Verification Page

Fill in and separate this page from the rest and place at a corner of your desk.

Name	
Student Number	
Signature	

Under the honor code, I will not disclose the material contained in this exam to any person who has not taken it.

INFS614 - Midterm Exam

Fall 2007

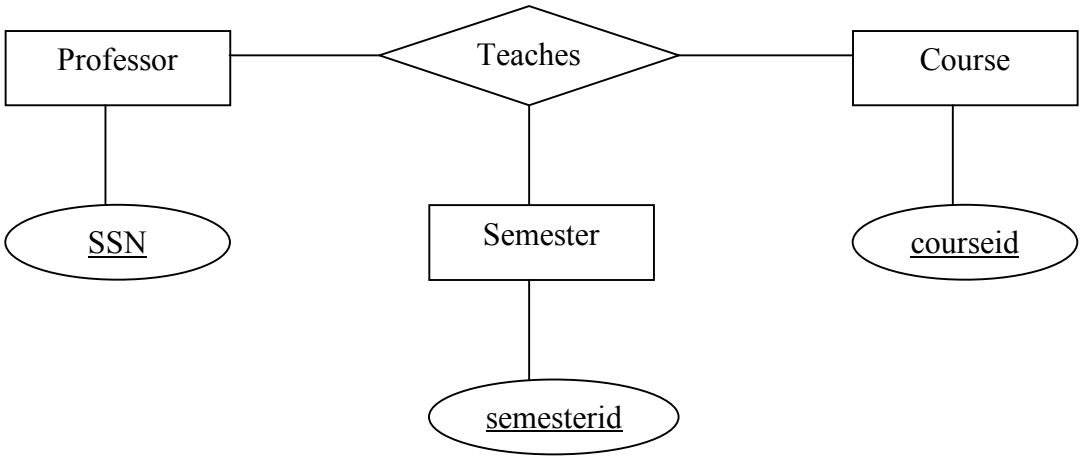
Name: _____

G-Number: _____

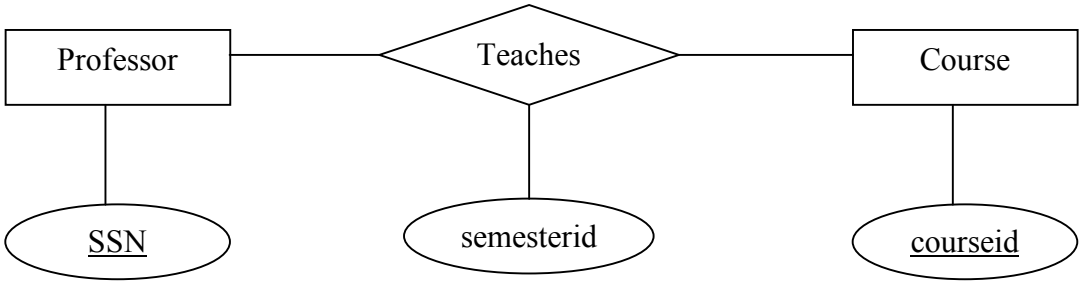
Problem 1. (25 points)

A university database contains information about professors (identified by social security number, or SSN) and courses (identified by courseid). Professors teach courses; each of the following situations concerns the Teaches relationship set. For each situation, draw an ER diagram that describes it (assuming no further constraints hold).

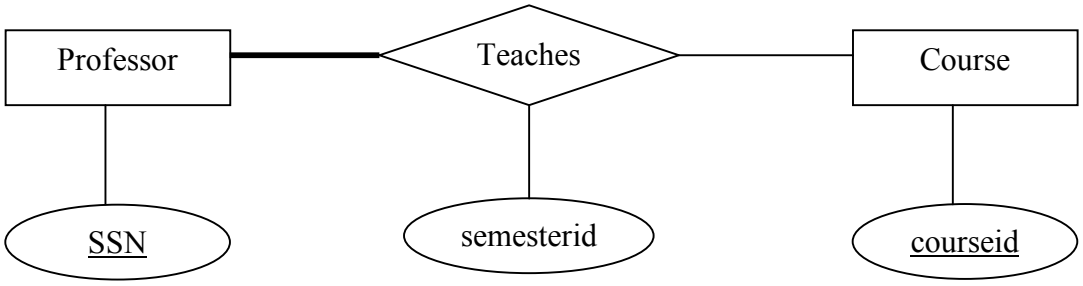
1. Professor can teach the same course in several semesters, and each offering must be recorded.



