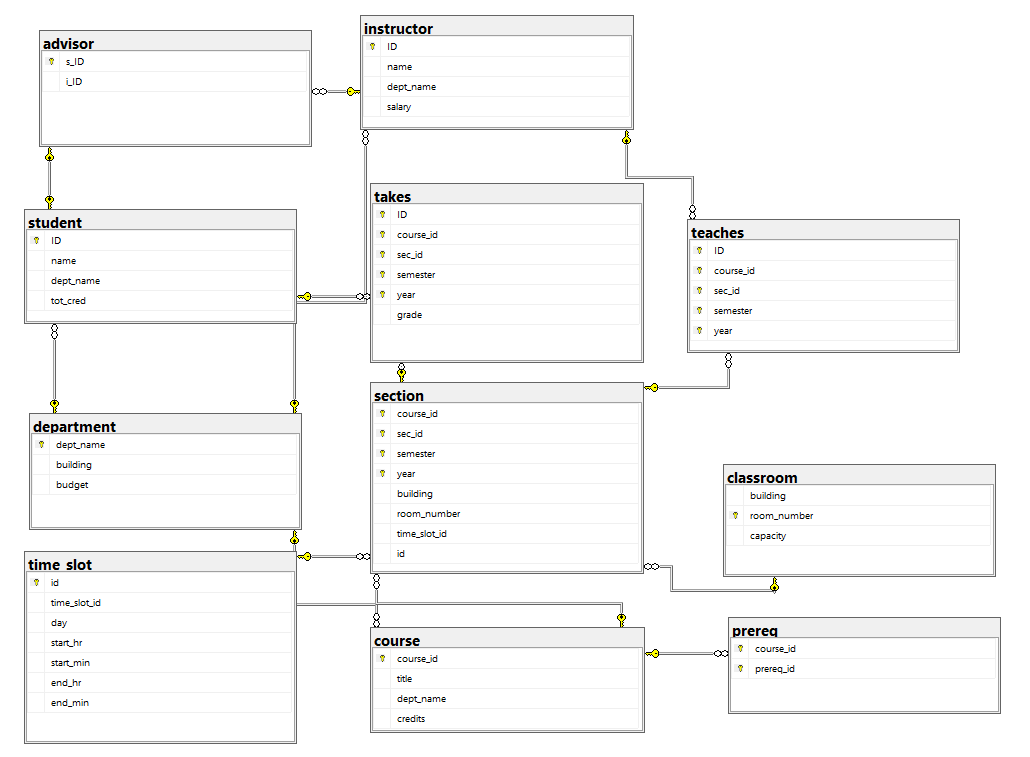
**SQL University assignment**

**Database Schema**

****

Initial queries for database, table creation and inserting values

Create and use database

create database university;

use university;

Create Tables

create table classroom

(building varchar(15),

room\_number varchar(7) primary key,

capacity numeric(4,0),

);

create table department

(dept\_name varchar(20),

building varchar(15),

budget numeric(12,2) check (budget > 0),

primary key (dept\_name)

);

create table course

(course\_id varchar(8),

title varchar(50),

dept\_name varchar(20),

credits numeric(2,0) check (credits > 0),

primary key (course\_id),

foreign key (dept\_name) references department (dept\_name)

on delete set null

);

create table instructor

(ID varchar(5),

name varchar(20) not null,

dept\_name varchar(20),

salary numeric(8,2) check (salary > 29000),

primary key (ID),

foreign key (dept\_name) references department (dept\_name)

on delete set null

);

create table time\_slot

(id int primary key,

time\_slot\_id varchar(4),

day varchar(1),

start\_hr numeric(2) check (start\_hr >= 0 and start\_hr < 24),

start\_min numeric(2) check (start\_min >= 0 and start\_min < 60),

end\_hr numeric(2) check (end\_hr >= 0 and end\_hr < 24),

end\_min numeric(2) check (end\_min >= 0 and end\_min < 60)

);

create table section

(course\_id varchar(8),

sec\_id varchar(8),

semester varchar(6)

check (semester in ('Fall', 'Winter', 'Spring', 'Summer')),

year numeric(4,0) check (year > 1701 and year < 2100),

building varchar(15),

room\_number varchar(7),

time\_slot\_id varchar(4),

id int,

primary key (course\_id, sec\_id, semester, year),

foreign key (course\_id) references course (course\_id)

on delete cascade,

foreign key (room\_number) references classroom (room\_number)

on delete set null,

foreign key (id) references time\_slot (id)

on delete set null,

);

create table teaches

(ID varchar(5),

course\_id varchar(8),

sec\_id varchar(8),

semester varchar(6),

year numeric(4,0),

primary key (ID, course\_id, sec\_id, semester, year),

foreign key (course\_id, sec\_id, semester, year) references section (course\_id, sec\_id, semester, year)

