7.18.a. Retrieve the names of all employees in department 5 who work more than 10 hours per week on the 'ProductX' project.

```
select   e.Name
from     e in EMPLOYEES, d in e.works_for
where    d.Number = 5
and      exists wo in e.works_on :
         wo.Hours > 10 and wo.project.Name = 'ProductX';
```


7.18.b. List the names of all employees who have a dependent with the same first name as themselves.

```
select   e.Name
from     e in EMPLOYEES
where    exists dep in e.has_dependents :
         dep.name = e.Name.Fname;
```

```
select   distinct e.Name
from     e in EMPLOYEES, dep in e.has_dependents
where    dep.name = e.Name.Fname;
```

Why is distinct needed only in version 2?

In version 1, each EMPLOYEES is selected only once. Thus the result is a set.
In version 2, each EMPLOYEES is paired with all DEPENDENTS. Thus a bag may result.

7.18.c. Find the names of all employees who are directly supervised by 'Franklin Wong'.

Starting with the *employees*.

```
select   e.Name
from     e in EMPLOYEES, super in e.supervisor
where    super.Name.Fname = 'Franklin'
and      super.Name.Lname = 'Wong;
```

```
select   e.Name
from     e in EMPLOYEES
where    e.supervisor.Name.Fname = 'Franklin'
and      e.supervisor.Name.Lname = 'Wong;
```


7.18.c. Find the names of all employees who are directly supervised by 'Franklin Wong'.

Starting with the *supervisors*.

```
select   e.Name
from     super in EMPLOYEEs, e in super.supervisee
where    super.Name.Fname = 'Franklin'
and      super.Name.Lname = 'Wong;
```

```
select   super.supervisee.Name
from     super in EMPLOYEES
where    super.Name.Fname = 'Franklin'
and      super.Name.Lname = 'Wong;
```


7.18.d. For each project, list the project name and the total hours per week (by all employees) spent on the project.

```
select     projName,
           TotHours : sum(select p.wo.Hours from p in partition)
from       wo in WORKS_ON_PROJECT
group by   projName : wo.project.Name;
```

Must have an instance for each employee on each project. Thus must start with WORKS_ON_PROJECT

7.18e. Retrieve the names of all employees who work on every project.

Examine all the WORKS_ON_PROJECT for the employee "e". Check project.

```
select   e.Name
from     e in EMPLOYEES
where    for all p in
           (select p
            from p in PROJECTS) :
            exists wo in
              (select wo
               from   e.works_on :
                      wo.project = p);
```

Ranges over projects for this e.

7.18e. Retrieve the names of all employees who work on every project.

Examine all the WORKS_ON_PROJECT for the project "p". Check employee.

```
select   e.Name
from     e in EMPLOYEES
where    for all p in
           (select p
            from p in PROJECTS) :
            exists wo in
              (select  wo
               from    p.employees_on :
                       wo.employee = e);
```

Ranges over projects for this p.

7.18e. Retrieve the names of all employees who work on every project.

Examine all the WORKS_ON_PROJECT Check employee and project.

```
select   e.Name
from     e in EMPLOYEES
where    for all p in
           (select p
            from p in PROJECTS) :
            exists wo in
              (select  wo
               from    wo in WORKS_ON_PROJECT :
                       wo.employee = e and wo.project = p);
```

Ranges over **all** projects.

7.18.f. Retrieve the names of all employees who do not work on any project.

```
select   e.Name
from     e in EMPLOYEES
where    not exists wo in e.works_on;
```


7.18.g. For each department, retrieve the department name and the average salary of all employees working in that department.

```
select      deptName,
            avgSal : AVG(select p.e.Salary from p in partition)
from        e in EMPLOYEES
group by    deptName : e.works_for.Name
```


7.18.h. Retrieve the average salary of all female employees.

```
avg( select   e.Salary
     from     e in EMPLOYEES
     where    e.Sex = F) ;
```


7.18.i. Find the names and addresses of all employees who work on at least one project located in Houston, but whose department has no location in Houston.

```
select   e.Name, e.Address
from     e in EMPLOYEES, d in e.works_for
where    exists wo in e.works_on :
             wo.project.located_at.Name = 'Houston'
and      'Houston' not in d.located_at.Name;
```

Note that d.located_at.Name is a **set**.

7.18.j. List the last names of all department managers who have no dependents.

```
select  e.Name
from    d in DEPARTMENTS, e in d.managed_by.manager
where   not exists dep in e.has_dependents ;
```