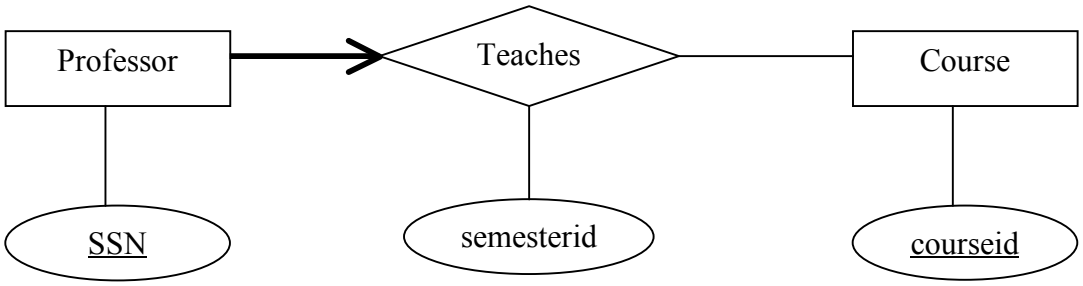
2. Professors can teach the same course in several semesters, and only the most recent such offering needs to be recorded. (Assume this condition applies in all subsequent questions).



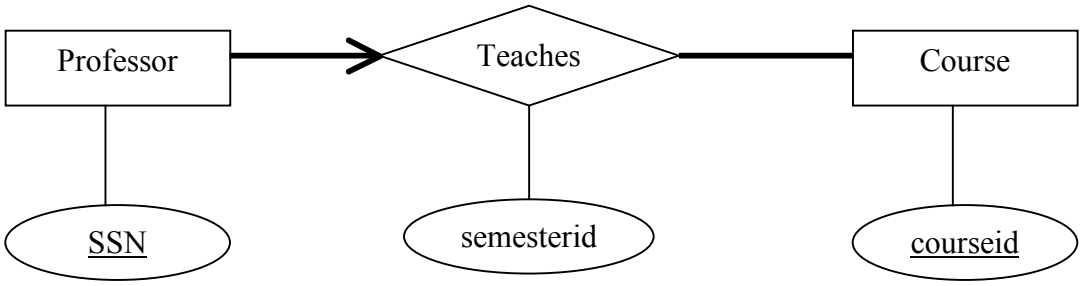
3. Every professor must teach some course.



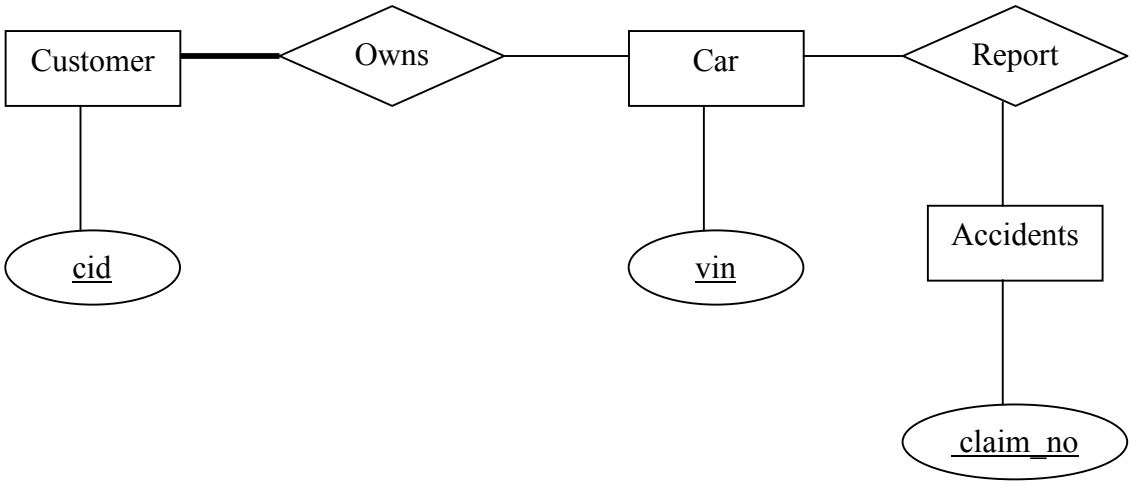
4. Every professor teaches exactly one course (no more, no less).