on delete cascade,

foreign key (ID) references instructor (ID)

on delete cascade

);

create table student

(ID varchar(5),

name varchar(20) not null,

dept\_name varchar(20),

tot\_cred numeric(3,0) check (tot\_cred >= 0),

primary key (ID),

foreign key (dept\_name) references department (dept\_name)

on delete set null

);

create table takes

(ID varchar(5),

course\_id varchar(8),

sec\_id varchar(8),

semester varchar(6),

year numeric(4,0),

grade varchar(2),

primary key (ID, course\_id, sec\_id, semester, year),

foreign key (course\_id, sec\_id, semester, year) references section (course\_id, sec\_id, semester, year)

on delete cascade,

foreign key (ID) references student (ID)

on delete cascade

);

create table advisor

(s\_ID varchar(5),

i\_ID varchar(5),

primary key (s\_ID),

foreign key (i\_ID) references instructor (ID)

on delete set null,

foreign key (s\_ID) references student (ID)

on delete cascade

);

create table prereq

(course\_id varchar(8),

prereq\_id varchar(8),

primary key (course\_id, prereq\_id),

foreign key (course\_id) references course (course\_id)

on delete cascade,

foreign key (prereq\_id) references course (course\_id)

);

Inserting Values

insert into classroom values ('Packard', '101', '500');

insert into classroom values ('Painter', '514', '10');

insert into classroom values ('Taylor', '3128', '70');

insert into classroom values ('Watson', '100', '30');

insert into classroom values ('Watson', '120', '50');

insert into classroom values ('Taylor', '112', '30');

insert into classroom values ('Painter', '234', '50');

insert into classroom values ('Packard', '303', '56');

insert into department values ('Biology', 'Watson', '90000');

insert into department values ('Comp. Sci.', 'Taylor', '100000');

insert into department values ('Elec. Eng.', 'Taylor', '85000');

insert into department values ('Finance', 'Painter', '120000');

insert into department values ('History', 'Painter', '50000');

insert into department values ('Music', 'Packard', '80000');

insert into department values ('Physics', 'Watson', '70000');

insert into course values ('BIO-101', 'Intro. to Biology', 'Biology', '4');

insert into course values ('BIO-301', 'Genetics', 'Biology', '4');

insert into course values ('BIO-399', 'Computational Biology', 'Biology', '3');

insert into course values ('CS-101', 'Intro. to Computer Science', 'Comp. Sci.', '4');

insert into course values ('CS-190', 'Game Design', 'Comp. Sci.', '4');

insert into course values ('CS-315', 'Robotics', 'Comp. Sci.', '3');

insert into course values ('CS-319', 'Image Processing', 'Comp. Sci.', '3');

insert into course values ('CS-347', 'Database System Concepts', 'Comp. Sci.', '3');

insert into course values ('EE-181', 'Intro. to Digital Systems', 'Elec. Eng.', '3');

insert into course values ('FIN-201', 'Investment Banking', 'Finance', '3');

insert into course values ('HIS-351', 'World History', 'History', '3');

insert into course values ('MU-199', 'Music Video Production', 'Music', '3');

insert into course values ('PHY-101', 'Physical Principles', 'Physics', '4');

insert into instructor values ('10101', 'Srinivasan', 'Comp. Sci.', '65000');

insert into instructor values ('12121', 'Wu', 'Finance', '90000');

insert into instructor values ('15151', 'Mozart', 'Music', '40000');

insert into instructor values ('22222', 'Einstein', 'Physics', '95000');

insert into instructor values ('32343', 'El Said', 'History', '60000');

insert into instructor values ('33456', 'Gold', 'Physics', '87000');

insert into instructor values ('45565', 'Katz', 'Comp. Sci.', '75000');

insert into instructor values ('58583', 'Califieri', 'History', '62000');

insert into instructor values ('76543', 'Singh', 'Finance', '80000');

insert into instructor values ('76766', 'Crick', 'Biology', '72000');

insert into instructor values ('83821', 'Brandt', 'Comp. Sci.', '92000');

insert into instructor values ('98345', 'Kim', 'Elec. Eng.', '80000');

insert into time\_slot values (1,'A', 'M', '8', '0', '8', '50');

insert into time\_slot values (2,'A', 'W', '8', '0', '8', '50');

insert into time\_slot values (3,'A', 'F', '8', '0', '8', '50');

insert into time\_slot values (4,'B', 'M', '9', '0', '9', '50');

insert into time\_slot values (5,'B', 'W', '9', '0', '9', '50');

insert into time\_slot values (6,'B', 'F', '9', '0', '9', '50');

insert into time\_slot values (7,'C', 'M', '11', '0', '11', '50');

