به نام خدا

پاسخنامه تکلیف شماره سه

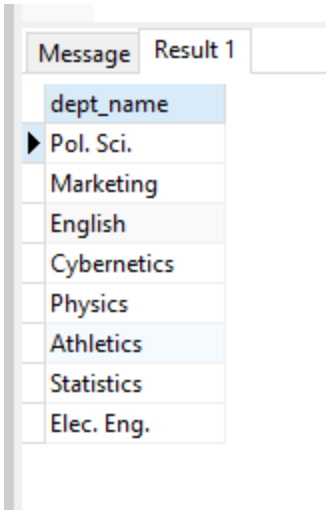
پایگاه داده ترم پاییز 1400

- **آشنایی با WITH و CRUD**

1. برای هر یک از موارد زیر بنویسید که چه نتیجه ای دارند و WITH را به SUB QUERY و برعکس تبدیل کنید.
   a. تبدیل از WITH به SUB QUERY و توضیح خروجی کوئری

```
SELECT dept_name FROM
(SELECT dept_name, sum(salary) FROM instructor GROUP BY dept_name) as
dept_total(dept_name,value) ,
(SELECT avg(value) FROM (select dept_name, sum(salary) FROM instructor GROUP
BY dept_name) as dept_total(dept_name,value)) as dept_total_avg(value)
WHERE dept_total.value >= dept_total_avg.value;
```

کوئری مربوطه در واقع جمع حقوق که به اساتید تعلق میگیرد به تفکیک دپارتمان محاسبه کرده،سپس میانگین این حقوق ها را محاسبه کرده، در نهایت کوئری کلی دپارتمان هایی را می آورد که جمع حقوق پرداختی آن دپارتمان ها به اساتیدشان از میانگین حقوق پرداختی دپارتمان ها به استاد ها بیشتر باشد.



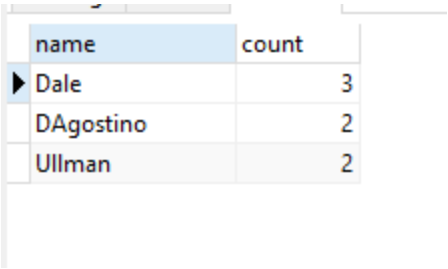
b. تبدیل از SUB QUERY به WITH و توضیح خروجی کوئری

```
WITH InsTeachAvg(TCAvg) AS( SELECT AVG(InsTeachCount.TeachCount)
            FROM (SELECT instructor.id,COUNT(*) AS TeachCount
                  FROM teaches,instructor
                 WHERE teaches.id=instructor.id AND year=2003
                      GROUP BY instructor.id,year) AS InsTeachCount )

SELECT instructor.name,COUNT(*)
```

```
FROM teaches,instructor
WHERE teaches.id=instructor.id AND year=2003
GROUP BY instructor.id,year
HAVING COUNT(*)>(SELECT TCAvg FROM InsTeachAvg)
```

نام استاد به همراه تعداد دروس ارائه شده آن در سال 2003 را برای اساتیدی که در سال 2003 از میانگین تعداد دروس ارائه شده توسط هر استاد در آن سال تعداد درس های بیشتری ارائه داده باشند را لیست می کند.



name	count
Dale	3
DAgostino	2
Ullman	2

2. با توجه به پایگاه داده معرفی شده در کلاس (University: Large Version) موارد زیر را به ترتیب با کمک CRUD انجام دهید
   a. جدولی با اطلاعات

uni_data(stu_id, stu_name,stu_dept_name, year,semester,course_name, score, is_rank)

بسازید. نوع فیلد ها با توجه فیلد های دیتابیس انتخاب کنید.

```
CREATE TABLE "public"."uni_data" (
  "stu_id" varchar(5),
  "stu_name" varchar(20),
  "stu_dept_name" varchar(255),
  "year" numeric(4),
  "semester" varchar(6),
  "course_name" varchar(50),
  "score" int4,
  "is_rank" int2
);
```

```
CREATE TABLE "public"."uni_data" (
  "stu_id" varchar(5) NOT NULL,
  "stu_name" varchar(20),
  "stu_dept_name" varchar(255),
  "year" numeric(4),
  "semester" varchar(6),
  "course_name" varchar(50),
  "score" int4,
  "is_rank" int2
)
> OK
> Time: 0.039s
```

b. با توجه به جدولی که در قسمت قبل ساختید کوئری بنویسید که داده های دانشجو ها و درس هایی که برداشته اند را در جدول وارد کند و score دانشجو را مطابق grade زیر ثبت کند.

