Exercise 1

- Considering the relational model of the university, answer the following queries in relational algebra and in sql:

  - Find the titles of courses in the Comp. Sci. department that have 3 credits.
  - Find the highest salary of any instructor.
  - Find all instructors earning the highest salary (there may be more than one with the same salary).
  - Find the enrollment of each section that was offered in Autumn 2009.
  - Find the maximum enrollment, across all sections, in Autumn 2009.
  - Find the sections that had the maximum enrollment in Autumn 2009.

Exercise 1

- Find the titles of courses in the Comp. Sci. department that have 3 credits.

$$\Pi_{title}(\sigma_{dept\_name\ =\ \text{'Comp. Sci'}\ \wedge\ credits=3}(course))$$

```
select  title
from    course
where   dept_name = 'Comp. Sci.'
        and credits = 3
```

Exercise 1

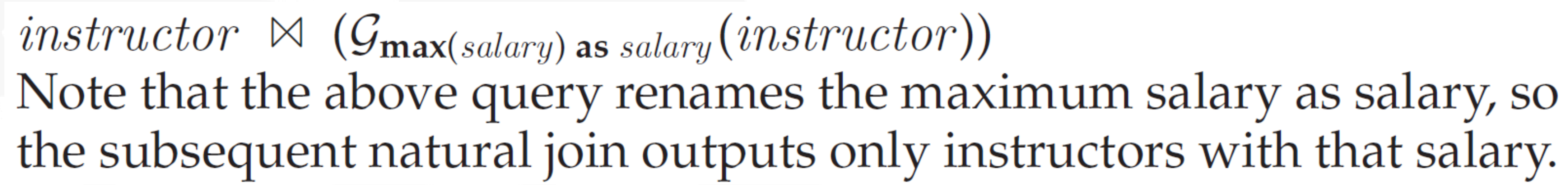
- Find the highest salary of any instructor.

$$\mathcal{G}_{\mathbf{max}(salary)}(instructor)$$

```
select max(salary)
from   instructor
```

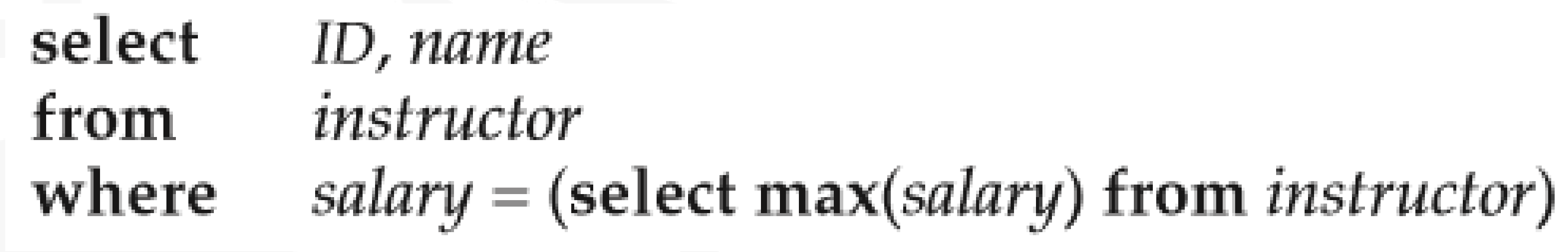
Exercise 1

- Find all instructors earning the highest salary (there may be more than one with the same salary).



$$instructor \bowtie (\mathcal{G}_{\mathbf{max}(salary)\ \mathbf{as}\ salary}(instructor))$$

Note that the above query renames the maximum salary as salary, so the subsequent natural join outputs only instructors with that salary.



```
select    ID, name
from      instructor
where     salary = (select max(salary) from instructor)
```

Exercise 1

- Find the enrollment of each section that was offered in Autumn 2009.