insert into time\_slot values (8,'C', 'W', '11', '0', '11', '50');

insert into time\_slot values (9,'C', 'F', '11', '0', '11', '50');

insert into time\_slot values (10,'D', 'M', '13', '0', '13', '50');

insert into time\_slot values (11,'D', 'W', '13', '0', '13', '50');

insert into time\_slot values (12,'D', 'F', '13', '0', '13', '50');

insert into time\_slot values (13,'E', 'T', '10', '30', '11', '45 ');

insert into time\_slot values (14,'E', 'R', '10', '30', '11', '45 ');

insert into time\_slot values (15,'F', 'T', '14', '30', '15', '45 ');

insert into time\_slot values (16,'F', 'R', '14', '30', '15', '45 ');

insert into time\_slot values (17,'G', 'M', '16', '0', '16', '50');

insert into time\_slot values (18,'G', 'W', '16', '0', '16', '50');

insert into time\_slot values (19,'G', 'F', '16', '0', '16', '50');

insert into time\_slot values (20,'H', 'W', '10', '0', '12', '30');

insert into section values ('BIO-101', '1', 'Summer', '2017', 'Painter', '514', 'B',1);

insert into section values ('BIO-301', '1', 'Summer', '2018', 'Painter', '514', 'A',2);

insert into section values ('CS-101', '1', 'Fall', '2017', 'Packard', '101', 'H',3);

insert into section values ('CS-101', '1', 'Spring', '2018', 'Packard', '101', 'F',4);

insert into section values ('CS-190', '1', 'Spring', '2017', 'Taylor', '3128', 'E',5);

insert into section values ('CS-190', '2', 'Spring', '2017', 'Taylor', '3128', 'A',6);

insert into section values ('CS-315', '1', 'Spring', '2018', 'Watson', '120', 'D',7);

insert into section values ('CS-319', '1', 'Spring', '2018', 'Watson', '100', 'B',8);

insert into section values ('CS-319', '2', 'Spring', '2018', 'Taylor', '3128', 'C',9);

insert into section values ('CS-347', '1', 'Fall', '2017', 'Taylor', '3128', 'A',10);

insert into section values ('EE-181', '1', 'Spring', '2017', 'Taylor', '3128', 'C',11);

insert into section values ('FIN-201', '1', 'Spring', '2018', 'Packard', '101', 'B',12);

insert into section values ('HIS-351', '1', 'Spring', '2018', 'Painter', '514', 'C',13);

insert into section values ('MU-199', '1', 'Spring', '2018', 'Packard', '101', 'D',14);

insert into section values ('PHY-101', '1', 'Fall', '2017', 'Watson', '100', 'A',15);

insert into teaches values ('10101', 'CS-101', '1', 'Fall', '2017');

insert into teaches values ('10101', 'CS-315', '1', 'Spring', '2018');

insert into teaches values ('10101', 'CS-347', '1', 'Fall', '2017');

insert into teaches values ('12121', 'FIN-201', '1', 'Spring', '2018');

insert into teaches values ('15151', 'MU-199', '1', 'Spring', '2018');

insert into teaches values ('22222', 'PHY-101', '1', 'Fall', '2017');

insert into teaches values ('32343', 'HIS-351', '1', 'Spring', '2018');

insert into teaches values ('45565', 'CS-101', '1', 'Spring', '2018');

insert into teaches values ('45565', 'CS-319', '1', 'Spring', '2018');

insert into teaches values ('76766', 'BIO-101', '1', 'Summer', '2017');

insert into teaches values ('76766', 'BIO-301', '1', 'Summer', '2018');

insert into teaches values ('83821', 'CS-190', '1', 'Spring', '2017');

insert into teaches values ('83821', 'CS-190', '2', 'Spring', '2017');

insert into teaches values ('83821', 'CS-319', '2', 'Spring', '2018');

insert into teaches values ('98345', 'EE-181', '1', 'Spring', '2017');

insert into student values ('00128', 'Zhang', 'Comp. Sci.', '102');

insert into student values ('12345', 'Shankar', 'Comp. Sci.', '32');

insert into student values ('19991', 'Brandt', 'History', '80');

insert into student values ('23121', 'Chavez', 'Finance', '110');

insert into student values ('44553', 'Peltier', 'Physics', '56');

insert into student values ('45678', 'Levy', 'Physics', '46');

insert into student values ('54321', 'Williams', 'Comp. Sci.', '54');

insert into student values ('55739', 'Sanchez', 'Music', '38');

insert into student values ('70557', 'Snow', 'Physics', '0');

insert into student values ('76543', 'Brown', 'Comp. Sci.', '58');

insert into student values ('76653', 'Aoi', 'Elec. Eng.', '60');