5. Every professor teaches exactly one course (no more, no less), and every course must be taught by some professor.



Problem 2. (15 points)
Construct an ER diagram for a car insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents.



Problem 3. (15 points) Consider the following relation schemas, where primary keys are underlined.

Employee(person_name, street, city)
Works(person_name, company_name, salary)
Company(company_name, city)
Manages(person_name, manager_name)

Give an expression in relational algebra, tuple relational calculus, and SQL to express each of the following queries:

(15 points) Find all managers (names) who manage employees that work for company ABC or company XYZ. The managers might not necessarily work in the same company as the employees they manage.

Relational Algebra:

$\pi_{\text{manager\_name}} (\pi_{\text{person\_name}} (\sigma_{\text{company\_name='ABC'} \vee \text{company\_name='XYZ'}} \text{Works}) \bowtie \text{Manages}$

Relational Calculus:

{P | ∃W ∈ Works ∃M ∈ Manages((W.company_name = 'ABC' ∨ W.company_name = 'XYZ') ∧ M.person_name = W.person_name ∧ P.manager_name = M.manager_name)}

SQL:

```
SELECT M.manager_name
FROM   Manages M, Works W
WHERE (W.company_name = 'ABC' OR W.company_name = 'XYZ') AND
          M.person_name = W.person_name
```

Employee(person_name, street, city)
Works(person_name, company_name, salary)
Company(company_name, city)
Manages(person_name, manager_name)

(15 points) Find the names of the employee who earns the highest salary in company ABC.

Relational Algebra:

ρ(ABC_Emp, $\pi_{\text{person\_name, salary}}$ $\sigma_{\text{company\_name='ABC'}}$ Works)

ρ(ABC_Emp_Pairs(1→name1, 2→sal1, 3→name2, 4→sal2), ABC_Emp × ABC_Emp)

$\pi_{\text{person\_name}}$ ABC_Emp - π_{name1} ($\sigma_{\text{sal1<sal2}}$ ABC_Emp_Pairs)

Relational Calculus:

{P | ∃W ∈ Works (∀W2 ∈ Works(W2.salary ≤ W.salary ∧ W.company_name = 'ABC' ∧ W2.company_name = 'ABC') ∧ P.person_name = W.person_name)}

OR

{P | ∃W ∈ Works (P.person_name = W.person_name ∧ W.company_name = 'ABC' ∧ ∀W2 ∈ Works(W2.company_name = 'ABC' => W.salary ≥ W2.salary))}

SQL:

```
SELECT W.person_name
FROM   Works W
WHERE W.company_name = 'ABC' AND
        W.salary = (SELECT MAX(W2.salary)
                    FROM   Works W2
                    WHERE  W2.company_name = 'ABC')
```

Employee(person_name, street, city)
Works(person_name, company_name, salary)
Company(company_name, city)
Manages(person_name, manager_name)

(Bonus: 6 points) Find the companies that have a branch in every city in the database.

Note: Assuming that a company can have multiple branches (i.e. ignore the key constraint in Company)

Relational Algebra:

Company ÷ π_{city} Company

Relational Calculus:

{P | ∀C ∈ Company(∃C2 ∈ Company(C2.city = C.city) ∧ P.company_name = C2.company_name)}

SQL:

```
SELECT C.company_name
FROM   Company C
WHERE NOT EXIST (SELECT city
                 FROM   Company
                 EXCEPT
                 SELECT C2.city
                 FROM   Company C2
                 WHERE C.company = C2.company_name)
```

Problem 4. Consider the schemas in Problem 3:

Employee(person_name, street, city)
Works(person_name, company_name, salary)
Company(company_name, city)
Manages(person_name, manager_name)

(2 points)
A) T/**F** {person_name, salary} in the Works relation is a candidate key.

False

(2 points)
B) **T**/F {person_name, salary} in the Works relation is a super key.

True

(5 points)
C) Name one foreign key constraint that should be specified on the schemas.

Any one of the followings:

person_name in Works (references Employee)
person_name in Manages (references Employee)
manager_name in Manages (references Employee)
company_name in Works (references Company)

(5 points)
D) Write the SQL statement for the foreign key constraint from the previous question.

person_name in Works (others are similar):

```
CREATE TABLE Works (person_name      CHAR(20),
                    company_name     CHAR(20),
                    salary           REAL,
                    FORIGN KEY(person_name)
                    REFERENCES Employee)
```