$$_{course\_id,\, section\_id}\mathcal{G}_{\textbf{count}(*)\ \textbf{as}\ enrollment}(\sigma_{year=2009 \land semester=\text{Autumn}}(takes))$$

```
select    course_id, sec_id, count(ID)
from      section natural join takes
where     semester = 'Autumn'
and       year = 2009
group by  course_id, sec_id
```

Exercise 1

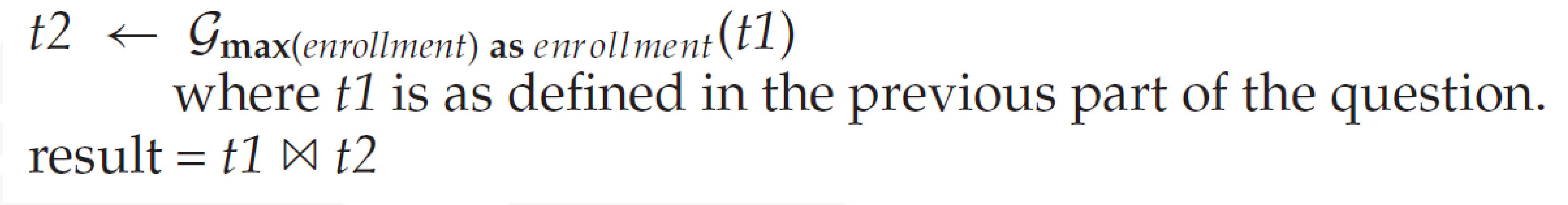
- Find the maximum enrollment, across all sections, in Autumn 2009.

$$t1 \leftarrow {}_{course\_id,\, section\_id}\mathcal{G}_{\textbf{count}(*)\ \textbf{as}\ enrollment}(\sigma_{year=2009 \wedge semester=\text{Autumn}}(takes))$$
$$\text{result} = \mathcal{G}_{\textbf{max}(enrollment)}(t1)$$

```
select   max(enrollment)
from     (select      count(ID) as enrollment
          from        section natural join takes
          where       semester = 'Autumn'
          and         year = 2009
          group by course_id, sec_id)
```

Exercise 1

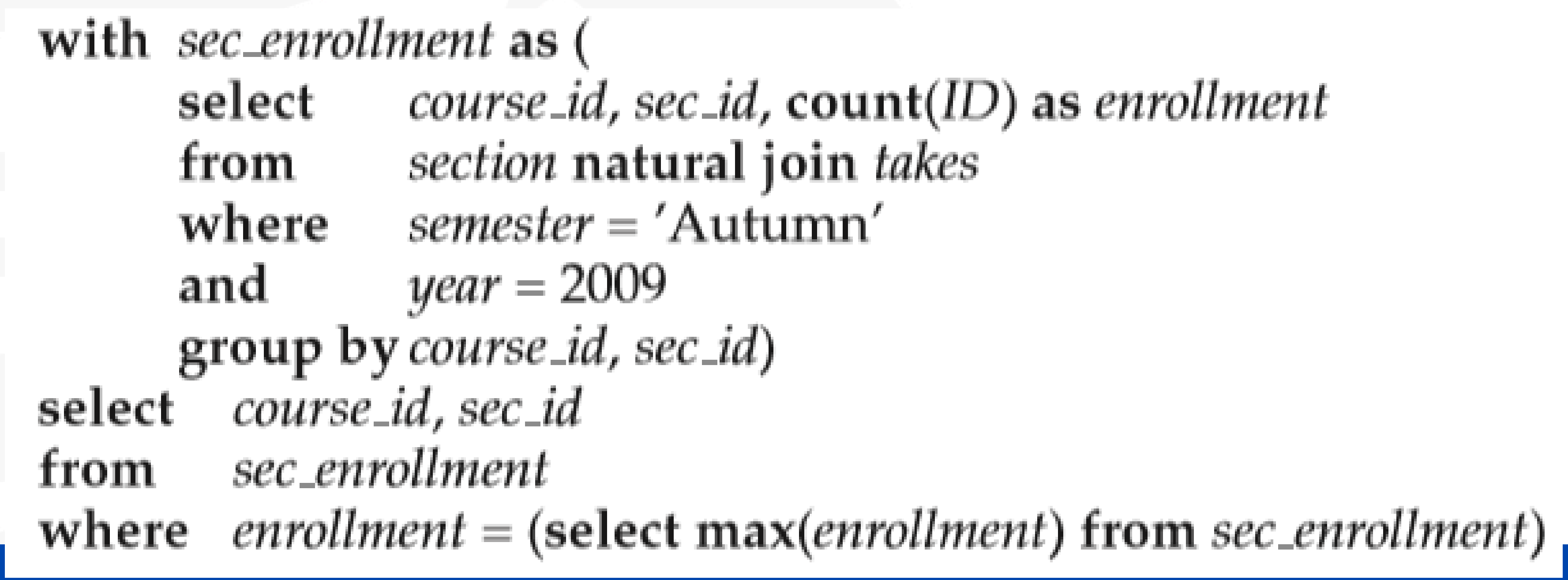
- Find the sections that had the maximum enrollment in Autumn 2009.