A+ = 100 , A = 95, A- = 90

B+ = 85, B = 80, B- = 75

C+ = 70, C = 65, C- = 60

Else: 0

و مقدار ستون is_rank اگر امتیاز شخصی بزرگتر از 70 بود یک باشد و در غیر اینصورت 0

```
INSERT INTO uni_data
SELECT s."id", s."name",s.dept_name,T."year",T.semester,C.title,
(CASE
      WHEN T.grade = 'A+' THEN 100
      WHEN T.grade = 'A ' THEN 95
      WHEN T.grade = 'A-' THEN 90
      WHEN T.grade = 'B+' THEN 85
      WHEN T.grade = 'B ' THEN 80
      WHEN T.grade = 'B-' THEN 75
      WHEN T.grade = 'C+' THEN 70
      WHEN T.grade = 'C ' THEN 65
      WHEN T.grade = 'C-' THEN 60 ELSE 0 END ),
    ( CASE WHEN T.grade IN ( 'A+', 'A ', 'A-', 'B+', 'B ', 'B-' ) THEN 1 ELSE
0 END )
    FROM takes AS T JOIN student AS s ON T."id" = s."id"
    JOIN course AS C ON C.course_id = T.course_id
```

```
INSERT INTO uni_data
SELECT s."id", s."name",s.dept_name,T."year",T.semester,C.title,
(CASE
      WHEN T.grade = 'A+' THEN 100
      WHEN T.grade = 'A ' THEN 95
      WHEN T.grade = 'A-' THEN 90
      WHEN T.grade = 'B+' THEN 85
      WHEN T.grade = 'B ' THEN 80
      WHEN T.grade = 'B-' THEN 75
      WHEN T.grade = 'C+' THEN 70
      WHEN T.grade = 'C ' THEN 65
      WHEN T.grade = 'C-' THEN 60 ELSE 0 END ),
    ( CASE WHEN T.grade IN ( 'A+', 'A ', 'A-', 'B+', 'B ', 'B-' ) THEN 1 ELSE 0 END )
    FROM takes AS T JOIN student AS s ON T."id" = s."id"
    JOIN course AS C ON C.course_id = T.course_id
> Affected rows: 30000
> Time: 0.868s
```

c. کوئری بنویسید که امتیاز درس های دانشجویان دانشکده فیزیک را در جدولی که ساخته اید، اگر کوچکتر از 75 بود 10 امتیاز و در غیر اینصورت 15 امتیاز افزایش دهد.

```
UPDATE uni_data
SET score = ( CASE WHEN score < 75 THEN score + 10 ELSE score + 15 END )
WHERE
  uni_data.stu_dept_name = 'Physics'
```

Message

```
UPDATE uni_data
SET score = ( CASE WHEN score < 75 THEN score + 10 ELSE score + 15 END )
WHERE
  uni_data.stu_dept_name = 'Physics'
> Affected rows: 1416
> Time: 0.157s
```

d. کوئری بنویسید که رکورد هایی که اسم دانشجو با T شروع می شوند و امتیازشان از میانگین امتیاز ها در آن دانشکده کمتر است حذف شوند.

```
DELETE FROM uni_data AS ud
WHERE
  stu_name LIKE 'T%'
  AND score < ( SELECT AVG ( score ) FROM uni_data AS sub_ud WHERE
sub_ud.stu_dept_name = ud.stu_dept_name)
```

```
DELETE FROM uni_data AS ud
WHERE
  stu_name LIKE 'T%'
  AND score < ( SELECT AVG ( score ) FROM uni_data AS sub_ud WHERE sub_ud.stu_dept_name = ud.stu_dept_name)
> Affected rows: 557
> Time: 6.948s
```

