Group No. 16

Manjal Shah(202003037)

Piyush Parmar(202003038)

Lab No. 10(29/10/2021)

Online Learning Platform Management System

❖ Requirements :-

1) Find the name of the course taught by an educator having EID=5.
   - ➢ Query in relational algebra :

     $\Pi_{course\_name}(\sigma_{(eid=5)}\ Course)$

   - ➢ SQL query :

     select course_name from course where eid = 5

```
507  select course_name from course where eid = 5
```

Data Output | Explain | Messages | Notifications

	course_name character varying (50)
1	Probability and statistics

2) Find Email IDs of educators working in Google.
   - ➢ Query in relational algebra :

     $\Pi_{email\_id}(\sigma_{(Company\_Name='Google')}\ Educator)$

➢ SQL query :

select email_id from educator where company_name = 'Google'

```
508  select email_id from educator where company_name = 'Google'
```

Data Output | Explain | Messages | Notifications

	email_id character varying (60)
1	davidwilliams@gmail.com
2	meganfox@gmail.com

3) Find the names of the cities of students who have taken the course having CID=MA101

➢ Query in relational algebra :

$\Pi_{city}(\sigma_{(CID='MA101')}(\text{Student} \bowtie \text{Learn}))$

➢ SQL query :

select city from learn natural join student where cid = 'MA101'

```
509  select city from learn natural join student where cid = 'MA101'
```

Data Output | Explain | Messages | Notifications

	city character varying (20)
1	mumbai
2	delhi
3	delhi
4	bangalore
5	ajmer

4) Find contact numbers of all female students.

➢ Query in relational algebra :

$\Pi_{\text{contact no.}}(\sigma_{(\text{gender='F'})}(\text{Student\_Contact} \bowtie \text{Student}))$

➢ SQL query :

select contact_no from student natural join student_contact where gender = 'F'

```
510  select contact_no from student natural join student_contact where gender = 'F'
```

Data Output | Explain | Messages | Notifications

	contact_no bigint
1	9854741122
2	9855223636
3	9745255585
4	9985774455
5	9985212121
6	9985744545
7	9685668574
8	9752452365
9	9998545125
10	9987452522
11	9985653322
12	9874521452
13	9726712365
14	7874542536
15	9985632145

5) Find name of all the courses who belongs to domain "Competitive programming"

➢ Query in relational algebra :

$\Pi_{\text{course\_name.}}(\sigma_{(\text{d.name='Competitive Programming'})}(\text{Domain} \ \theta_{(\text{Domain.DID = course.Domain\_ID})} \ \text{Course}))$

➢ SQL query :

select course_name from course join domain on course.domain_id = domain.did and dname = 'Competitive Programming'

```
511 select course_name from course join domain on course.domain_id = domain.did and dname = 'Competitive Programming'
```

Data Output Explain Messages Notifications

	course_name character varying (50)
1	DSA in C++
2	JAVA Core and Advance
3	Graphs and algorithms

6) Find the total number of students who have taken a course on “DSA in c++”.

➢ Query in relational algebra :

$$_{(\text{course\_name})}\mathcal{F}_{(\text{count (sid) , course\_name})} \left(\sigma_{(\text{ course\_name = 'DSA in C++' })} (\text{course} \bowtie \text{learn}) \right)$$

➢ SQL query :

select course_name,count(sid) from course natural join learn where course_name = 'DSA in C++' group by(course_name)

```
516 select course_name,count(sid) from course natural join learn where course_name = 'DSA in C++' group by(course_name)
```

Data Output Explain Messages Notifications

	course_name character varying (50)	count bigint
1	DSA in C++	3

7) Find rating for an educator having eid=2.

➢ Query in relational algebra :

$\Pi_{(\text{ eid, rating })}(\sigma_{(\text{eid=2})}\ \text{educator})$

➢ SQL query :

select eid, rating from educator where eid=2

```
8   select eid, rating from educator where eid=2
```

Data Output | Explain | Messages | Notifications

	eid [PK] integer	rating numeric (3,2)
1	2	4.75

8) Find the name of domains which have more than 2 courses.

