What is database?

A collection of data and holds this data in the form of tables.

What is table?

Holds the data in form of rows and columns.

The database provides us the capability to access and manipulate this data.

There are two types of databases

1. Relational database – here the data store in the form of rows and columns and also the tables have relation between them. Example: MySQL, SQL server, PostgreSQL, SQLite, MariaDB, Oracle.
2. NoSQL database – here the data are not store in the form of rows and columns. And there is no relation between the tables. Examples: key values, document, graph; HBase, MongoDB, Cassandra.

SQL – Structured query language and used to query a relational database.

MySQL vs SQL:

MySQL is a software where SQL is a query language.

# Creating a database which name is Stamford

CREATE DATABASE Stamford;

#Showing database

SHOW DATABASES;

#Deleting a database

DROP DATABASE Stamford;

#If I have multiple databases inside my database then we need to say which database I want to use.

USE Stamford;

#How should I know which database I use now

SELECT DATABASE ();

#How I create a table called student in my database

CREATE TABLE student

(

Name varchar (50),

Id varchar (10),

Age int,

GPA double (3,2)

);

# If I want to see how many tables in my databases

SHOW TABLES;

# If I want to see the tables content

DESCRIBE student;

# How to drop table

DROP TABLE student;

Session2

CRUD operation

Create -> insert statements

Read -> select statements

Update -> update statements

Delete -> delete statements

#To show the content of the table

SELECT \* FROM student;

#Insert values to student table

INSERT INTO student VALUES

(‘Mamun’, ‘cse109’, 25, 3.95),

(‘Mahfuz’, ‘cse111’, 23, 3.90),

(‘Atik’, ‘cse107’, 22, 3.70);

#Create table using null properties

CREATE TABLE student

(

Name varchar (50) NOT NULL,

Id varchar (10) NOT NULL,

Age int NOT NULL,

GPA double (3,2) NOT NULL

);

#Create table using default value

CREATE TABLE student

(

Name varchar (50) NOT NULL,

Id varchar (10) NOT NULL,

Age int NOT NULL,

GPA double (3,2) NOT NULL,

Country varchar (20) DEFULT “Bangladesh”

);

INSERT INTO student (Name, Id, Age, GPA) VALUES (“Mamun”, “cse 109”, 20, 3.90);

#Change default value when need

INSERT INTO student (Name, Id, Age, GPA, Country) VALUES (“Atik”, “cse 107”, 20, 3.70, India);

# Setting null value when already set default value

INSERT INTO student (Name, Id, Age, GPA, Country) VALUES (“Mahfuz”, “cse 111”, 20, 3.80, null);

Session3

#primary key helps us to uniquely identify each record in the table

CREATE TABLE student

(

Name varchar (50) NOT NULL,

Id int (10) NOT NULL,

Age int NOT NULL,

GPA double (3,2) NOT NULL,

Country varchar (20) DEFAULT “Bangladesh”,

Primary key (Id)

);

#A table that contains primary key and auto increment key

CREATE TABLE student

(

Name varchar (50) NOT NULL,

Id int (10) AUTO\_INCREMENT,

Age int NOT NULL,

GPA double (3,2) NOT NULL,

Country varchar (20) DEFAULT “Bangladesh”,

Primary key (Id)

);

#composite primary key -> where two columns are use as primary key.

#Unique key -> unique key also does not hold duplicate just like primary key but the difference is it holds null values.

We have only one primary key but multiple unique in a table.

CREATE TABLE student

(

Id int AUTO\_INCREMENT,

Name varchar (50) NOT NULL,

Email varchar (30),

Age int NOT NULL,

GPA double (3,2) NOT NULL,

Country varchar (20) DEFAULT “Bangladesh”,

Primary key (Id), UNIQUE KEY(Email)

);

Session4:

Select:

Select all\_columns from table\_name;

SELECT \* FROM student;

Select some\_columns from table\_name;

SELECT Id, Name, Email FROM student;

How to filter data from the table

SELECT \* FROM student where age>20;

SELECT \* FROM student where Name = “atik”;

Normally SQL is case insensitive but if we use binary key word then It became case sensitive.

