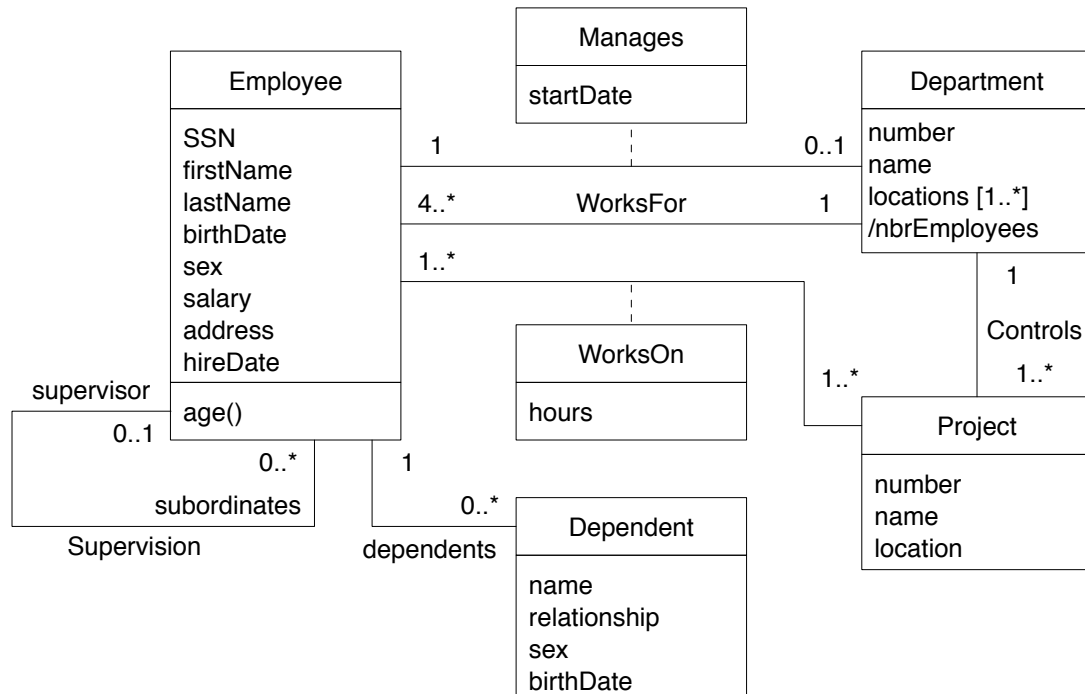


Object Constraint Language (OCL)

Integrity Constraints



Given the class diagram above, define in OCL the following constraints.

- (1) The age of employees must be greater than or equal to 18
- (2) The supervisor of an employee must be older than the employee
- (3) The salary of an employee cannot be greater than the salary of his/her supervisor
- (4) The hire date of employees must be greater than their birth date
- (5) The start date of an employee as manager of a department must be greater than his/her hire date
- (6) A supervisor must be hired before every employee s/he supervises
- (7) The manager of a department must be an employee of the department
- (8) The SSN of employees is an identifier (or a key)
- (9) The Name and Relationship of dependents is a partial identifier: they are unique among all dependents of an employee
- (10) The location of a project must be one of the locations of its department
- (11) The attribute nbrEmployees in Department keeps the number of employees that works for the department
- (12) An employee works at most in 4 projects
- (13) An employee may only work on projects controlled by the department in which s/he works
- (14) An employee works at least 30h/week and at most 50 h/week on all its projects
- (15) A project can have at most 2 employees working on the project less than 10 hours
- (16) Only department managers can work less than 5 hours on a project
- (17) Employees without subordinates must work at least 10 hours on every project they work

- (18) The manager of a department must work at least 5 hours on all projects controlled by the department
- (19) An employee cannot supervise him/herself
- (20) The supervision relationship must not be cyclic

Answers

- (1) The age of employees must be greater than or equal to 18

```
context Employee inv:
  self.age() >= 18
```

- (2) The supervisor of an employee must be older than the employee

```
context Employee inv:
  self.supervisor->notEmpty() implies
  self.age() < self.supervisor.age()
```

The condition `notEmpty` must be tested since the multiplicity of the role is not mandatory.