➢ Query in relational algebra :

$\sigma_{(\text{ count (domain\_id) > 2 })}\ (\ {}_{\text{dname}}\mathcal{F}_{(\text{count(domain\_id) , dname })}\ (\text{course}\ \theta\ \text{domain})_{(\text{course.domain\_id=domain.did})}\ )$

➢ SQL query :

select dname,count(domain_id) from course join domain on course.domain_id = domain.did group by(dname) having count(domain_id) > 2

```
528 elect dname,count(domain_id) from course join domain on course.domain_id = domain.did group by(dname) having count(domain_id) > 2
```

Data Output | Explain | Messages | Notifications

	dname character varying (50)	count bigint
1	Competitive Programming	3
2	Machine learning and Artifici...	4

9) Find the name of the courses which have more fees than the average fees for all live courses.

➢ Query in relational algebra :

$\Pi_{course\_name}(course \bowtie_{cid=live\_id} (\sigma_{(fees>avg\_fees)}(live\_course \bowtie (\mathcal{F}_{avg(fees)\rightarrow avg\_fees}(live\_course))))$

➢ SQL query :

```
select course_name from course where cid in(
    select live_id from live_course natural join(
        select avg(fees) as avg_fees from live_course
    )as r1 where fees>avg_fees
)
```

```
3  select course_name from course where cid in(
4      select live_id from live_course natural join(
5          select avg(fees) as avg_fees from live_course
6      )as r1 where fees>avg_fees
7  )
```

Data Output | Explain | Messages | Notifications

	course_name character varying (50)
1	Logistic regression
2	Probability and statistics
3	DSA in C++

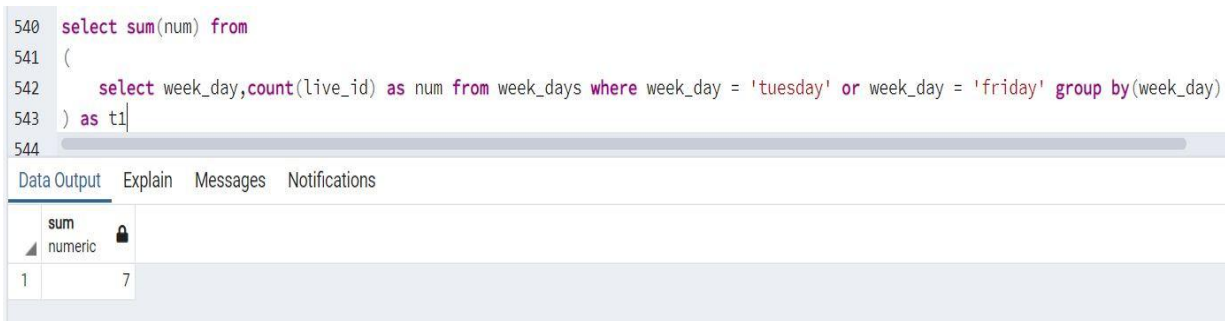
10) Find the total number of live courses held on Tuesdays or Fridays.

➢ Query in relational algebra :

$\mathcal{F}_{(sum(num))}$ ($_{week\_day}\mathcal{F}_{(count(live\_id) \rightarrow num,\ week\_day)}$ ($\sigma_{(week\_day = 'tuesday' \text{ or } 'friday')}$ (week_day)))

➢ SQL query :

```
select sum(num) from
(
        select week_day,count(live_id) as num from week_days where week_day = 'tuesday' or week_day = 'friday' group by(week_day)
) as t1
```



11) Find the name of the students who have taken the maximum number of courses.

➢ Query in relational algebra :

$\Pi_{((fname\ ||' '||\ lname) \rightarrow name,\ noc)}$ (student $\bowtie$ (($_{sid}\mathcal{F}_{sid,\ count(cid) \rightarrow noc}$ (learn)) θ

($\mathcal{F}_{max(noc) \rightarrow mc}$ ($_{sid}\mathcal{F}_{sid,\ count(cid) \rightarrow noc}$ (learn)))$_{noc=mc}$))

➢ SQL query :

```
select fname ||' '|| lname as name, noc from student natural join (
    select * from ( select sid,count(cid) as noc from learn group by sid ) as r1
    join (
```

select max(noc) as mc from (select sid,count(cid) as noc from learn group by sid) as r2

) as r3 on noc=mc

) as r4

```
select fname ||' '|| lname as name, noc from student natural join (
    select * from ( select sid,count(cid) as noc from learn group by sid ) as r1
    join (
    select max(noc) as mc from ( select sid,count(cid) as noc from learn group by sid ) as r2
    ) as r3 on noc=mc
) as r4
```

Data Output | Explain | Messages | Notifications

	name text	noc bigint
1	donal b...	3

12) Find the total number of students who have passed the exam for a live course which has live_id=MA104.