- **آشنایی با مفاهیم انواع JOIN**

3. جداول

users(user_id, name)
numbers(user_id, phone_number)

در نظر بگیرید. فیلد هایی که زیرشان خط کشیده شده است کلید هستند.
جدول کاربران شامل 100 رکورد یکتا است و جدول شماره ها نیز شامل 100 رکورد یکتا است. هر کاربر می تواند چند شماره داشته باشد و یک شماره نمی تواند متعلق به چند کاربر باشد.ممکن است کاربری وجود داشته باشد که شماره تلفن ندارد اما شماره تلفن بدون کاربر ذخیره نمی شود.
با توجه به توضیحات گفته شده در جدول زیر وارد کنید که به ازای هر کدام از JOIN های گفته شده،کمترین و بیشترین تعداد خروجی ممکن چقدر خواهد بود.

	کمترین تعداد خروجی حاصل	بیشترین تعداد خروجی حاصل
users INNER JOIN numbers	100	100
users LEFT OUTER JOIN numbers	100	199
users RIGHT OUTER JOIN numbers	100	100
users FULL OUTER JOIN numbers	100	199

4. هرکدام از کوئری های زیر را بدون استفاده از امکانات (OUTER JOIN , LEFT, RIGHT) پیاده سازی کنید

a.

```
SELECT * FROM course NATURAL LEFT OUTER JOIN section
```

```
(select * from course natural join section)
union
(select course_id, title, dept_name, credits, NULL, NULL, NULL, NULL, NULL,
NULL
from course c1
where not exists
(select course_id from section s1 where c1.course_id = s1.course_id))
```

course_id	title	dept_name	credits	sec_id	semester	year	building	room_number	time_slot_id	numa
893	Systems Software	Cybernetics	3	1	Fall	2007	Fairchild	145	B	83
974	Astronautics	Accounting	3	1	Fall	2003	Polya	808	H	104
959	Bacteriology	Physics	4	1	Fall	2006	Saucon	180	M	82
345	Race Car Driving	Accounting	4	1	Spring	2008	Taylor	183	A	79
324	Ponzi Schemes	Civil Eng.	3	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
781	Compiler Design	Finance	4	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
200	The Music of the Ramones	Accounting	4	2	Fall	2002	Chandler	375	D	81
642	Video Gaming	Psychology	3	1	Fall	2004	Saucon	113	D	91
209	International Trade	Cybernetics	4	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
864	Heat Transfer	Geology	3	1	Spring	2006	Power	972	D	94
857	UNIX System Programmmin	Geology	4	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
468	Fractal Geometry	Civil Eng.	4	2	Fall	2007	Power	717	L	103
571	Plastics	Comp. Sci.	4	1	Spring	2004	Power	972	I	95
457	Systems Software	History	3	1	Spring	2001	Saucon	844	D	83
922	Microeconomics	Finance	4	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)

(select * from course natural join section) union (select course_id, title, dept_name, credits, NULL, NULL, NULL, NU | Read Only | Query time: 0.007s | Record 1 of 215

.b

```
SELECT * FROM course NATURAL RIGHT OUTER JOIN section

(select * from course natural join section)
union
(select course_id, NULL, NULL, NULL, sec_id, semester, year, building,
room_number, time_slot_id
from section s1
where not exists
(select course_id from course c1 where c1.course_id = s1.course_id));
```

