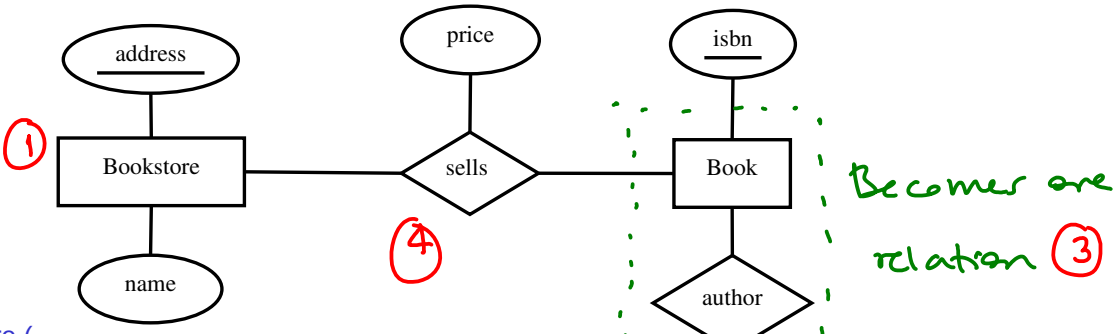
2. **Entity Relationship**

(a) [4 points] Provide `CREATE TABLE` statements for the SQL relations that represent the following ER diagram. Choose appropriate types for the attributes. Make sure you include in each relation all its attributes and constraints.




```
1  CREATE TABLE Bookstore (
       address varchar,
       name    varchar,
       PRIMARY KEY (address)
   );

2  CREATE TABLE Person(
       firstname  varchar,
       lastname  varchar,
       PRIMARY KEY (firstname, lastname)
   );

3  CREATE TABLE BookAuthor(
     isbn    varchar,
     authorfirstname varchar NOT NULL,
     authorlastname varchar NOT NULL,
     PRIMARY KEY (isbn),
     FOREIGN KEY (authorfirstname, authorlastname) REFERENCES Person
   );

4  CREATE TABLE Sells(
     isbn  varchar,
     address varchar,
     price real,
     PRIMARY KEY (isbn, address),
     FOREIGN KEY (isbn) REFERENCES BookAuthor
   );
```

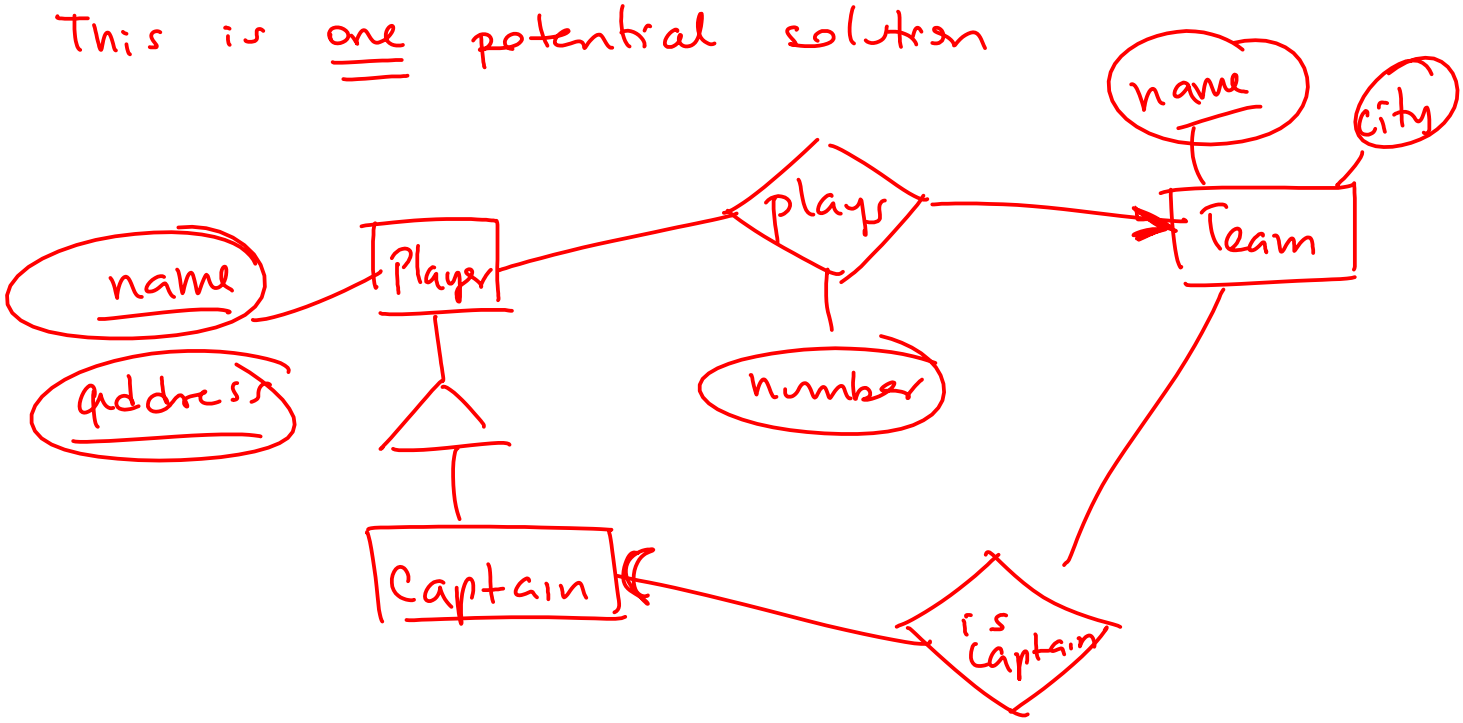
Any name is ok. the type of attributes can be any.

any names, but must match.

order must match

(b) [4 points] Draw the ER diagram for the following database requirements. Choose appropriate names for the entities and attributes. **Hint:** do not over complicate it and do not worry about things that are not explicitly indicated below (e.g. there is no restriction on how many players can have the same number in a given team).