insert into student values ('98765', 'Bourikas', 'Elec. Eng.', '98');

insert into student values ('98988', 'Tanaka', 'Biology', '120');

insert into takes values ('00128', 'CS-101', '1', 'Fall', '2017', 'A');

insert into takes values ('00128', 'CS-347', '1', 'Fall', '2017', 'A-');

insert into takes values ('12345', 'CS-101', '1', 'Fall', '2017', 'C');

insert into takes values ('12345', 'CS-190', '2', 'Spring', '2017', 'A');

insert into takes values ('12345', 'CS-315', '1', 'Spring', '2018', 'A');

insert into takes values ('12345', 'CS-347', '1', 'Fall', '2017', 'A');

insert into takes values ('19991', 'HIS-351', '1', 'Spring', '2018', 'B');

insert into takes values ('23121', 'FIN-201', '1', 'Spring', '2018', 'C+');

insert into takes values ('44553', 'PHY-101', '1', 'Fall', '2017', 'B-');

insert into takes values ('45678', 'CS-101', '1', 'Fall', '2017', 'F');

insert into takes values ('45678', 'CS-101', '1', 'Spring', '2018', 'B+');

insert into takes values ('45678', 'CS-319', '1', 'Spring', '2018', 'B');

insert into takes values ('54321', 'CS-101', '1', 'Fall', '2017', 'A-');

insert into takes values ('54321', 'CS-190', '2', 'Spring', '2017', 'B+');

insert into takes values ('55739', 'MU-199', '1', 'Spring', '2018', 'A-');

insert into takes values ('76543', 'CS-101', '1', 'Fall', '2017', 'A');

insert into takes values ('76543', 'CS-319', '2', 'Spring', '2018', 'A');

insert into takes values ('76653', 'EE-181', '1', 'Spring', '2017', 'C');

insert into takes values ('98765', 'CS-101', '1', 'Fall', '2017', 'C-');

insert into takes values ('98765', 'CS-315', '1', 'Spring', '2018', 'B');

insert into takes values ('98988', 'BIO-101', '1', 'Summer', '2017', 'A');

insert into takes values ('98988', 'BIO-301', '1', 'Summer', '2018', null);

insert into advisor values ('00128', '45565');

insert into advisor values ('12345', '10101');

insert into advisor values ('23121', '76543');

insert into advisor values ('44553', '22222');

insert into advisor values ('45678', '22222');

insert into advisor values ('76543', '45565');

insert into advisor values ('76653', '98345');

insert into advisor values ('98765', '98345');

insert into advisor values ('98988', '76766');

insert into prereq values ('BIO-301', 'BIO-101');

insert into prereq values ('BIO-399', 'BIO-101');

insert into prereq values ('CS-190', 'CS-101');

insert into prereq values ('CS-315', 'CS-101');

insert into prereq values ('CS-319', 'CS-101');

insert into prereq values ('CS-347', 'CS-101');

insert into prereq values ('EE-181', 'PHY-101');

Assignment Questions

1. **Display average salary given by each department.**

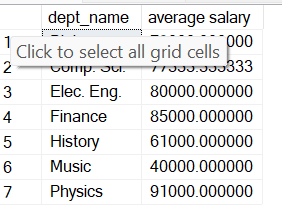
**Solution –**

Select dept\_name , AVG(salary) as average\_salary

From instructor

Group By dept\_name;

Output:



### 2. Display the name of students and their corresponding course IDs.

**Solution –**

select name , course\_id

From student st

JOin course co

On st.dept\_name = co.dept\_name

Group By st.name , co.course\_id;

output:



**3. Display number of courses taken by each student.**

**Solution –**

Select name , count(course\_id) as 'Number of Course'

From student, takes

Where student.ID = takes.ID

Group By name ;

Output:

****

### 4. Get the prerequisites courses for courses in the Spring semester.

**Solution –**

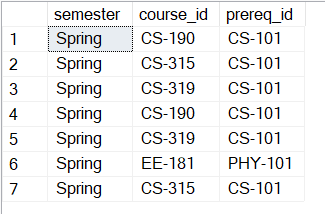
Select takes.semester , takes.course\_id , prereq.prereq\_id

From takes , prereq

Where takes.course\_id = prereq.course\_id

### And semester='spring' ;

### output:

****

### 5. Display the instructor name who teaches student with highest 5 credits.

**Solution –**

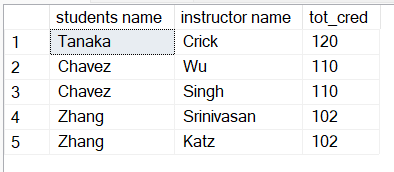
Select Top 5 student.name, instructor.name , student.tot\_cred