- Query in relational algebra :

$$\mathcal{F}_{(\text{count (sid)})}\ \sigma_{(\text{live\_id= 'MA104' and status = 'Pass'})}\ (\text{result} \bowtie \text{exam})$$

- SQL query :

select count(sid) from result natural join exam where status = 'pass' and live_id = 'MA104'

```
select count(sid) from result natural join exam where status = 'pass' and live_id = 'MA104'
```

Data Output | Explain | Messages | Notifications

	count bigint
1	3

13) Find the name of companies which have educators with less experience than the average experience of all educators.

➢ Query in relational algebra :

$\Pi_{company\_name}(\sigma_{((exeperience\_in\_years < (\mathcal{F}\_(avg(exeperience\_in\_years))(\sigma\_(company\_name!='Platform')(educator)))) \text{ and } (company\_name != 'Platform'))}(educator))$

➢ SQL query :

```
select company_name from educator

        where exeperience_in_years <

(

        select avg(exeperience_in_years) from educator where
company_name <> 'Platform'

) and company_name <> 'Platform'
```

```
552 select company_name from educator
553     where exeperience_in_years <
554 (
555     select avg(exeperience_in_years) from educator where company_name <> 'Platform'
556 ) and company_name <> 'Platform'
```

Data Output | Explain | Messages | Notifications

	company_name character varying (30)
1	Google
2	Amazon
3	Facebook

14) Find the total number of girl students who have got at least 1 certificate.

➢ Query in relational algebra :

$\mathcal{F}_{(count(distinct(sid)))}(\sigma_{(gender = 'F' \text{ and } status = 'Pass')}(result \bowtie student))$

➢ SQL query :

select count(distinct(sid)) from result natural join student where gender = 'F' and status = 'pass'

```
560  select count(distinct(sid)) from result natural join student where gender = 'F' and status = 'pass'
```

Data Output | Explain | Messages | Notifications

	count bigint
1	6

15) Find the total number of providing skills for each course having more no. of lessons than average number of lessons for all courses.

➢ Query in relational algebra :

$$_{cid}\mathcal{F}_{(count(providing\_skill),\ cid)}\ (\ \sigma_{(\ no\_of\_lesson\ >\ \mathcal{F}\_(avg(no\_of\_lesson))\ (course)}$$
$$(course\_provide \bowtie course\ )\ )$$

➢ SQL query :

select count(providing_skill),cid from course natural join course_provide where no_of_lesson > (select avg(no_of_lesson) from course) group by cid

```
1
2  select count(providing_skill),cid from course natural join course_provide
3      where no_of_lesson > (select avg(no_of_lesson) from course) group by cid
```

Data Output | Explain | Messages | Notifications

	count bigint	cid [PK] character (5)
1	1	AM101
2	2	MA102
3	1	CP103
4	1	CP102
5	1	AM102

16) Find the list of students who have taken all the recorded courses.

➢ Query in relational algebra :

$\Pi_{(sid)}$(student $\bowtie_{(sid <> sid)}$($\Pi_{(sid,cid)}$(student cross join ($\Pi_{(cid)}$($\sigma_{(type\_of\_course = 'Record')}$(course)) - $\Pi_{(sid,cid)}$(learn)))))

➢ SQL query :

```
select sid from student where sid not in (
        select sid from (select sid,cid from student cross join (
                select cid from course where type_of_course =
'Record')as c except select sid,cid from learn)
        as t1
)
```

```
select sid from student where sid not in (
    select sid from (select sid,cid from student cross join (
        select cid from course where type_of_course = 'Record')as c except select sid,cid from learn)
    as t1
)
```

Data Output | Explain | Messages | Notifications

	sid [PK] integer
1	13

17) Find the list of educators who have taught all the courses which have "Basic Java" as a prerequisite.