course_id	title	dept_name	credits	sec_id	semester	year	building	room_number	time_slot_id
735	Greek Tragedy	Geology	3	2	Spring	2010	Taylor	183	D
599	Mechanics	Psychology	4	1	Spring	2003	Chandler	804	D
991	Transaction Processing	Psychology	3	1	Spring	2008	Lamberton	134	J
482	FOCAL Programming	Psychology	4	1	Fall	2005	Whitman	434	H
338	Graph Theory	Psychology	3	1	Spring	2007	Fairchild	145	G
735	Greek Tragedy	Geology	3	1	Spring	2003	Drown	757	D
867	The IBM 360 Architecture	History	3	2	Fall	2010	Lamberton	134	M
400	Visual BASIC	Psychology	4	1	Spring	2007	Lambeau	348	M
408	Bankruptcy	Accounting	3	1	Spring	2007	Taylor	812	C
642	Video Gaming	Psychology	3	1	Fall	2004	Saucon	113	D
962	Animal Behavior	Psychology	3	1	Spring	2008	Nassau	45	L
349	Networking	Finance	4	1	Spring	2008	Saucon	113	K
852	World History	Athletics	4	1	Spring	2008	Gates	707	F
445	Biostatistics	Finance	3	1	Spring	2001	Alumni	547	J
476	International Communicatic	Astronomy	4	1	Fall	2010	Drown	757	C

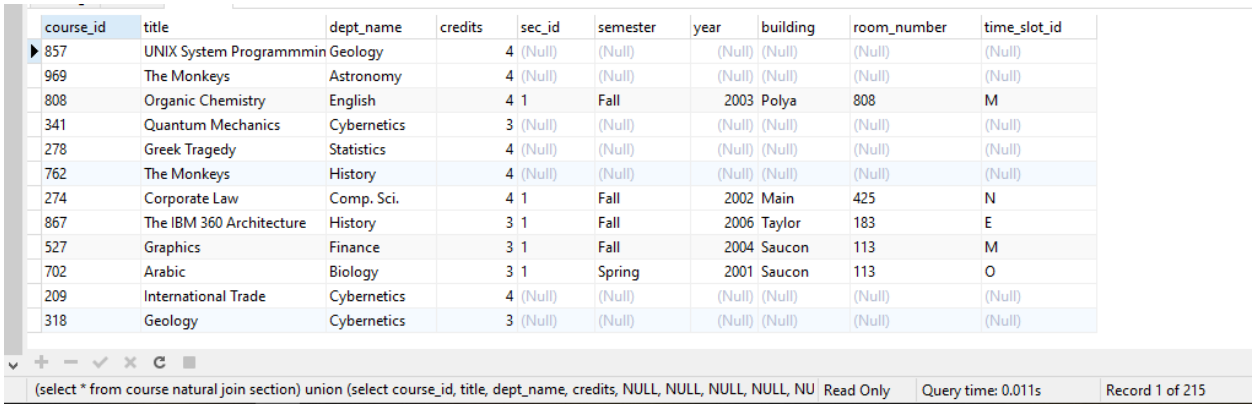
(select * from course natural join section) union (select course_id, NULL, NULL, NULL, sec_id, semester, year, building, roo | Read Only | Query time: 0.006s | Record 1 of 100

.c

```
SELECT * FROM course NATURAL FULL OUTER JOIN section


(select * from course natural join section)
union
(select course_id, title, dept_name, credits, NULL, NULL, NULL, NULL, NULL,
NULL
from course c1
where not exists
```

```
(select course_id from section s1 where c1.course_id = s1.course_id))
union
(select course_id, NULL, NULL, NULL, sec_id, semester, year, building,
room_number, time_slot_id
from section s1
where not exists
(select course_id from course c1 where c1.course_id = s1.course_id));
```



course_id	title	dept_name	credits	sec_id	semester	year	building	room_number	time_slot_id
857	UNIX System Programmmin	Geology	4	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
969	The Monkeys	Astronomy	4	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
808	Organic Chemistry	English	4	1	Fall	2003	Polya	808	M
341	Quantum Mechanics	Cybernetics	3	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
278	Greek Tragedy	Statistics	4	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
762	The Monkeys	History	4	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
274	Corporate Law	Comp. Sci.	4	1	Fall	2002	Main	425	N
867	The IBM 360 Architecture	History	3	1	Fall	2006	Taylor	183	E
527	Graphics	Finance	3	1	Fall	2004	Saucon	113	M
702	Arabic	Biology	3	1	Spring	2001	Saucon	113	O
209	International Trade	Cybernetics	4	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)
318	Geology	Cybernetics	3	(Null)	(Null)	(Null)	(Null)	(Null)	(Null)

