MYSQL

To start the shell –

# Mysql -h (host) localhost -u (user) root

To show the database list –

# Show databases;

To create a database –

# Create database cse\_333(name of the database);

To use/enable a database –

# Use cse\_333;

To create a table in the database –

# Create table lab\_grades (std\_id char(4) (name datatype), name varchar(30), present int, marks double, cgpa decimal(3,2), submission\_date date );

To see table list –

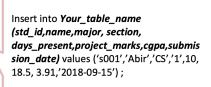
# Show tables;

To see the columns type –

# Describe lab\_grades;

*Insert*

To insert values in the certain columns –



To insert values in the column serially –



(can insert multiple row by just adding comma one by one)

To check the whole table –

# Select \* from lab\_grades;

To check certain columns –

# Select (column name 1) (column name 2) from lab\_grades;

*Editing*

To add a column –

# Alter table lab\_grades add project\_title(new column) varchar(10);

To modify a column type –

# Alter table lab\_grades modify column project\_title char(23);

To remove a column –

# Alter table lab\_grades drop column project\_title;

To update a certain value in a column –

# Update lab\_grades set project\_marks = 23 where std\_id = ‘s003’;

To update multiple values in a row –

# Update lab\_grades set project\_marks = 33, section = 3, major = ‘EEE’ where std\_id = ‘s003’;

To change a column name –

# Alter table users change influencer\_count Followers(newname) int(type);

*Delete*

To delete a row –

# Delete from lab\_grades where Name = ‘Roza’;

To delete multiple rows –

# Delete from lab\_grades where days\_present < 10 ;

To delete a table –

# Drop table lab\_grades;

To delete a database –

# Drop database cse\_333;

*Formula*

To create a formula to create a new column –

# Select name, (project\_marks + Days\_present\*5/12) from lab\_grades;

To create new formula and a new column with new name –

# Select name , (project\_ marks + Days\_present\*5/12) as total\_marks from lab\_grades;

# select name, ((Followers\*100/1000000)\*(multiplier\*100/20))/100 as Efficiency from users;

*Functions*

To convert the string into upper case –

# select upper(Name) from users;

To show the unique values –

# select distinct Major from users;

To sort –

# select \* from lab\_grades order by cgpa; (ascending order)

# select \* from lab\_grades order by cgpa desc; (descending order)

This helps to sort firstly by days present. And if it cant sort it by just the days present, then it takes help of CGPA.

# select \* from lab\_grades order by days\_present, CGPA

*Condition with where*

To Print rows with certain thing –

# select Name, Major, CGPA from lab\_grades where Major = ‘CSE’;

Between –

# select name,project\_marks from lab\_grades where project\_ marks between 16.5 and 20;

In –

# select name,project\_marks from lab\_grades where Major in (‘CS’, ‘CSE’);

And -

# select name, project\_marks, CGPA from lab\_grades where Major = ‘CSE’ AND days\_present > 10;

A field that start with a –

# Select Name, major from lab\_grades where Name like ‘a%’;

# Select Name, major from lab\_grades where Name like ‘a%a%’; ( a-something-a-something )

# Select Name, major from lab\_grades where Name like ‘a\_\_\_’; ( 3 underscore means 3 letters after a )

*Aggregate Functions*

Max –

# select MAX (CGPA) from lab\_grades;

# select MAX (CGPA) as high\_cgpa from lab\_grades;

Min –

# select MAX (CGPA) from lab\_grades;

# select MAX (CGPA) as high\_cgpa from lab\_grades;

Average –

# select AVG (CGPA) from lab\_grades;

# select AVG (CGPA) as high\_cgpa from lab\_grades;

SUM –

# select SUM (CGPA) from lab\_grades;

# select SUM (CGPA) as high\_cgpa from lab\_grades;

Max and Min can be used in the strings to detect the string with number of letters –

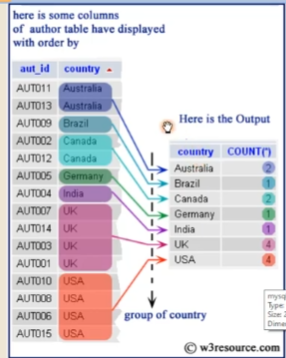
# select MIN (Name) from lab\_grades;

Max and Min can be used in the dates to detect the near and far date –

# select MIN/MAX (subsmission\_date) from lab\_grades;

*Grouping Records*

Create a subtable –



Maximum cgpa for each department –

# select major, Max(cgpa) from lab\_grades group by major;

Count rows –

# select count(\*) from lab\_grades;

Count number of student in each group –

# select major, count (\*) from lab\_grades group by major;

Students present less than 10 days I each group –

# select major, count (\*) from lab\_grades where days\_present < 10 group by major;

The group that has a number of students greater than 2 –

# select major, count (\*) from lab\_grades group by major having count(\*) >2 ; (after group by we can not use where)

*Nested Query*

To find the max cgpa and print the name that got the max cgpa –

# select name from lab\_grades where cgpa = (select max(cgpa) from lab\_grades);

To find the name from each department with max cgpa –

# select name,major,cgpa where (major, cgpa) in (select major, max(cgpa) from lab\_grades group by major);

To find the students of CS department who’s cgpa is greater than any of the students of CSE department -

