

```
select * from instructor_r order by salary;
select * from instructor_r order by salary asc;
select * from instructor_r order by salary desc;
```

 # oreder by clause - If some instructors have same salary, then we can sort the table in descending order by salary and in ascending order by name

```
select * from instructor_r order by salary desc, name asc;
```

 # between operator

```
select name, dept_name
from instructor_r
where salary <=110000 and salary >=105000;

select name, dept_name
from instructor_r
where salary between 105000 and 110000;

select name, dept_name
from instructor_r
where salary not between 105000 and 110000;
```

 # Now create one more table:- instructor_nba.

```
create table instructor_nba (ID int, name varchar(25) not null, dept_name varchar(15) not null, salary numeric (10,3), primary key (ID));
show tables;
```

#Inserting records into instructor_nba

```
insert into instructor_nba values (101, 'Krishna', 'Computer', 95000);
insert into instructor_nba values (102, 'Uday', 'IT', 105000);
insert into instructor_nba values (103, 'Swapnil', 'ETC', 112000);
insert into instructor_nba values (104, 'Soumitra', 'Mechanical', 115000);
select * from instructor_nba;
```

 # union operation

```
select name from instructor_r
union
select name from instructor_nba;

select name from instructor_r
union all                    # to retain all duplicates
select name from instructor_nba;

select * from instructor_r
union
select name from instructor_nba;   # Error. See Error

select * from instructor_r
union
select * from instructor_nba;
```

 # The intersect operation.. But MySQL does not support intersect operator. So it can be done by join on

```
select instructor_r.name
from instructor_r
join instructor_nba
on instructor_r.name = instructor_nba.name;
```

```
select instructor_r.*
from instructor_r
join instructor_nba
on instructor_r.name = instructor_nba.name;

select *
from instructor_r
join instructor_nba
on instructor_r.name = instructor_nba.name;
```