- (3) The salary of an employee cannot be greater than the salary of his/her supervisor

```
context Employee inv:
  self.supervisor->notEmpty() implies
  self.salary < self.supervisor.salary
```

- (4) The hire date of employees must be greater than their birth date

```
context Employee inv:
  self.hireDate > self.birthDate
```

- (5) The start date of an employee as manager of a department must be greater than his/her hire date

```
context Employee inv:
  self.manages->notEmpty() implies
  self.manages.startDate > self.hireDate
```

- (6) A supervisor must be hired before every employee s/he supervises

```
context Employee inv:
  self.subordinates->notEmpty() implies
  self.subordinates->forall( e | e.hireDate >= self.hireDate )
```

- (7) The manager of a department must be an employee of the department

```
context Department inv:
  self.worksFor->includes(self.manages.employee)
```

- (8) The SSN of employees is an identifier (or a key)

```
context Employee inv:
  Employee.allInstances->forall( e1, e2 |
  e1 <> e2 implies e1.SSN <> e2.SSN )
```

- (9) The name and relationship of dependents is a partial identifier: they are unique among all dependents of an employee

```
context Employee inv:
  self.dependents->notEmpty() implies
  self.dependents->forall( e1, e2 | e1 <> e2 implies
  ( e1.name <> e2.name or e1.relationship <> e2.relationship ) )
```

- (10) The location of a project must be one of the locations of its department
- ```
context Project inv:
 self.controls.locations->includes(self.location)
```
- (11) The attribute nbrEmployees in Department keeps the number of employees that works for the department
- ```
context Department inv:
    self.nbrEmployees = self.worksFor->size()
```
- (12) An employee works at most in 4 projects
- ```
context Employee inv:
 self.worksOn->size() <= 4
```
- (13) An employee may only work on projects controlled by the department in which s/he works
- ```
context Employee inv:
    self.worksFor.controls->includesAll(self.worksOn.project)
```
- (14) An employee works at least 30h/week and at most 50 h/week on all its projects
- ```
context Employee inv:
 let totHours : Integer = self.worksOn->collect(hours)->sum() in
 totHours >= 30 and totHours <=50
```
- (15) A project can have at most 2 employees working on the project less than 10 hours
- ```
context Project inv:
    self.worksOn->select( hours < 10 )->size() <= 2
```
- (16) Only department managers can work less than 5 hours on a project
- ```
context Employee inv:
 self.worksOn->select(hours < 5)->notEmpty() implies
 Department.allInstances()->collect(manages.employee)->includes(self)
```
- If it is supposed that the manager of a department must be an employee of the department (constraint (7) above), then this constraint can be specified as follows
- ```
context Employee inv:
    self.worksOn->select( hours < 5 )->notEmpty() implies
    self.worksFor.manages.employee=self
```
- (17) Employees without subordinates must work at least 10 hours on every project they work
- ```
context Employee inv:
 self.subordinates->isEmpty() implies
 self.worksOn->forall(hours >=10)
```
- (18) The manager of a department must work at least 5 hours on all projects controlled by the department
- ```
context Department inv:
    self.controls->forall( p:Project |
    self.manages.employee.worksOn->select(hours >= 5)->contains(p) )
```

(19) An employee cannot supervise him/herself

```
context Employee inv:  
  self.subordinates->excludes(self)
```

(20) The supervision relationship must not be cyclic

```
context Employee  
def: allSubordinates = self.subordinates->union(  
  self->subordinates->collect( e:Employee | e.allSubordinates ) )  
inv: self.allSubordinates->excludes(self)
```