#select name, cgpa from lab\_grades where major = ‘CS’ and cgpa > any (select cgpa from lab\_grades where major = ‘CSE’)

To find the students of ECE department who’s cgpa is greater than all of the students of CSE department -

#select name, cgpa from lab\_grades where major = ‘ECE’ and cgpa > all (select cgpa from lab\_grades where major = ‘CSE’)

*Correlated Sub-Query*

#

select distinct Major

from lab\_grades L1

where exists

( select \*

from lab\_grades L2

where L2.Major=L1.Major AND L2.CGPA<3.7);

#

Select Name

From lab\_grades L1

Where not exists

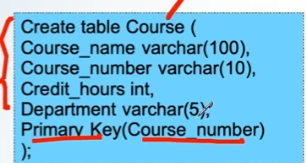
(select \*

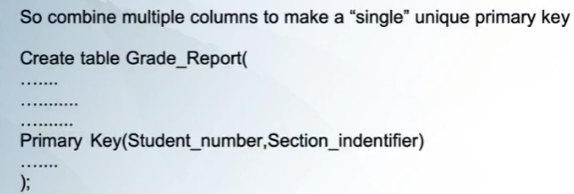
from lab\_grades L2

where L2.std\_id!=L1.std\_id and L2.project\_marks>L1.Project\_marks);

*Primary key*

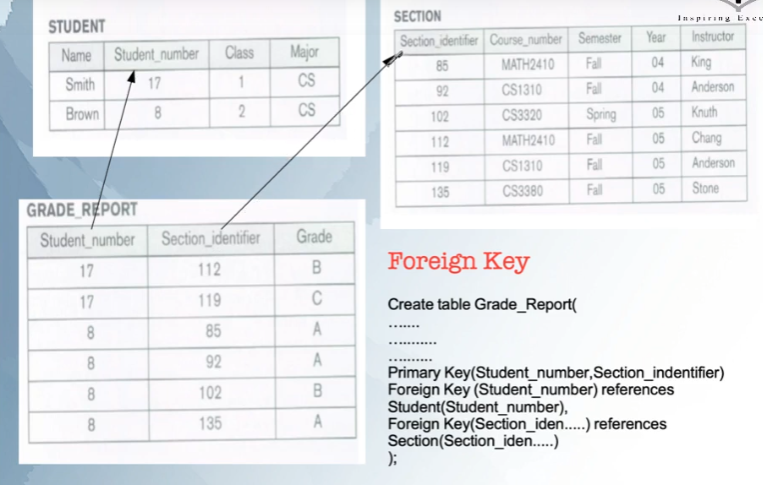
There have to be only one primary key in a table. But there can be multiple attributes.





*Foreign key*

Connecting multiple columns of different tables. There can be multiple foreign keys

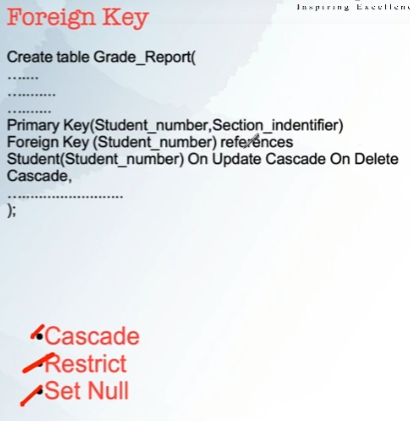


If we delete or update a table, 3 operations happens –

> Cascade : deletes all the related rows and informations

> Restrict : By default if nothing is set then it stays restricted. Restrict don’t let anything to delete

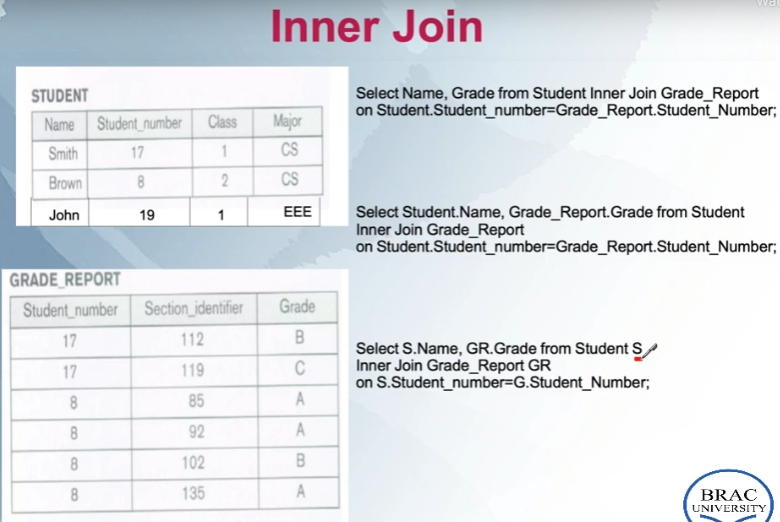
> Set Null : If any info is deleted, then that info in other table becomes null



*Joins*

4 types of joins :

Inner join, Left join, Right join, Full outer join(not supported in mysql)



Left join : takes all values from the left table but only the common values from the right table.

Right join : takes all values from the Reft table but only the common values from the left table.