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* FROM student where [binary](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/cast-functions.html%23operator_binary) Name ='atik';

(This code works)

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* FROM student where [binary](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/cast-functions.html%23operator_binary) Name ='Atik';

(This code doesn’t work)

**#**Alias in columns

Select Name as st\_name, Id as st\_id from student;

#Update key word

Update student set Email = [mamunjsr@gmail.com](mailto:mamunjsr@gmail.com) where Name = ‘mamun’;

Update student age 3 years.

Update student set age = age+3;

#Delete key word

Delete from student where Id = 104;

Difference between update and alter:

Update is data manipulation on the other hand alter is structure manipulation.

Alter:

If I want to add column to our existing table

Alter table student add column university varchar (30);

If I want to delete any column from our existing table then

Alter table student drop column university;

If I want to modify a column type in our existing table then

Alter table student modify column Name varchar (30);

If I want to change the primary key then

Alter table student drop primary key;

Alter table student add primary key(Id);

Truncate:

It is a DDL command.

Truncate also removes all records. Truncate internally drops the table and recreates it.

Truncate table student;

DDL -> deals with table structure

Create, Alter, Drop

DML -> here we deal with the data directly

Insert, update, delete

Session5:

Now create another table:

Create table students

(

St\_id int auto\_increment,

St\_fname varchar (20) NOT NULL,

St\_lname varchar (20) NOT NULL,

St\_mname varchar (20),

St\_email varchar (20) NOT NULL,

St\_phone varchar (20) NOT NULL,

St\_department varchar (10) NOT NULL,

St\_enrollment\_date TIMESTAMP NOT NULL,

St\_location varchar (30),

Primary key (St\_id),

Unique key (St\_email),

);

Insert into students (St\_id, St\_fname, St\_mname, St\_lname, St\_email, St\_phone, St\_department, St\_location)

Values (101,'abdullah','Mamun', 'al', 'mamunsub@gmail.com', '01674386366', 'cse','jessore');

Insert into students (St\_fname, St\_mname, St\_lname, St\_email, St\_phone, St\_department, St\_location)

Values ('atikur','rahman', 'atik', 'atik@gmail.com', '01751851445', 'cse','pabna);

Insert into students (St\_fname, St\_mname, St\_lname, St\_email, St\_phone, St\_department, St\_location,selected\_course)

Values ('mahfujur','rahman', 'mahfuj', 'mahfuj@gmail.com', '01751851445', 'cse','Kishorgong',305);

#create another course table

Create table courses

(

Course\_id int not null,

Course\_name varchar (20) not null,

Course\_duration int not null,

Course\_fee int not null,

Primary key (course\_id)

);

Insert into courses (Course\_id, Course\_name, Course\_duration, Course\_fee) values(301, 'Java', 20, 8000);

Insert into courses (Course\_id, Course\_name, Course\_duration, Course\_fee) values(302, 'Python', 18, 10000);

# we want to add a column in the students

Alter table students add column selective\_course int not null;

#insert values in this field

Update students set selective\_course = 301 where St\_id =101;

Update students set selective\_course = 302 where St\_id =102;

#here we add foreign key constraint with the reference of courses table course\_id column so that if there is no match with the course\_id when fill the form so he/she does not complete the form.

Create table students

(

St\_id int auto\_increment,

St\_fname varchar (20) NOT NULL,

St\_lname varchar (20) NOT NULL,

St\_mname varchar (20),

St\_email varchar (20) NOT NULL,

St\_phone varchar (20) NOT NULL,

Selected\_course int NOT NULL DEFAULT 1,

St\_department varchar (10) NOT NULL,

St\_enrollment\_date TIMESTAMP NOT NULL,

St\_location varchar (30),

Primary key (St\_id),

Unique key (St\_email),

Foreign key (selected\_course) references courses (course\_id)

);

[Insert](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) into students (St\_fname, St\_mname, St\_lname, St\_email, St\_phone, St\_department, St\_location,selected\_course) [Values](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values) ('samia','rahman', 'sultana', 'samia@gmail.com', '01751851445', 'cse','Kishorgong',301), ('afiya','rahman', 'abdullah', 'afiya@gmail.com', '01751851445', 'cse','Dhaka',302), ('asif','rahman', 'asif', 'asif@gmail.com', '01751851445', 'cse','jessore',306);

Here courses table is parent table

And students table is the child table.