From student , instructor

Where student.dept\_name =instructor.dept\_name and tot\_cred> 5

### Order By tot\_cred Desc;

### Output:

****

### 6. Which semester and department offers maximum number of courses.

### Solution –

select top 1

section.semester,course.dept\_name,count(section.course\_id) as 'max'

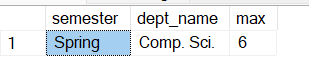
from section inner join course

on section.course\_id=course.course\_id

group by section.semester,course.dept\_name

### order by count(section.course\_id) desc;

### Output:

****

### 7. Display course and department whose time starts at 8.

**Solution –**

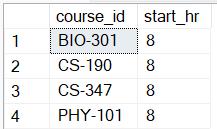
select section.course\_id,time\_slot.start\_hr

from section,time\_slot

where time\_slot.time\_slot\_id=section.time\_slot\_id and start\_hr='8'

### group by section.course\_id,time\_slot.start\_hr;

### Output:

****

### 8. Display the salary of instructors from Watson building.

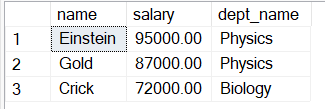
**Solution –**

Select instructor.name , instructor.salary , department.dept\_name

From instructor ,department

### Where instructor.dept\_name = department.dept\_name and building= 'Watson';

### Output:

****

### 9. Show the title of courses available on Monday.

**Solution –**

select course.title,time\_slot.day

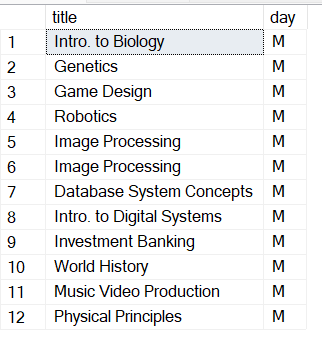
from time\_slot inner join section

on time\_slot.time\_slot\_id=section.time\_slot\_id

inner join course on course.course\_id=section.course\_id

### where day='M';

### output:

****

### 10. Find the number of courses that start at 8 and end at 8.

**Solution –**

select start\_hr,end\_hr,count(course\_id) as 'number of courses'

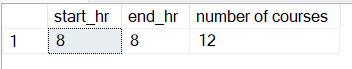
from time\_slot Join section

ON time\_slot.time\_slot\_id=section.time\_slot\_id and start\_hr='8'

and end\_hr='8'

group by start\_hr,end\_hr;

### Output:

****

### 11. Find instructors having salary more than 90000.

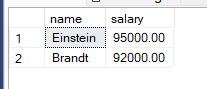
**Solution –**

Select name , salary

From instructor

where salary > 90000;

Output:

****

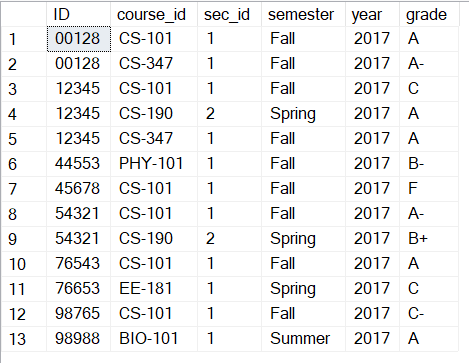
### 12. Find student records taking courses before 2018.

**Solution –**

select \* from takes

where year<'2018';

### Output:

****

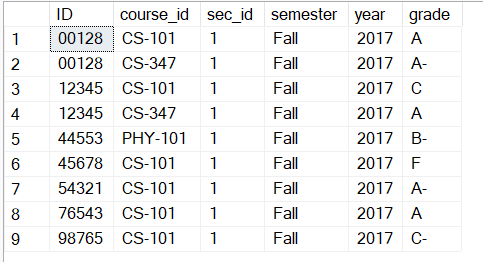
### 13. Find student records taking courses in the fall semester and coming under first section.

**Solution –**

select \* from takes

where sec\_id=1 and semester='Fall';

### Output:

****

### 14. Find student records taking courses in the fall semester and coming under second section.

**Solution –**

select \* from takes

where sec\_id=2 and semester='Fall';

### Output:

****

**15. Find student records taking courses in the summer semester, coming under first section in the year 2017.**

**Solution –**

select \* from takes

where year=2017 and semester='Summer';

Output:

****

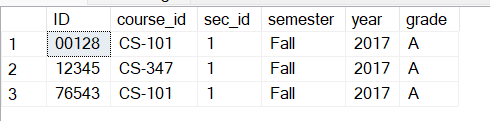
### 16. Find student records taking courses in the fall semester and having A grade.

**Solution –**

select \* from takes

where grade='A' and semester='Fall';

Output:

****

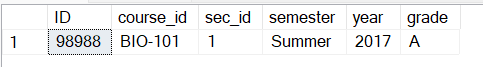
### 17. Find student records taking courses in the summer semester and having A grade.