- **آشنایی با کوئری های عملی JOIN**

5. با توجه به پایگاه داده معرفی شده در کلاس (AdventureWorks) موارد زیر را انجام دهید

a. لیست نام تمام محصولات به همراه شناسه سفارش حتی اگر سفارشی برای آن محصول ثبت نشده باشد.

```
SELECT p.name  , sd.salesorderid
FROM production.product as p
LEFT JOIN sales.salesorderdetail as sd ON p.productid = sd.productid
```

name	salesorderid
Mountain-100 Black, 42	43659
Mountain-100 Black, 44	43659
Mountain-100 Black, 48	43659
Mountain-100 Silver, 38	43659
Mountain-100 Silver, 42	43659
Mountain-100 Silver, 44	43659
Mountain-100 Silver, 48	43659
Long-Sleeve Logo Jersey, M	43659
Long-Sleeve Logo Jersey, XL	43659
Mountain Bike Socks, M	43659
AWC Logo Cap	43659
Sport-100 Helmet, Blue	43659
Road-650 Red, 44	43660
Road-450 Red, 52	43660
HL Mountain Frame - Black, 48	43661

b. لیست نام تمام محصولاتی را بیاورید که هرگز برای آن ها سفارشی ثبت نشده است.

```
SELECT p.name
FROM production.product as p
LEFT JOIN sales.salesorderdetail as sd ON p.productid = sd.productid AND
sd.salesorderid is null
```

name
Adjustable Race
Bearing Ball
BB Ball Bearing
Headset Ball Bearings
Blade
LL Crankarm
ML Crankarm
HL Crankarm
Chainring Bolts
Chainring Nut
Chainring
Crown Race
Chain Stays
Decal 1
Decal 2

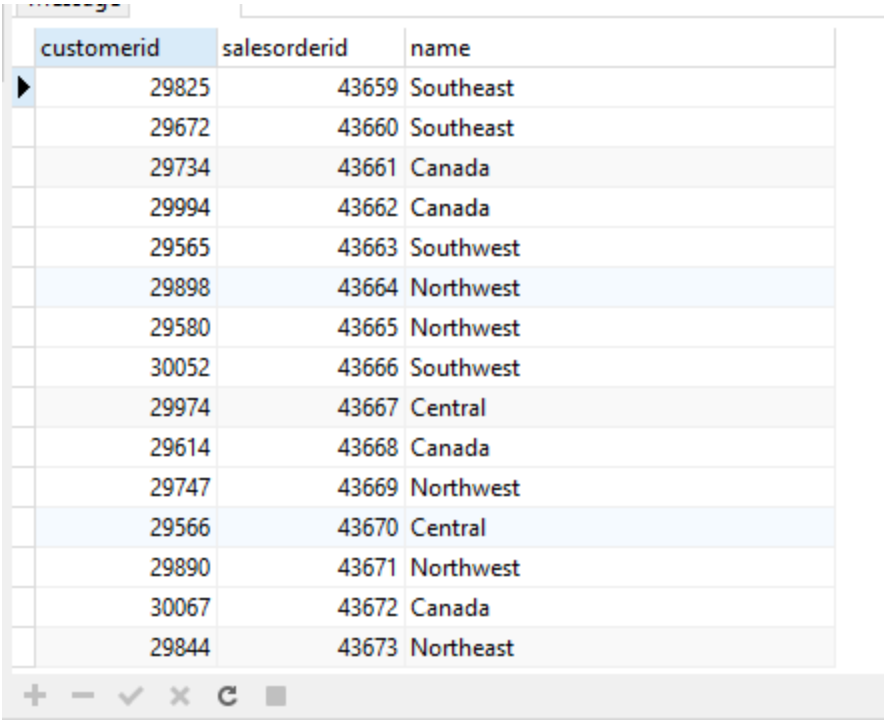
c. لیست شناسه مشتری و نام کالاهایی که مشتری خریده است.

```
SELECT
  c.customerid,
  p.name
FROM sales.customer as c
JOIN sales.salesorderheader as soh ON soh.customerid = c.customerid
JOIN sales.salesorderdetail as sod ON sod.salesorderid = soh.salesorderid
JOIN production.product as p ON p.productid  = sod.productid
```