CHECK CONSTRAINT (is not supported in MySQL)

Constraints are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data is maintained. If there is any violation then the action is aborted.

Session6:

DISTINCT:

Select St\_location from students;

It returns all the location with duplicates.

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) distinct St\_location from students;

It returns location but no duplicates.

ORDER BY:

Order by used to sort the data ascending or descending order. Ascending is by default.

Select \* from students order by selected\_course;

Select \* from students order by selected\_course desc;

Select St\_fname from students order by St\_fname;

Without writing the column name after order by we should write by their column number that is select.

Select St\_fname, selected\_course from students order by 2 desc;

Without selecting the column we should use any column from the table along with the order by statement.

Select St\_fname from students order by selected\_course;

Select St\_fname, selected\_course from students order by St\_fname, selected\_course ;

LIMIT:

Limit is used to see the number of rows from the table.

Want see the last 3 students details from the database.

Select \* from students order by St\_enrollment\_date desc limit 3;

Select \* from students order by St\_enrollment\_date limit 0,1;

Select \* from students order by St\_enrollment\_date desc limit 0,1;

LIKE:

Like is used to find word like this.

% is a wild card character which means one or more.

\_ is a wild card character which means exactly one.

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_fname [LIKE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-comparison-functions.html%23operator_like) '%h';

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_fname [LIKE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-comparison-functions.html%23operator_like) '%\_k';

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_fname [LIKE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-comparison-functions.html%23operator_like) '%\\_jur';

**Order of execution:**

Select distinct St\_department, St\_location from students order by St\_location;

Here we want to apply distinct key word only in St\_department but it works both St\_department, St\_location column.

**Session8:**

Aggregate functions(count,MIN, MAX, sum)

Count():

How to find how many rows in the table?

Select count(\*) from students;

Find how many departments are in the table?

Select count(distinct St\_department) from students;

Select count(distinct St\_location) from students;

Group by:

Select St\_department from students group by St\_department;

Select St\_department, count(\*) from students group by St\_department;

Select St\_location, count(\*) from students group by St\_location;

Select St\_department, count(\*) from students group by St\_location;

Min and max:

Want a add a column name cgpa in the table students.

Alter table students add column cgpa double(3,2);

Add value to the table.

update students set cgpa=3.87 where St\_id=102;

Find minimum value of cgpa.

Select MIN(cgpa) from students;

Select MAX(cgpa) from students;

Select St\_fname, St\_mname, St\_lname,cgpa from students order by cgpa limit 1;

Select St\_fname, St\_mname, St\_lname,cgpa from students order by cgpa desc limit 1;

SUM:

Select sum(cgpa) from students;

AVG:

Select avg(cgpa) from students;

Using subquery:

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) St\_fname, St\_mname, St\_lname,cgpa from students where cgpa > ([Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) [avg](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html%23function_avg)(cgpa) from students);

Session9:

#If we to change our timestamp column when I change anything in our table, for this we need to add something in table creation.

Changed\_at timestamp default now() on update now();

Now() = current\_timestamp()

Session10

Logical operator:

#Get student from Jessore.

Select \* from students where St\_location = ‘jessore’;

# get student not from Jessore.

Select \* from students where St\_location != ‘jessore’;

Same as

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_location <> 'jessore';

#get all students who have the word Rahman in there mname

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_mname [like](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-comparison-functions.html%23operator_like) '%rahman%';

#get all students who do not have the word Rahman in there mname.

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_mname not [like](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-comparison-functions.html%23operator_like) '%rahman%';

And:

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_location = 'jessore' and cgpa>(select avg(cgpa) from students);

OR:

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_location = 'jessore' or cgpa>(select avg(cgpa) from students);

Between:

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where cgpa between 3.50 and 3.80;

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where cgpa not between 3.50 and 3.80;

IN:

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_location [in](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/comparison-operators.html%23function_in) ('jessore','pabna');

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* from students where St\_location not [in](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/comparison-operators.html%23function_in) ('jessore','pabna');

CASE:

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) St\_id,St\_fname,cgpa,

[Case](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/control-flow-functions.html%23operator_case) When cgpa > 3.8 then 'A+'

Else 'A'

End as Grade

From students;