**Solution –**

select \* from takes

where grade='A' and semester='Summer';

### Output:

****

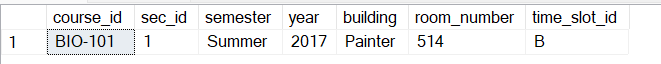
### 18. Display section details with B time slot, room number 514 and in the Painter building.

**Solution –**

select \* from section

where time\_slot\_id = 'B' and room\_number = 514 and building = 'Painter' ;

### Output:

****

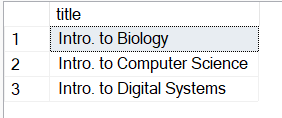
**19. Find all course titles which have a string "Intro.".**

**Solution –**

select title from course

where title like '%Intro.%';

Output:

****

### 20. Find the titles of courses in the Computer Science department that have 3 credits.

**Solution –**

select title from course

where dept\_name='Comp. Sci.' and credits=3;

Output:

### 

### 21. Find IDs and titles of all the courses which were taught by an instructor named Einstein. Make sure there are no duplicates in the result.

**Solution –**

select course.course\_id,course.title from

course,instructor,teaches

where course.course\_id=teaches.course\_id and

teaches.ID=instructor.ID

and instructor.name='Einstein';

Output:

### 

### 22. Find all course IDs which start with CS

**Solution –**

Select course\_id

From course

wHERE course\_id like 'CS%' ;

### Output:

### 

**23. For each department, find the maximum salary of instructors in that department.**

**Solution –**

Select dept\_name , max(salary) as 'Max salary'

from instructor

group by dept\_name;

Output:

### 

**24. Find the enrollment (number of students) of each section that was offered in Fall 2017.**

**Solution –**

Select sec\_id , count(id) as 'no of student ' , year , semester

from takes

where year = '2017' and semester = 'Fall'

group by sec\_id,year,semester;

Output:

### 

### 25. Increase(update) the salary of each instructor by 10% if their current salary is between 0 and 90000.

**Solution –**

update instructor

set salary=salary \* 1.1

where salary between 0 and 90000;

### select name, salary from instructor;

### Output:

### 

### 26. Find the names of instructors from Biology department having salary more than 50000.

**Solution –**

Select name , dept\_name , salary

from instructor

### where salary > 50000 and dept\_name = 'Biology';

### Output:

### 

### 27. Find the IDs and titles of all courses taken by a student named Shankar.

**Solution –**

Select course.course\_id , course.title , student.name

from course , student

Where course.dept\_name = student.dept\_name and

student.name = 'Shankar' ;

### Output:

### 

### 28. For each department, find the total credit hours of courses in that department.

**Solution –**

select dept\_name ,sum(credits) as 'total credits'

from course

group by dept\_name;

### Output:

### 

### 29. Find the number of courses having A grade in each building.

**Solution –**

Output: Select section.building, COUNT(takes.course\_id) as 'Number of courses ', takes.grade

From section Join takes

ON section.course\_id = takes.course\_id and takes.grade ='A'

group by section.building, takes.grade;

### 

### 30. Display number of students in each department having total credits divisible by course credits.

**Solution –**

Select student.dept\_name , count(student.ID) as 'No of students'

From student join course ON student.dept\_name = course.dept\_name

Where (tot\_cred % credits =0)

Group By student.dept\_name;

Output:

### 

### 31. Display number of courses available in each building.

**Solution –**

select department.building,count(course.course\_id) as 'number of courses'

from department,course

where course.dept\_name=department.dept\_name

### group by building;

### Output:

### 

### 32. Find number of instructors in each department having 'a' and 'e' in their name.

**Solution –**

Select dept\_name , COUNT(id) as 'no of instructor'

From instructor

Where name LIKe '%a%%e%'

Group by dept\_name;

### Output:

### 

### 33. Display number of courses being taught in classroom having capacity more than 20.

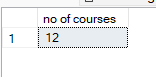
**Solution –**

Select count(section.course\_id) as 'no of courses'

From section , classroom

Where section.room\_number = classroom.room\_number and

classroom.capacity > 20 ;



### [or]

Select count(section.course\_id) as 'No of courses' , classroom.room\_number , classroom.capacity

From section , classroom

Where section.room\_number = classroom.room\_number and

classroom.capacity > 20

### Group by classroom.room\_number,classroom.capacity;

### 

### 34. Update the budget of each department by Rs. 1000

**Solution –**

update department

set budget = budget+1000 ;