$$t2 \leftarrow \mathcal{G}_{\mathbf{max}(enrollment)\ \mathbf{as}\ enrollment}(t1)$$

where $t1$ is as defined in the previous part of the question.

result = $t1 \bowtie t2$



```
with  sec_enrollment as (
      select      course_id, sec_id, count(ID) as enrollment
      from        section natural join takes
      where       semester = 'Autumn'
      and         year = 2009
      group by course_id, sec_id)
select   course_id, sec_id
from     sec_enrollment
where    enrollment = (select max(enrollment) from sec_enrollment)
```

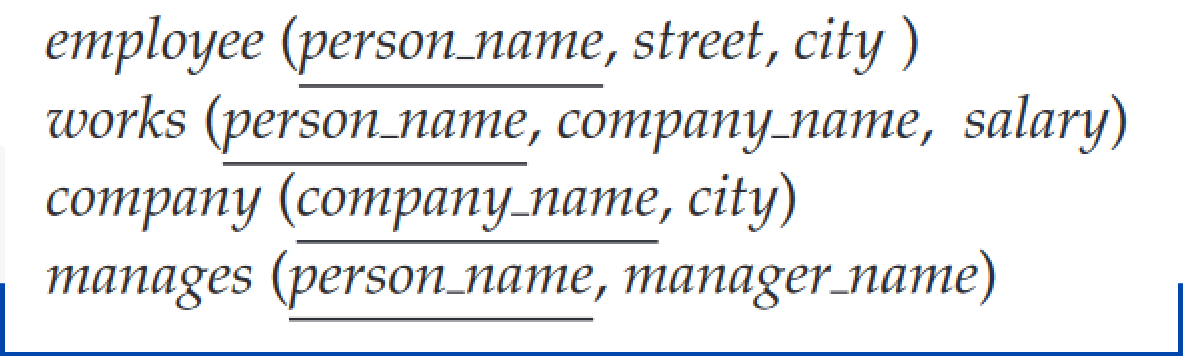
Exercise 2

Consider the relational database of Figure 6.22, where the primary keys are underlined. Give an expression in the relational algebra to express each of the following queries:

a. Find the names of all employees who live in the same city and on the same street as do their managers.

b. Find the names of all employees in this database who do not work for “First Bank Corporation”.

c. Find the names of all employees who earn more than every employee of “Small Bank Corporation”.