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) St\_id,St\_fname,St\_location,

[Case](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/control-flow-functions.html%23operator_case) When St\_location [in](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/comparison-operators.html%23function_in) ('jessore','pabna') then 'Khulna'

Else 'Dhaka'

End as Division

From students;

Select fname, lname, age

Case

When age> 30 ‘old’

Else ‘Young’

From employee

Order by age;

Select fname, lname, jobtitle, salary,

Case

When jobtitle = ‘salesman’ then salary + ( salary \* .10)

When jobtitle = ‘accountant’ then salary + ( salary \* .05)

When jobtitle = ‘hr’ then salary + ( salary \* .15)

else salary + ( salary \* .05)

end as after\_increment

from employee join employee\_salary

on employee.id = employee\_salary.id

Session11:

Joins:

Join are four types:

1. Inner join
2. Left join
3. Right join
4. Full join

Now create table for join:

CREATE table student\_details

(

Roll int not null auto\_increment,

Name varchar (15) not null,

Gender varchar (10),

Age int,

Primary key (Roll)

);

Insert into student\_details

(Roll, Name, Gender, Age)

values

(101, ‘rahim’, ‘male’, 18),

(101, ‘karim’, ‘male’, 19),

(101, ‘nasim’, ‘male’, 20),

(101, ‘farin’, ‘female’, 17),

(101, ‘nasrin’, ‘female’, 18);

Create table exam\_result

(

Reg\_number int not null,

Roll int,

Gpa double (3,2),

Group\_name varchar (20),

Primary key(Reg\_number)

);

Insert into exam\_result

(Reg\_number, Roll, Gpa, Group\_name)

values

( 201, 102, 3.25, ‘science’),

(202, 101, 3.45, ‘arts’),

(203, 103, 4, ‘science’),

(204, 104, 3.87, ‘commerce’);

JOIN:

Select Roll, Reg\_number, Name, Gender, Group\_name, Gpa

From student\_details as sd

Join

Exam\_result as er

On

Sd.Roll = er.Roll

Inner Join:

(returns only the matching from the two table)

Select Roll, Reg\_number, Name, Gender, Group\_name, Gpa

From student\_details as sd

Inner Join

Exam\_result as er

On

Sd.Roll = er.Roll

Left join:

(Returns all from left table and the matching rows from the right table)

Select Roll, Reg\_number, Name, Gender, Group\_name, Gpa

From student\_details as sd

Left Join

Exam\_result as er

On

Sd.Roll = er.Roll

RIGHT JOIN:

(Returns all from the right table and matching rows from the left table)

Select Roll, Reg\_number, Name, Gender, Group\_name, Gpa

From student\_details as sd

Right Join

Exam\_result as er

On

Sd.Roll = er.Roll

FULL JOIN:

(Returns all rows from the both tables)

Select Roll, Reg\_number, Name, Gender, Group\_name, Gpa

From student\_details as sd

Full Join

Exam\_result as er

On

Sd.Roll = er.Roll

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) course\_name from courses where course\_id =( [Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) selected\_course from students where St\_fname='Abdullah');

Inner join:

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) St\_id, St\_fname, St\_mname, St\_lname, course\_name, St\_location From students join courses on Students.selected\_course= courses.course\_id;

[Select](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) St\_id, St\_fname, St\_mname, St\_lname, course\_name, St\_location From students left join courses on Students.selected\_course= courses.course\_id;

Union -> removes all duplicates

When we preform union operation then we must ensure that both tables selected columns are same type. Otherwise it doesn’t work.

Union all -> keep all data with duplicates

Session 12:

Difference between where and having:

Where is used before the aggregation of individual records to filter out but having is used after the aggregation of records to filter.

Select sourse\_of\_joining, count(\*) as total from students group by sourse\_of\_joining where total > 1; (this code is not work)

Select sourse\_of\_joining, count(\*) as total from students group by sourse\_of\_joining having total > 1;

Select sourse\_of\_joining, count(\*) as total from students group by sourse\_of\_joining where sourse\_of\_joining=’linkedin’; (this code is not work)

Select sourse\_of\_joining, count(\*) as total from students group by sourse\_of\_joining having sourse\_of\_joining=’linkedin’;

Select sourse\_of\_joining, count(\*) as total

from students

where sourse\_of\_joining=’linkedin’

group by sourse\_of\_joining; (more efficient)

Is it possible to use where and having in a same query?