➢ Query in relational algebra :

$\Pi_{(fname)}$(educator $\bowtie_{(eid <> eid)}$ ($\Pi_{(eid)}$($\Pi_{(eid,ci\text{->}cid)}$(educator cross join ($\Pi_{(cid\text{->}ci)}$($\sigma_{(prerequisite\_skill = \text{'Basic Java'})}$(course_prerequiste $\bowtie$ course)) - $\Pi_{(eid,cid)}$ (course)))))))

➢ SQL query :

```
select fname from educator where eid not in (
        select eid from (
                select ci as cid,eid from educator cross join (
                        select cid as ci from course_prerequiste natural
join course where prerequiste_skill = 'Basic Java')
                as t1 except select cid,eid from course
) as t2
)
```

```
146 select fname from educator where eid not in (
147     select eid from (
148         select ci as cid,eid from educator cross join (
149             select cid as ci from course_prerequiste natural join course where prerequiste_skill = 'Basic Java')
150         as t1 except select cid,eid from course
151 ) as t2
152 )
```

Data Output | Explain | Messages | Notifications

	fname character varying (20)
1	Rahul

18) Find the list of students who have given feedback for all the courses which they have taken.

➢ Query in relational algebra :

$\Pi_{(sid\ ,\ (fname\ ||'\ '||\ name)\ \rightarrow\ name)}\ \sigma_{(learn\ \bowtie_{<sid\ =\ sid>}\ \bowtie\ (sid\ \mathcal{F}_{(count(cid)\ \rightarrow\ c\ ,\ sid)}(feedback)\ )\sigma_{(c\ =\ \mathcal{F}_{count(cid)}\sigma_{(sid\ =\ sid)}(learn)\ )\ )}(student)$

➢ SQL query :

```
select sid,fname || ' ' || lname as name from student where sid in(
        select sid from learn as l natural join(
                select sid,count(cid) as c from feedback group by sid
        ) as r1 where c=(select count(cid) from learn where sid=r1.sid)
)
```

```
select sid,fname || ' ' || lname as name from student where sid in(
    select sid from learn as l natural join(
        select sid,count(cid) as c from feedback group by sid
    ) as r1 where c=(select count(cid) from learn where sid=r1.sid)
)
```

Data Output | Explain | Messages | Notifications

	sid [PK] integer	name text
1	3	mukun...
2	4	manjal ...
3	7	shidha...
4	10	nishan ...
5	15	arishfa ...
6	19	jigisha ...
7	23	piyush ...
8	24	maulik ...
9	25	ankit si...
10	28	manav ...
11	30	aditi sh...

19) Find the list of students who have passed all the exams which they have given.

➢ Query in relational algebra :

$\Pi_{(sid\ ,\ (fname\ ||'\ '||\ name)\ \rightarrow\ name)}\ (\sigma\ (\ \bowtie_{<sid\ !=\ sid>}\ (\ \Pi_{sid}\ \sigma_{(status\ =\ 'fail'\ )}(result)\ )\ )\ )$
$(student \bowtie result\ )$

➢ SQL query :

select sid,fname || ' ' || lname as name from student natural join result

where sid not in (select sid from result where status='fail')

```
select sid,fname || ' ' || lname as name from student natural join result
where sid not in (select sid from result where status='fail')
```

Data Output | Explain | Messages | Notifications

	sid [PK] integer	name text
1	4	manjal ...
2	14	vidhi p...
3	23	piyush ...
4	5	prayag ...
5	15	arishfa ...
6	24	maulik ...
7	6	ishan p...
8	25	ankit si...
9	7	shidha...
10	17	kim jen...
11	8	paras c...
12	23	piyush ...
13	10	nishan ...
14	28	manav ...
15	29	nishita ...
16	19	jigisha ...
17	20	bhumi ...

20) Find the list of live courses which have at least 3 hours of live classes every week.

➢ Query in relational algebra :

Π_{live_id} ($\sigma_{(time*c >= '3:00:00')}$(time_table $\bowtie$ ($\Pi_{(live\_id,\ end\_time-start\_time \rightarrow time,\ c)}$ (time_table $\bowtie$ $_{live\_id}\mathcal{F}_{(\ live\_id,\ count(live\_id) \rightarrow c\ )}$(week_days)))))

➢ SQL query :

select live_id from time_table natural join (

select live_id, end_time - start_time as time, c from time_table natural join (

select live_id, count(live_id) as c from week_days group by live_id

) as r

)as t where time*c>='03:00:00'

```
select live_id from time_table natural join (
    select live_id,end_time - start_time as time,c from time_table natural join (
        select live_id,count(live_id) as c from week_days group by live_id
    )as r
)as t where time*c>='03:00:00'
```

Data Output | Explain | Messages | Notifications

	live_id character (5)	?column? interval
1	MA103	06:00:00
2	MA104	04:00:00
3	MA101	04:00:00
4	AM102	03:00:00
5	CP102	04:30:00

21) Find the list of students who have taken all the courses from the 'Applied Mathematics' domain.