1. A player has a name, age and address. A player can be uniquely identified by providing both: his/her name and address.
2. Each team has a unique name. We want to keep track of the city where the team plays.
3. A player plays for at most one team.
4. A team assigns each of its players a number.
5. A team must have exactly one captain (a player).



 Points earned: _____ out of a possible 4 points

For this exam, consider the following schema of a simple university database. It includes information about instructors, students, and the courses offered. Feel free to remove this page from the exam.

- The **Students** table contains the id of the student (**sid**), his/her name (**sname**), age (in years), and gpa.

```
Students(sid: integer, sname: string,
         age: integer, gpa: real)
```

  - Key: **sid**

- The **Instructors** table contains information about instructors of the courses: their id (**iid**), name (**iname**) and department they belong to (**dept**).

```
Instructors(iid: string, iname: string, dept: string)
```

  - Key: **sid**

- The **Courses** table contains information about courses: their id (**cid**), their name (**cname**), the department that offers it (**dept**), the id of its instructor (**iid**), and the maximum number of students who can take it (**maxenrol**). Every **iid** in this table is also found in the table **Instructors**.

```
Courses(cid: string, cname: string,
        dept: string, iid: string,
        maxenrol:integer
)
```

  - Key: **cid**

- The table **Enrolled** contains what students are registered to which courses, and the grade they receive (NULL if they have not received one yet). A student can only register once to any given course, but he/she can register to as many courses as necessary. Neither **sid** nor **cid** can be NULL. Every **sid** in this table is also found in the table **Students**, and every **cid** in this table is also found in the table **Courses**.

```
Enrolled(sid: integer, cid: string,
         grade: integer)
```

  - Key: (**sid,cid**)

Relational Algebra and SQL

For each of the following questions, provide a relational algebra expression to answer them, and its equivalent SQL query:

1. Part 1

   (a) [4] ~~List the **iid**~~ of instructors that are not teaching any course. Result should have only one column.

   (b) [4] ~~List the **iid**~~ of the instructors of classes in which student with **sId** = 12345 is enrolled. Result should have only one column.

   (c) [4] Suppose relations R(a,b,c) and S(a,b,c) have the following tuples.

   R

A	B	C
1	2	3
4	2	3
4	5	6
2	5	3
1	2	6

   S

a	b	c
2	5	3
2	5	4
4	5	6
1	2	3

   - Compute the result of $(R - S) \cup (S - R)$.

   - Write an SQL query for this Relational Algebra expression.

2. Part 2

How many

(a) [4] ~~List the~~ **sId** of students ~~who~~ are currently taking the course "csc370" and "seng360". Your result should have only one column.

$$\gamma_{count(*)}\left[\left(\Pi_{sid}\,\sigma_{cid='370'}\,E\right)\cap\left(\Pi_{sid}\,\sigma_{cid='seng360'}\,E\right)\right]$$

count #

(b) [4] ~~For every~~ student who has exactly the same grade in two different courses, ~~list their~~ **sid**, ~~the course id~~ **cid**, ~~and grade~~ **grade** ~~of both courses~~. Make sure each pair of courses is listed only once. Your result should have 5 columns.

$E2 = E$

$$\gamma_{count(distinct\ sid)}\left(E \bowtie_{E.sid=E2.sid\ \wedge\ E.cid\neq E2.cid\ \wedge\ E.grade=E2.grade} E2\right)$$

(c) [4] For every course in the table Enrolled, lists its **cId**, and the number the spaces left (i.e. **maxEnrol** minus the number of students who are currently enrolled in it). Result should have two columns.

$$CE = \gamma_{cid,\ count(*)\Rightarrow c}\ E$$

$$\Pi_{cid,\ maxEnrol-c}\left(CE \bowtie C\right)$$

3. Part 3

(a) [4] For every student who has received a **grade** equal to zero, list the **sid** of the student. Make sure your result contains only one tuple per **sid**. You cannot use the SQL keyword DISTINCT. Make sure your SQL query matches the relational algebra. Your result should have only one column: *sid*

$\gamma_{sid, count(*) \to c} \sigma_{grade=0} E$

and the # of courses grade equal to zero.

$JW = \Pi_{sid} \sigma_{sname = 'John Wayne'} S$

$JWC = \gamma_{count(*) \to n} JW$

(b) [4] List the course id (**cid**) of all the courses in which student with **sname** *John Wayne* is not enrolled. Your result should have only one column **cid**.

list (cid, # of students with sname ...)

$CJW = \Pi_{cid} (E \bowtie JW)$

$CNJW = \gamma_{count(*) \to nc} (\Pi_{cid} E - CJW)$

$\Rightarrow$ $CNJW \times JWC$

---

c) List instructors names who are teaching 3 courses.

$IC = \gamma_{iid, count(*) \to c} C$

$IC3 = \sigma_{c=3} IC$

$\Pi_{iname} (IC3 \bowtie I)$

End of examination
Total pages: 5
Total marks: 32