$\textit{employee}\ (\underline{\textit{person\_name}}, \textit{street}, \textit{city}\ )$

$\textit{works}\ (\underline{\textit{person\_name}}, \textit{company\_name}, \ \textit{salary})$

$\textit{company}\ (\underline{\textit{company\_name}}, \textit{city})$

$\textit{manages}\ (\underline{\textit{person\_name}}, \textit{manager\_name})$

Exercise 2

- Find the names of all employees who live in the same city and on the same street as do their managers

$$\Pi_{person\_name}((employee \bowtie manages) \bowtie_{(manager\_name = employee2.person\_name \wedge employee.street = employee2.street \wedge employee.city = employee2.city)} (\rho_{employee2}(employee)))$$

Select E1.person_name from (Employee as E1 natural inner join Manages as M) inner join Employee as E2 on M.manager_name=E1.person_name and E1.street=E2.street and E1.city=E2.city

Exercise 2

- Find the names of all employees in this database who do not work for "First Bank Corporation".

$$\Pi_{person\_name}\ (\sigma_{company\_name\ \neq\ \text{"First Bank Corporation"}}(works))$$

If people may not work for any company:

$$\Pi_{person\_name}(employee)\ -\ \Pi_{person\_name}$$
$$(\sigma_{(company\_name\ =\ \text{"First Bank Corporation"})}(works))$$

Select person_name from Works where company_name<>'First Bank Corporation'

Select person_name from Employee except select person_name from Works where company_name='First Bank Corporation'

Exercise 2

- Find the names of all employees who earn more than every employee of "Small Bank Corporation".

$$\Pi_{person\_name}(works) - (\Pi_{works.person\_name}(works \bowtie_{(works.salary \leq works2.salary \land works2.company\_name = \text{"Small Bank Corporation"})} \rho_{works2}(works)))$$

Select person_name from works except

Select works.person_name from works inner join works as works2 on works.salary<=works2.salary and works2.company_name='Small Bank Corporation'

Exercise 3

- Considering the following relational model (PKs are underlined):
  - Employee (ssn, name, surname, dateOfBirth, address, supervisorSSN, nD)
  - Department (numberD, named, managerSSN, initDateManager)
  - Places (numberD, placeD)
  - WorksIn (ssnE, np, hours)
  - Project (numberP, nameP, place, nD)
  - Dependent (ssnE, dependentName, dateOfBirth, relationship)

- Answer the following queries in relational algebra and in sql.

Exercise 3

1. Show the name and address of every employee who works in the "Energy" department.

Exercise 3

1. Show the name and address of every employee who works in the "Energy" department.

```
Select name, address
From Employee inner join Department on numberD=nD
Where nameD="Energy"
```

$\Pi_{\text{name, address}}(\sigma_{\text{nameD=“Energy”}}(\text{Employee } x_{\text{nD=numberD}} \text{ Department}))$

x: inner join

Exercise 3

2. For each project located in Santiago, show the number of the project, the number of the deparment that controls the project, and the surname, address and date of birth of the department's manager.

Exercise 3

2. For each project located in Santiago, show the number of the project, the number of the deparment that controls the project, and the surname, address and date of birth of the department's manager.

Select numberP, numberD, surname, address, dateOfBirth

From (Employee inner join Department on ssn=managerSSN) inner join Project on numberD=nD

Where place="Santiago"

$\Pi_{\text{numberP, numberD, surname, address, dateOfBirth}} (\sigma_{\text{place=“Santiago”}}$
$(\text{Employee } \text{x}_{\text{ssn=managerSSN}} \text{ Department } \text{x}_{\text{nD=numberD}} \text{ Project}))$

Exercise 3

3. Show the name of the employees who work in **every** project controlled by the department number 5.

Exercise 3

3. Show the name of the employees who work in **every** project controlled by the department number 5.

```
Select E.name from Employee as E
Where not exists (( select P.numberP from Project as P
                         where P.nD=5) except
                         (select W.nP from WorksIn as W
                         where E.ssn=W.ssnE))
```

Temp1 $\leftarrow \Pi_{ssnE,\ nP}$ (WorksIn)

Temp2 $\leftarrow \Pi_{numberP}(\sigma_{nrD=5}$(Project))

Temp3 $\leftarrow$ Temp1 $\div$ Temp2

Π_{name} (Temp3 $\times_{ssn=ssnE}$ Employee)

Exercise 3

4. Show a list with the numbers of the projects in which an employee whose surname is ‘Smith’ works as a normal employee or as the manager of the department that controls the project.