- Query in relational algebra :

$\Pi_{(sid)}$ σ(t2 $\bowtie_{(sid\ !=\ sid)}$ $\Pi_{(cid\ ,\ si\ \rightarrow\ sid\ )}$(learn $\bowtie_{(cid\ =\ cid)}$ $\bowtie$ domain $\sigma_{(dname\ =\ \text{'applied mathematics'})}$) x ($\Pi_{sid\ \rightarrow\ si}$(student)) - $\Pi_{cid,sid}$ (learn)) (student)

- SQL query :

select sid from student where sid not in(

select sid from (

select cid,si as sid from (

select * from learn where cid in (

select cid from course join domain on course.domain_id = domain.did where dname = 'Applied Mathematics'))

as t2 cross join (

select sid as si from student)

as s except select cid,sid from learn

) as t3

)

```
select sid from student where sid not in(
    select sid from (
        select cid,si as sid from (
            select * from learn where cid in (
                select cid from course join domain on course.domain_id = domain.did where dname = 'Applied Mathematics'))
        as t2 cross join (
            select sid as si from student)
        as s except select cid,sid from learn
) as t3
)
```

Data Output | Explain | Messages | Notifications

sid [PK] integer

22) Find the list of students who have not taken any live course after october 2020.

- Query in relational algebra :

$\Pi_{((fname \,||\, ' \,'\, || \,lname) \rightarrow name)}$ (student $\bowtie_{(sid=sid)}$ (take $\bowtie_{(pid=pid)}$ ($\sigma_{(pay\_date < '2020\text{-}09\text{-}30')}$ (payment))))

- SQL query :

select fname ||' '|| lname as name from student where sid in (

select sid from take where pid in (

select pid from payment where pay_date < '2020-09-30')

)

```
select fname ||' '|| lname as name from student where sid in (
    select sid from take where pid in (
        select pid from payment where pay_date < '2020-09-30')
)
```

Data Output | Explain | Messages | Notifications

	name text
1	manjal shah
2	prayag patel
3	ishan patel
4	paras chhabda
5	vidhi pandya
6	arishfa hussain
7	afsana khan
8	dakota johnson
9	jigisha gujjar
10	bhumi sharma
11	piyush parmar
12	maulik patel
13	nishita patel
14	aditi shah

23) Find the total amount of payment done by students whose name starts with A.

- Query in relational algebra :

$\mathcal{F}_{(\text{sum (amount))}}(\Pi(\sigma_{(\text{fname like 'a\%'})}(\text{take} \bowtie \text{payment} \bowtie \text{student})))$

- SQL query :

select sum(amount) from (select * from take natural join payment natural join student where fname like 'a%') as t1

```
select sum(amount) from (
    select * from take natural join payment natural join student where fname like 'a%'
) as t1
```

Data Output | Explain | Messages | Notifications

	sum bigint
1	76000

24) Give the list of courses in their descending order of rating.

➢ Query in relational algebra :

$\Pi_{(\text{rating (desc)})}(\Pi_{(\text{cid,rating})}(\text{course}))$

➢ SQL query :

select cid, rating from course order by rating desc

```
select cid, rating from course order by rating desc
```

Data Output | Explain | Messages | Notifications

	cid [PK] character (5)	rating numeric (3,2)
1	CP102	5.00
2	MA103	4.50
3	CP103	4.00
4	MA102	4.00
5	AM101	3.67
6	MA104	3.50
7	CP101	3.00
8	MA101	3.00
9	AM102	2.00
10	WD101	0.00

25) Find the list of companies which have provided only live courses and not recorded.

- Query in relational algebra :

  Π(colab_company $\bowtie_{(comp\_name <> company\_name)}$ ($\sigma_{(company\_name='Platform' \text{ or } type\_of\_course='Record')}$(educator as e $\bowtie_{(e.eid=c.eid)}$ course as c)))

- SQL query :

```
select * from colab_company where comp_name not in (

select company_name from educator as e join course as c on
e.eid=c.eid where company_name = 'Platform' or type_of_course =
'Record'

)
```

```
select * from colab_company where comp_name not in (
    select company_name from educator natural join course
    where company_name = 'Platform' or type_of_course = 'Record'
)
```

Data Output | Explain | Messages | Notifications

	comp_name [PK] character varying (30)
1	Google
2	Microsoft
3	Oracle