### select budget from department;

### before

### After

### 35. Find number of students in each room.

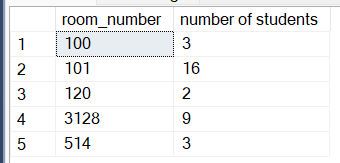
### Solution –

select section.room\_number,COUNT(takes.ID) as 'number of students'

from section,takes

where takes.course\_id=section.course\_id

group by room\_number;

****

### 36. Give the prerequisite course for each student.

**Solution –**

select name,prereq\_id from student,takes,prereq

where student.id=takes.id and takes.course\_id=prereq.course\_id;

### Output:

****

### 37. Display number of students attending classes on Wednesday.

**Solution –**

select day,count(takes.id) as 'number of students'

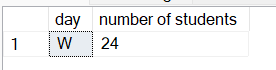
from time\_slot ,section,takes

where time\_slot.time\_slot\_id=section.time\_slot\_id

and section.course\_id=takes.course\_id and day='W'

group by day

**Output:**

****

### 38. Display number of students and instructors in each department

**Solution –**

SELECT

s.dept\_name,

COUNT(DISTINCT s.id) AS 'number\_of\_students',

COUNT(DISTINCT i.id) AS 'number\_of\_instructors'

FROM

student s

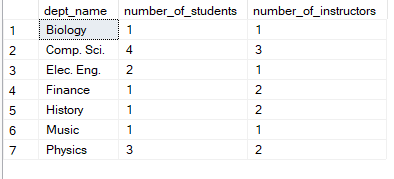
LEFT JOIN

instructor i ON s.dept\_name = i.dept\_name

GROUP BY

s.dept\_name;

Output:



### 39. Display number of students in each semester and their sum of credits.

### Solution –

select takes.semester,count(student.ID) as 'number of student',

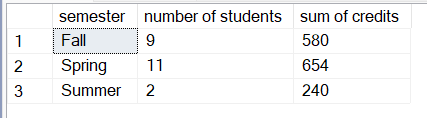
sum(student.tot\_cred) as 'sum of credits'

from takes,student

where takes.ID=student.ID

group by semester;

Output:



### 40. Give number of instructors in each building.

**Solution –**

select section.building,count(instructor.id) as 'number of instructor'

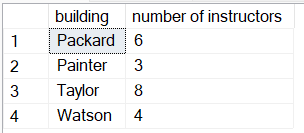
from instructor,teaches,section

where teaches.course\_id=section.course\_id and

teaches.ID=instructor.ID

group by section.building;

Output:



### 41. Display advisor IDs for instructors in Painter building.

**Solution –**

select section.building,instructor.name,advisor.s\_id

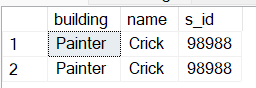
from instructor,advisor,teaches,section

where instructor.id=advisor.i\_ID and teaches.ID=instructor.ID

and teaches.course\_id=section.course\_id

and building='Painter';

Output:



### 42. Find total credits earned by students coming at 9am

**Solution –**

select student.name,time\_slot.start\_hr,student.tot\_cred

from student,takes,section,time\_slot

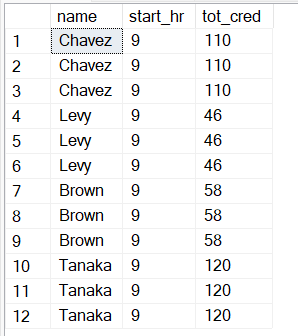
where student.ID=takes.ID and

section.time\_slot\_id=time\_slot.time\_slot\_id

and takes.course\_id=section.course\_id

and time\_slot.start\_hr='9';

Output:



### 43. Display student names ordered by room number

**Solution –**

select student.name,section.room\_number

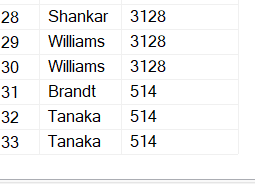
from student,takes,section

where student.ID=takes.ID and takes.course\_id=section.course\_id

order by room\_number;

Output:





### 44. Find the number of capacity left after occupying all the students.

**Solution –**

select classroom.room\_number,classroom.capacity-COUNT(takes.id) as

'remaining capacity'

from takes,section,classroom

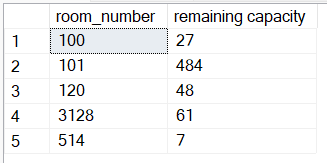
where section.course\_id=takes.course\_id and

section.room\_number=classroom.room\_number

group by

section.room\_number,classroom.room\_number,classroom.capacity;

Output:



### 45. Find the duration for which each student has to attend each lecture.

**Solution –**

Select student.name,takes.course\_id,time\_slot.end\_hr\_-time\_slot.start\_hr as 'duration',

time\_slot.end\_min-\_time\_slot.start\_min as 'duration in minutes'

from student,takes,time\_slot,section

where student.ID=takes.ID and

section.time\_slot\_id=time\_slot.time\_slot\_id

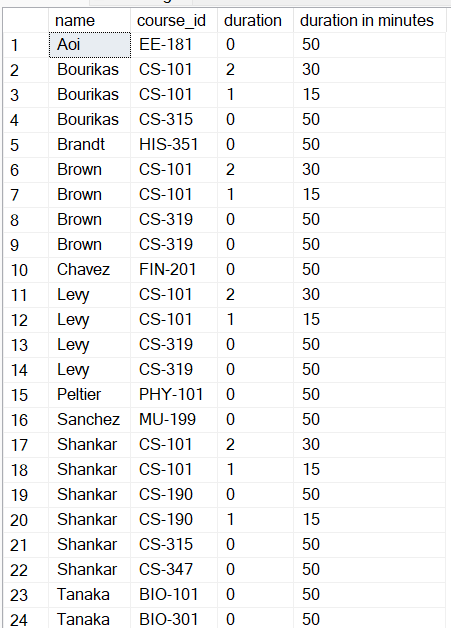
and takes.course\_id=section.course\_id

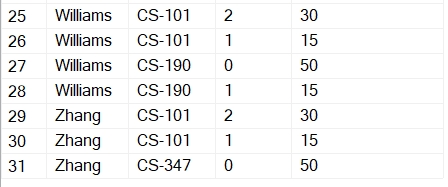
group by

student.name,takes.course\_id,time\_slot.start\_hr,time\_slot.end\_hr,

time\_slot.end\_min,time\_slot.start\_min;

Output:





### 46. Create a timetable for the university.

**Solution –**

Select day , building , room\_number ,takes.course\_id

From takes , section ,time\_slot

Where section.time\_slot\_id=time\_slot.time\_slot\_id

and takes.course\_id=section.course\_id

group by day,building, room\_number ,takes.course\_id ;

**[Or]**

select

time\_slot.day,section.building,section.room\_number,section.course\_id

from time\_slot,section

where section.time\_slot\_id=time\_slot.time\_slot\_id

group by day,room\_number,building,course\_id;

Output:

### 

### 

### 

### 47. Find the average salary that's distributed to teachers for each course and sort them in descending order

**Solution –**

Select course.title , AVG(instructor.salary) as 'Average Salary'

From instructor , teaches ,course

Where instructor.ID = teaches.ID

and teaches.course\_id = course.course\_id

Group BY course.title

Order by [Average Salary] Desc ;

### Output:

### 

### 48. Find the average duration of classes for each course id

**Solution –**

Select course.course\_id , AVG(time\_slot.end\_min - time\_slot.start\_min) as ‘duration’

From time\_slot , section , course

Where time\_slot.time\_slot\_id = section.time\_slot\_id and

section.course\_id = course.course\_id

Group by course.course\_id;

[Or]

with time\_slot\_duration as (

select time\_slot\_id,(end\_min-start\_min) as duration

from time\_slot

)

select section.course\_id,AVG(duration) as duration

from section join time\_slot\_duration

on section.time\_slot\_id=time\_slot\_duration.time\_slot\_id

group by section.course\_id;

### Output:

### 

### 49 Get the name of the instructor with highest salary from each department.

**Solution –**

Select dept\_name , name ,MAX(salary) as 'Max salary'

From instructor

Group by dept\_name , name ;

Output:

### 

### 50. Get the sum of the total credits of students that is dealt by the instructors along with their names

**Solution –**

select instructor.name , sum(student.tot\_cred) as 'Total Credits'

from advisor , student , instructor

Where advisor.s\_ID = student.ID and

advisor.i\_ID = instructor.ID

Group BY instructor.name ;

### 

### 51. Perform division between student credits and department total credits.

**Solution –**

with dept\_creds as (

select dept\_name,sum(credits) as dept\_total\_creds

from course

group by dept\_name

)

select student.name,student.tot\_cred/dept\_creds.dept\_total\_creds as 'Division'

from student join dept\_creds

on student.dept\_name=dept\_creds.dept\_name;

[Or]

Select student.name, (student.tot\_cred / sum(course.credits)) as 'Division'

From course Join student

On course.dept\_name = student.dept\_name

Group By course.dept\_name ,student.name , student.tot\_cred;

### 

### 52. If the department budget was to be distributed among the buildings, how much amount can be allocated to each room in a building

**Solution –**

With building\_room\_data as (

select building,count(room\_number) as num\_rooms

from classroom

group by building

)

select bb.building,bb.building\_budget/num\_rooms as room\_budget

from building\_room\_data as brd

join

(

select building,sum(budget) as building\_budget

from department

group by building

) as bb

On

bb.building=brd.building;

### output:

### 