customerid	name
29825	Mountain-100 Black, 42
29825	Mountain-100 Black, 44
29825	Mountain-100 Black, 48
29825	Mountain-100 Silver, 38
29825	Mountain-100 Silver, 42
29825	Mountain-100 Silver, 44
29825	Mountain-100 Silver, 48
29825	Long-Sleeve Logo Jersey, M
29825	Long-Sleeve Logo Jersey, XL
29825	Mountain Bike Socks, M
29825	AWC Logo Cap
29825	Sport-100 Helmet, Blue
29672	Road-650 Red, 44
29672	Road-450 Red, 52
29734	HL Mountain Frame - Black, 48

d. لیست شناسه مشتری و شناسه سفارش مشتری و نام منطقه مشتری(territory) حتی اگر مشتری سفارشی ثبت نکرده باشد.(با کمک RIGHT JOIN)

```
SELECT
  c.customerid,
  soh.salesorderid,
    st."name"
FROM sales.salesorderheader as soh
RIGHT JOIN sales.customer as c ON soh.customerid = c.customerid
JOIN sales.salesterritory as st ON st.territoryid = c.territoryid
```



customerid	salesorderid	name
29825	43659	Southeast
29672	43660	Southeast
29734	43661	Canada
29994	43662	Canada
29565	43663	Southwest
29898	43664	Northwest
29580	43665	Northwest
30052	43666	Southwest
29974	43667	Central
29614	43668	Canada
29747	43669	Northwest
29566	43670	Central
29890	43671	Northwest
30067	43672	Canada
29844	43673	Northeast

- **آشنایی با VIEW و ایجاد آن**

6. یک view طراحی کنید که در آن مجموع تعداد کالاهای سفارش داده شده در سفارش کارها(workorder) که در فرایند سفارش خود به مرحله نهایی یعنی مرحله 7 نرسیده اند و کامل نشده اند را به تفکیک زیر دسته به همراه شناسه و نام آن زیردسته را در 1000 سفارش کار آخر (بر اساس enddate و به صورت نزولی)محاسبه و به ترتیب مجموع تعداد کالای سفارش داده شده در هر دسته به صورت نزولی نمایش دهد. در نهایت از ویو ساخته شده کوئری بگیرید.

```
create view subCategoryWorkItemSum as
select wos.productsubcategoryid,(select name from
production.productsubcategory as psc1
                                 where
wos.productsubcategoryid=psc1.productsubcategoryid) as name ,sum(orderqty)
from (select p.productid,psc.productsubcategoryid,wo.orderqty from
production.workorder wo
inner join production.product as p on p.productid=wo.productid
inner join production.productsubcategory as psc on
psc.productsubcategoryid=p.productsubcategoryid
where not exists (select * from production.workorderrouting as wor where
wor.workorderid=wo.workorderid and wor.productid=wo.productid and
wor.OperationSequence=7)
order by wo.enddate desc
limit 1000) as wos
group by wos.productsubcategoryid
order by sum desc
```

```
create view subCategoryWorkItemSum as
select wos.productsubcategoryid,(select name from production.productsubcategory as psc1
                                 where wos.productsubcategoryid=psc1.productsubcategoryid) as name ,sum(orderqty)
from (select p.productid,psc.productsubcategoryid,wo.orderqty from production.workorder wo
inner join production.product as p on p.productid=wo.productid
inner join production.productsubcategory as psc on psc.productsubcategoryid=p.productsubcategoryid
where not exists (select * from production.workorderrouting as wor where wor.workorderid=wo.workorderid and wor.productid=wo.productid and
wor.OperationSequence=7)
order by wo.enddate desc
limit 1000) as wos
group by wos.productsubcategoryid
order by sum desc
> OK
> Time: 0.033s
```

productsubcategoryid	name	sum
9	Derailleurs	1812
17	Wheels	1682
10	Forks	873
5	Bottom Brackets	868
8	Cranksets	868
4	Handlebars	856
11	Headsets	847
14	Road Frames	230
12	Mountain Frames	158
16	Touring Frames	152

