CSC 8490

ODL-OQL handout
Figure 12.5  An example of a database schema. (b) A graphical object database schema for part of the UNIVERSITY database.
class PERSON
{
  extent PERSONS
  key Sen
  { attribute struct Pname { string Fname,
  string Mname,
  string Lname } Name;
  attribute string Ssn;
  attribute date Birth_date;
  attribute enum Gender(M, F)
  Sex;
  attribute struct Address { short No,
  string Street,
  short Apt_no,
  string City,
  short State,
  short Zip } Address;
  short Age( );
}

class FACULTY extends PERSON
{
  extent FACULTY
  { attribute string Rank;
  attribute float Salary;
  attribute string Office;
  attribute string Phone;
  relationship DEPARTMENT Works_in inverse DEPARTMENT::Hasfaculty;
  relationship set<GRAD_STUDENT> Advises inverse GRAD_STUDENT::Advisor;
  relationship set<GRAD_STUDENT> On_committee_of inverse GRAD_STUDENT::Committee;
  void give_raise(in float raise);
  void promote(in string new rank);
}

class GRADE
{
  extent GRADES
  { attribute enum GradeValues[A, B, C, D, F, I, P] Grade;
  relationship SECTION Section inverse SECTION::Students;
  relationship STUDENT Student inverse STUDENT::Completed_sections;
}

class STUDENT extends PERSON
{
  extent STUDENTS
  { attribute string Class;
  attribute Department Minors_in;
  relationship Department Majors_in inverse DEPARTMENT::Has_majors;
  relationship set<GRADE> Completed_sections inverse GRADE::Student;
  relationship set<CURR SECTION> Registered_in inverse CURR SECTION::Registered_students;
  void change_major(in string dname) raises(dname_not_valid);
  float gpa();
  void register(in short secno) raises(section_not_valid);
  void assign_grade(in short secno; in GradeValue grade)
  raises(section_not_valid, grade_not_valid);
}
Q0: Select d.dname
    From d in departments
    Where d.college = 'Engineering';

Q1: departments;

Q1a: csdepartment;

Q2: csdepartment.chair;

Q2a: csdepartment.chair.rank;

Q2b: csdepartment.has_faculty;

Q3: csdepartment.has_faculty.rank;

Q3a: select f.rank
    from f in csdepartment.has_faculty;

Q3b: select distinct f.rank
    from f in csdepartment.has_faculty;
Q4: csdepartment.chair.advises;

Q4a: select struct (name: struct(last_name: 
    s.name.lname, 
    first_name: 
    s.name.fname), 
    degrees: (select struct (deg: 
        d.degree, 
        yr: d.year, 
        College: 
        d.college) 
        from d in s.degrees)) 
    from s in csdepartment.chair.advises;

Q5a: select struct (last_name: s.name.lname, 
                   first_name: s.name.fname, 
                   gpa: s.gpa) 
    from s in csdepartment.has_majors 
    where s.class = 'senior' 
    order by gpa desc, last_name asc, 
           first_name asc;
Q5b:    select struct (last_name: s.name.lname,  
            first_name: s.name.fname,  
            gpa: s.gpa)  
    from s in students  
    where s.majors_in.dname =  
            'Computer Science'  
    and s.class = 'senior'  
    order by gpa desc, last_name asc,  
            first_name asc;  

V1:    define has_minors(dptname) as  
    select s  
    from s in students  
    where s.minor_in.dname = dptname;  

Q6:    element (select d  
            from d in departments  
            where d.dname = 'Computer Science');  

Q7:    count (s in has_minors('Computer Science'));  

Q8:    avg (select s.gpa  
            from s in students  
            where s.major_in.dname = 'Computer Science'  
            and s.class = 'senior');
Q9:  
\[
\text{select d.dname from d in departments where count (d.has_majors) > 100;}
\]

Q10:  
\[
\text{select s.name.lname, s.name.fname from s in students where 'Database Systems I' in (select c.cname from c in s.completed_sections.section.of_course);}
\]

Q11:  
\[
\text{Jeremy in has_minors('Computer Science');}
\]

Q12:  
\[
\text{for all g in (select s from s in grad_students where s.majors_in.dname = 'Computer Science') : g.advisor in csdepartment.has_faculty;}
\]

Q13:  
\[
\text{exists g in (select s from s in grad_students where s.major_in.dname = 'Computer Science') : g.gpa = 4;}
\]
Q10 (corrected):

```sql
select s.name.lname, s.name.fname
from s in students,
    q in s.completed_sections
where q.section.of_course.cname = 'Database Systems I';
```
Q14: `first (select struct(faculty: f.name.lname, salary: f.salary)
       from f in faculty
       order by f.salary desc);
``

Q15: `(select struct(last_name: s.name.lname,
                   first_name: s.name.fname,
                   gpa: s.gpa)
       from s in csdepartment.has_majors
       order by gpa desc) [0:2];`

Q16: `select struct(deptname, number_of_majors:
                   count (partition))
       from s in students
       group by deptname: s.major_in.dname;
``

Q17: `select deptname, avg_gpa: avg (select p.s.gpa
       from p
       in partition)
       from s in students
       group by deptname: s.majors_in.dname
       having count (partition) > 100;`