Exercise 3

4. Show a list with the numbers of the projects in which an employee whose surname is 'Smith' works as a normal employee or as the manager of the department that controls the project.

Select nP from WorksIn inner join Employee on ssn=ssnE

Where surname="Smith"

Union

Select nP from (Department inner join Employee on ssn=managerSSN) inner join Project on numberD=nD

Temp1 $\leftarrow \Pi_{nP} (\sigma_{surname=\text{“Smith”}}(\text{WoksIn } x_{ssn=ssnE} \text{ Employee}))$

Temp2 $\leftarrow \Pi_{nP} (\sigma_{surname=\text{“Smith”}}((\text{Department } x_{ssn=managerSSN} \text{ Employee}) \text{ x }_{nD=numberD} (\text{Project}))$

Temp1 U Temp2

Exercise 3

5. Show a list with the names of the employees who have two or more dependents

Exercise 3

5. Show a list with the names of the employees who have two or more dependents

```
Select name, count(dependentName)
From Dependent inner join Employee on ssn=ssnE
Group by name
Having count(dependentName)>=2
```

$\text{Temp1} \leftarrow {}_{\text{nssE}}\, G_{\text{count(dependentName) as n}} (\text{Dependent})$

$\Pi_{\text{name}} (\sigma_{n>=2}(\text{Temp1}) \times_{\text{ssn=ssnE}} \text{Employee})$

Exercise 3

6. Show the name of employees who don’t have dependents.

Exercise 3

6. Show the name of employees who don’t have dependents.

```
Select name from Employee
Where name not in (select name from
                          Employee inner join Dependent on ssn=ssnE)
```

$Temp1 \leftarrow \Pi_{ssn}(Employee) - \Pi_{ssnE}(Dependent)$

$\Pi_{name}(Temp1 \times Employee)$

Exercise 3

7. Show the name of the managers who have at least one dependent.

Exercise 3

7. Show the name of the managers who have at least one dependent.

Select name from (Department inner join Dependent on managerSSN=ssnE) inner join Employee on ssn=ssnE

Π_{name} (Department x $_{managerSSN=ssnE}$ Dependent x $_{ssn=ssnE}$ Employee)

Exercise 4

According to the following table named Grades:

a) Select (GradeA+GradeB) from Grades
b) Select (GradeA+GradeB+GradeT) from Grades
c) Select max(GradeA+GradeB) from Grades
d) Select max(GradeA)+max(GradeB) from Grades
e) Select count(GradeT) from Grades
f) Select avg(GradeT) from Grades
g) Select avg(GradeT+GradeB) from Grades
h) Select max(avg(GradeA)) from Grades

Grade A	Grade B	Grade T
8	5	NULL
10	NULL	NULL
8	6	NULL
7	5	NULL

Exercise 4

a) Select (GradeA+GradeB) from Grades
b) Select (GradeA+GradeB+GradeT) from Grades
c) Select max(GradeA+GradeB) from Grades
d) Select max(GradeA)+max(GradeB) from Grades

NULL + value = NULL
NULL is ignored in aggregated function max

Grade A	Grade B	Grade T
8	5	NULL
10	NULL	NULL
8	6	NULL
7	5	NULL

13
NULL
14
12

a)

NULL
NULL
NULL
NULL

b)

14

c)

16

d)

Exercise 4

e) Select count(GradeT) from Grades
f) Select avg(GradeT) from Grades
g) Select avg(GradeT+GradeB) from Grades
h) Select max(avg(GradeA)) from Grades

Grade A	Grade B	Grade T
8	5	NULL
10	NULL	NULL
8	6	NULL
7	5	NULL

e)	f)	g)	h)
0	NULL	NULL	error