1) Consider the following CREATE statements expressed in the object-relational SQL-99 standard:

```sql
CREATE TYPE professor_t AS
    P#    INTEGER,
    Lname  VARCHAR (20),
    Office#  INTEGER,
    Phone#  INTEGER
    NOT FINAL;

CREATE TYPE course_t AS
    C#    INTEGER,
    Name  VARCHAR (20),
    Classroom CHAR (8)
    NOT FINAL;

CREATE TABLE Course_assignment
    (C#    course_t,
    P#    professor_t,
    Semester  INTEGER);
```

(a) (pt.10) Write the SQL INSERT statement for inserting in table Course_assignment a tuple that assigns course 201 to Professor Smith for the fall semester.

- Course 201 has the following attribute values
  - C# is equal to 201
  - Name is equal to ‘Computer Security’
  - Classroom is equal to 123
- Professor Smith has the following attributes
  - P# is equal to 1457
  - Lname is equal to ‘Smith’
  - Office# is equal to 50
  - Phone is equal to 4962378

(b) (pt. 10) Suppose that the above insert has been executed and suppose that we need to change the phone number of Professor Smith to 4875344. Write the SQL UPDATE statement for making this change on table Course_assignment.

(c) (pt. 10) Suppose that we want to change the assignment of Professor Smith and assign him the course (205, ‘Computer Graphics’, 465). Write the SQL SQL UPDATE statement for making this change to table Course_assignment.

(d) (pt.10) Write an SQL query that selects from table Course_assignment the names of the courses that are scheduled to be held in classroom LWSN123 in the fall semester.
2) Consider the following CREATE statements expressed in object-relational SQL-99 standard:

```sql
CREATE TYPE professor_t AS
  P# INTEGER,
  Lname VARCHAR (20),
  Office# INTEGER,
  Phone# INTEGER
NOT FINAL;

CREATE TABLE Professors OF professor_t;
```

(a) (pt 10) Suppose that the following constraints must be enforced:
   a. The primary key is attribute P#.
   b. Attribute Lname cannot be null.

Extend the above statements to include the specification of these constraints.

3) (pt.10) Consider the following three SQL statements:

```sql
CREATE TYPE professor_t AS
  P# INTEGER,
  Lname VARCHAR (20),
  Office# INTEGER,
  Phone# INTEGER
NOT FINAL;

CREATE TYPE assistant_professor_t AS
  P# INTEGER,
  Lname VARCHAR (20),
  Office# INTEGER,
  Phone# INTEGER,
  Contract_years INTEGER
NOT FINAL;

CREATE TABLE Professors OF professor_t;
CREATE TABLE Assistant_Professors OF assistant_t UNDER Professors;
```

Are these statements correct? Answer YES or NO.
If your answer NO, indicate how the statements should be modified in order to be correct.

4) Consider the following three SQL statements:

```sql
CREATE TYPE professor_t AS
  P# INTEGER,
  Lname VARCHAR (20),
  Off# INTEGER,
  Phone# INTEGER
NOT FINAL;
```
CREATE TABLE Professors OF professor_t (REF IS prof_id);

CREATE TABLE Departments
(D#: INTEGER,
 Dname: VARCHAR (10),
 Head: REF (professor_t) SCOPE Professors);

(a) (pt. 10) Specify the SQL statement for creating a new department with the following attributes:
   a. The D# attribute is equal to 200;
   b. The Dname attribute is equal to ‘Computer Science’
   c. The Head is Professor Green.

(b) (pt. 10) Specify the SQL statement for retrieving the name of the department whose head is Professor Red.

(c) (pt. 10) Specify the SQL statement for retrieving the name of the professor who is the head of the department with name equal to ‘Foreign Languages’.

(d) (pt.10) Suppose that we cannot delete a tuple from the table Professors, if the professor corresponding to this tuple is the head of a department. Extend the above SQL statements to enforce this referential integrity constraint.