Select location, count(\*) as total

from students

where years\_of\_exp > 10

group by location

having count > 1;

Where is used before group by and do filtering on individual records

Having is used after group by and do filtering on aggregated records.

Where is more performant than having

Session13:

Partition by:

Now create a table :

CREATE TABLE employee

(

Fname varchar (20),

Lname varchar (20),

Age int,

Salary int,

Location varchar (20)

);

INSERT INTO employee VALUES (‘sachin’, ‘sharma’, 28, 10000,’bangalore’);

INSERT INTO employee VALUES (‘shane’, ‘warne’, 30, 20000,’bangalore’);

INSERT INTO employee VALUES (‘rohit’, ‘sharma’, 32, 30000,’hyderabad’);

INSERT INTO employee VALUES (‘shikhar’, ‘dhawan’, 32, 25000,’hyderabad’);

INSERT INTO employee VALUES (‘rahul’, ‘dravid’, 31, 20000,’bangalore’);

INSERT INTO employee VALUES (‘saurabh’, ‘ganguly’, 32, 15000,’pune’);

INSERT INTO employee VALUES (‘kapil’, ‘dev’, 34, 10000,’pune’);

If we want to group this data by location and count the number and also want to find out their avg salary so…

Select fname ,lname, employee.location total\_count, avg\_salary from employee

Join

( select location, count (location) as total\_count, avg(salary) as avg\_salary from employee group by location) temptable

On

Employee.location = temptable.location;

We can use over() partition by;

Select fname, lname, location,

count(location) over (partition by location) as total,

Avg(salary) over (partition by location) as average

from employee;

session14:

row\_number():

select fname, lname, salary,

row\_number() over (order by salary desc) as rownumber from employee;

#find the 5th highest salary from the table;

Select \* from

(select fname, lname, salary, row\_number() over (order by salary desc) as rownumber from employee) temptable

where rownumber = 5;

#Assign row number for partition based on each location….

select fname, lname, location, salary, row\_number() over (partition by location order by salary desc) as rownum from employee;

#i want to find the highest salary getters at each location

Select \* from (select fname, lname, salary, row\_number() over (order by salary desc) as rownumber from employee) where rownumber = 1;

Session15:

Rank and dense rank:

Row\_number do not solve the problem of duplicates.

select fname, lname, salary,

RANK() over (order by salary desc) from employee;

select fname, lname, salary,

DENSE\_RANK() over (order by salary desc) from employee;

If there are no duplicates then row number, rank and dense rank lead to similar results…

Only difference comes when there are duplicates…

Rank – for duplicates same rank is assigned and for the next entry it skips the ranks…

Like: 1,2,3,3,5,6,6,8

Dense rank – it does not skip any rank in between

Like – 1,2,3,3,4,5,5,6

When we use partition by then after every partition the rank starts from 1.

In row\_number() partition by is optional but order by is mandatory.

Session16:

VIEW:

View is a key word to control the view of the table. If I have a table that have five columns if I want that my viewer just sees four columns from them so I can use this view key word create a new table for the viewer.

Select \* from student\_details;

Create view table:

Create view student\_view as

Select Roll, Name, Gender

From student\_details;

How to access this view table? In view table everything is like normal table.

Select \* from student\_view;

Date function:

Dayname(); (Returns name of the day)

Monthname(): (Returns name of the month)

Session17:

CTEs:

Common table expression:

Cte sometime it calls with clause

Fetch employees who earn more than average salary of all employees…..

Create table employee

(

Emp\_id int not null auto\_increment,

Emp\_name varchar (20),

Salary int,

Primary key (Emp\_id)

);

Insert into employee values

(101, ‘Mohan’, 40000),

(102,’James’, 50000),

(103, ‘Robin’, 60000),

(104, ‘Carol’, 70000),

(105, ‘Alice’, 80000),

(106, ‘jimmy’,90000);

With avg\_salary (avg\_sal) as

(select cast( avg(salary) as int) from employee)

Select \* from employee e, avg\_salary av

Where e.salary >av.avg\_sal;

Need to learn:

Union, intersect, except