7. یک view طراحی کنید که در آن لیست دانشجویان و اساتید با فیلد های شناسه دانشجو یا استاد،نام دانشجو یا استاد و نوع (اگر در اسم دانشکده Eng باشد نوع Engineer و در غیر این صورت Scientist) و نوع شخص(اگر استاد بود INS و اگر دانشجو STU) را نمایش دهد.

```
CREATE VIEW "view_stu_ins" AS (SELECT id,name,
(CASE WHEN dept_name LIKE '%Eng%' THEN 'Engineer' ELSE 'Scientist' END) as
dept_type,
'STU' as person_type FROM student )
UNION
(SELECT id,name,
(CASE WHEN dept_name LIKE '%Eng%' THEN 'Engineer' ELSE 'Scientist' END) as
dept_type,
'INS' as person_type FROM instructor);
```

```
CREATE VIEW "view_stu_ins" AS (SELECT id,name,
(CASE WHEN dept_name LIKE '%Eng%' THEN 'Engineer' ELSE 'Scientist' END) as dept_type,
'STU' as person_type FROM student )
UNION
(SELECT id,name,
(CASE WHEN dept_name LIKE '%Eng%' THEN 'Engineer' ELSE 'Scientist' END) as dept_type,
'INS' as person_type FROM instructor)
> OK
> Time: 0.014s
```

id	name	dept_type	person_type
96153	Sawah	Scientist	STU
37284	Benabd	Scientist	STU
82402	Grant	Scientist	STU
41596	Abeggl	Scientist	STU
58413	Xiong	Scientist	STU
19603	Colu	Scientist	STU
68554	Larsson	Engineer	STU
87193	Pinkus	Engineer	STU
48660	Emam	Scientist	STU
9933	Pircher	Scientist	STU
51416	Shan	Scientist	STU
94697	Pettersen	Scientist	STU
32119	Nagashima	Engineer	STU
38545	Fok	Engineer	STU
61356	Vulp	Scientist	STU

8. با استفاده از view که در سوال قبل ایجاد کردید دستوری بزنید که برای هر استاد بگوید چند درصد بودجه دانشکده اش به او تعلق می گیرد و برای هر دانشجو بگوید چه میزان از بودجه دانشکده اش به او تعلق می گیرد.

خروجی مورد انتظار: name, person_type, calc_number

```
(SELECT v."name" ,v.person_type, (i.salary / d.budget * 100) as calc_number
from view_stu_ins as v
JOIN instructor as i ON i."id" = v."id"
JOIN department as d ON d.dept_name = i.dept_name
WHERE v.person_type = 'INS')
UNION

(SELECT v."name" , v.person_type,( d.budget / (SELECT COUNT(*) FROM student
WHERE dept_name = s.dept_name)) as calc_number
from view_stu_ins as v
JOIN student as s ON s."id" = v."id"
JOIN department as d ON d.dept_name = s.dept_name
WHERE v.person_type = 'STU')
```

name	person_type	calc_number
Kahs	STU	4647.6675294117647059
Shuming	INS	11.46424106170360628600
Jordan	STU	8545.1110989010989011
Stylian	STU	5052.8033613445378151
Yoshimoto	STU	9238.8541860465116279
Gleit	STU	5975.5629059829059829
Papakir	STU	5975.5629059829059829
Benkov	STU	4463.0395959595959596
Noda	STU	4463.0395959595959596
Morton	STU	984.9878703703703704
So	STU	7984.2467391304347826
Chowdhury	STU	9814.1954166666666667
Crimm	STU	984.9878703703703704
Yüksel	STU	6432.0280000000000000
Kok	STU	4419.1079347826086957