Q1. SQL

1.1. Explain the following query in English.

```
select givename, famname, instname
from academic natural join department
where acnum in (select acnum from author
        where acnum not in (select acnum
                from interest
                group by acnum))
        and deptNum in (select deptNum
                from academic
                where deptname = 'Computer Science');
```

The query will show the given name, family name and institution name from the academic and department table. These tables are joined using Natural Join where nested sub query is used to get academic number from author table but not from interest table grouped by academic number. Also, department number is selected from academic table with department name as computer science.

1.2. The following SQL query is meant to output a list of papers (panum) with the total number of authors for each paper. It has syntax errors and logic errors. Explain the syntax and logic errors and give the correct query

```
select  PaNum, count(A1.AcNum)
from Author A1, Author A2
where PaNum = A2.PaNum
group by PaNum;
```

The above query has a logical as well as syntax error.
Instead of Author A2, paper p should be used. The condition should be between panum of paper table and panum of author table, so that we get the total no of authors for papers.

```
Select p.panum, Count(a.acnum)
From author a, paper p
Where p.panum = a.panum
Group By p.panum;
```

1.3. Find departments that have a description (descrip) available in the database. Return all details of these departments.

```
Select *
From department
Where descrip is NOT null;
```

1.4. List the paper number and title of papers by the academic whose acnum is 100.

```
Select p.panum, p.title
From paper p
Where p.panum in (Select a.panum
        From author a
        Where a.acnum = 100);
```

1.5. For each academic, give the acnum, givename, famname and the total number of papers s/he has written. Note that if an academic has not written any paper, his/her total should be zero. You can use JOIN operators such as NATURAL, JOIN ...ON.

```
Select ac.acnum, ac.givename, a.cfamname,
  Count (au.panum)
From academic ac
  Join author au on ac.acnum = au.acnum
Group By  ac.acnum, ac.givename, ac.famname;
```

1.6. The research field ID is a research field classification code representing classes for three "Levels". These three Levels are separated by a "full stop" in a single string. For example the research field ID "B.1.6" represents that the research field belongs to Class "B" for Level one, Class "1" for Level two and Class "6" for Level three. For research fields in Class "1" for Level two, list the field IDs and the number of academics for each field ID

```
Select f.id,
  Count(i.acnum)
From field f, interest i
Where f.id like '_.1%'
  and f.fieldnum =  i.fieldnum
Group By f.id;
```

1.7. Find departments where at least one academic does not have research interest, and list the deptnum, deptname, instname of these departments. Must use a subquery

```
Select deptnum, deptname, instname
From department
Where deptnum in (select a.deptnum
   From academic a
     Left Join interest i on a.acnum =  i.acnum
   Where i.acnum is null);
```

1.8. Output in alphabetical order the acnum, famname, givename, and department number (deptnum), and description (descrip) of authors whose family name starts with "C".

```
Select  ac.acnum, ac.famname, ac.givename, d.deptnum, d.descrip
From academic ac
   Join department d On d.deptnum = ac.deptnum
Where ac.famname like 'c%'
Order By ac.famname, ac.givename;
```

1.9. List the fieldnum, title, and total number of interested academics (under the heading "NO. ACADEMICS INTERESTED") in each research field, in increasing order (i.e. ascending order) of fieldnum.

```
Select fieldnum, title,
  Count(acnum) as "NO. ACADEMICS INTERESTED"
From interest
  Natural Join field
Group By fieldnum, title
Order By fieldnum ASC;
```

1.10. List in alphabetical order the institution and name of departments where at least 10 academics have written papers.

```
Select d.instname, d.deptname
From department d
  Join academic ac On d.deptnum = ac.deptnum
Group By d.instname, d.deptname
Having count(ac.acnum) >= 10
Order By d.instname, d.deptname;
```

1.11. List the deptnum of departments whose postcodes are in the range 3000..3999 and that do not have any academics with the title of Professor (stored as "Prof" or "Prof." in the database) , including departments that do not have any academics.

```
Select d.deptnum
From department d
Where d.deptnum NOT IN (Select ac.deptnum
        From academic ac
        Where upper(ac.title) = 'PROF' or upper(ac.title) = 'PROF.')
            and d.postcode between 3000 and 3999;
```

1.12. Find the departments that have produced at least ten papers (that is, those departments where the sum of papers written by their academics is at least ten). Output their deptnum and deptname in ascending order.

```
Select d.deptnum, d.deptname
From department d
Where d.deptnum in (Select ac.deptnum
        From academic ac
          Join author au On ac.acnum = au.acnum
        Having count (au.panum) >= 10
        Group By ac.deptnum)
 Order by d.deptnum, d.deptname ASC ;
```

1.13. List the deptnum and deptname of departments whose academics have never written any papers.

```
Select deptnum , deptname
From department
  Natural Join academic
Where acnum NOT IN (Select Distinct a.acnum
          From author a);
```

1.14. List papers (panum) by academics with research interests in fields related to "data". You must use EXISTS. Note that "fields related to data" includes any occurrence of the four letters "data" within a field name, in any case.

```
Select Distinct a.panum
From author a
Where exists ( Select i.acnum
          From interest i, field f
          Where i.fieldnum = f.fieldnum
              and i.acnum = a.acnum
              and (upper(title) like '%DATA' or upper(title) like '%DATA%' or
              upper(title) like 'DATA%'));
```

1.15. The popularity of a field is measured by the number of interested academics. List details (fieldnum, ID and title) of the most popular field together with the total number of interested academics.

```
Select  f.fieldnum, f.id, f.title,
  Count(i.acnum)
From interest i
Join field f on i.fieldnum = f.fieldnum
Group By  f.fieldnum, f.id, f.title
Having Count(acnum) = (Select Max(Count(acnum))
          From interest
          Group By fieldnum);
```

Q2. The Relational Model

2.1. Give all likely FDs. Do not include trivial or redundant FDs.

deptID -> deptName
deptID -> manager
empID -> empName
empID -> email
empID -> deptID
projID -> startYear
projID -> deptID
empID, projID -> role
projID, evalDate -> manager
projID, evalDate -> grade

2.2. Give {empID, projID}+ and {deptID}+ based on the FDs in Question 2.1.

- {empID, empName, email, deptID, deptName, manager, projID, startYear, role} are the set of attributes for the closure **{empID, projID}+.**
- {deptID, deptName, manager} are the set of attributes for the closure **{deptID}+.**

2.3. Specify the primary key (by underlining) and any foreign keys (with *) for each relation.

Department	(deptID, deptName, manager)
Employee	(empID, empName, deptID*, email)
Project	(projID, startYear, deptID*)
EmpProj	(empID*, projID*, role)
Evaluation	(projID*, manager, evalDate, grade)

2.4. Discuss the normal form for the Evaluation relation using the FDs in Question 2.1.

Consider projID, evalDate -> manager and projID, evalDate -> grade, the l.h.s "projID, evalDate" are the keys for the relation Evaluation.
Since, the FDs of Evaluation relation have keys on l.h.s, therefore, Evaluation relation is in BCNF.

Q3. Normalisation

3.1. The given FDs may have redundancies. Give the minimal basis for the given FDs.

docID -> docName
patID -> patName
patID -> patDOB
patID, appDate -> appTime
patID, appDate -> roomNo
patID, appDate -> docID

3.2. Discuss all candidate keys for the APP relation. Explain your answer using the functional dependencies in Question 3.1.

The candidate key for the APP relation is {patID, appDate}.
The set of attributes for the closure {patID, appDate}+ are {patID, patName, patDOB, appDate, appTime, roomNo, docID, docName}

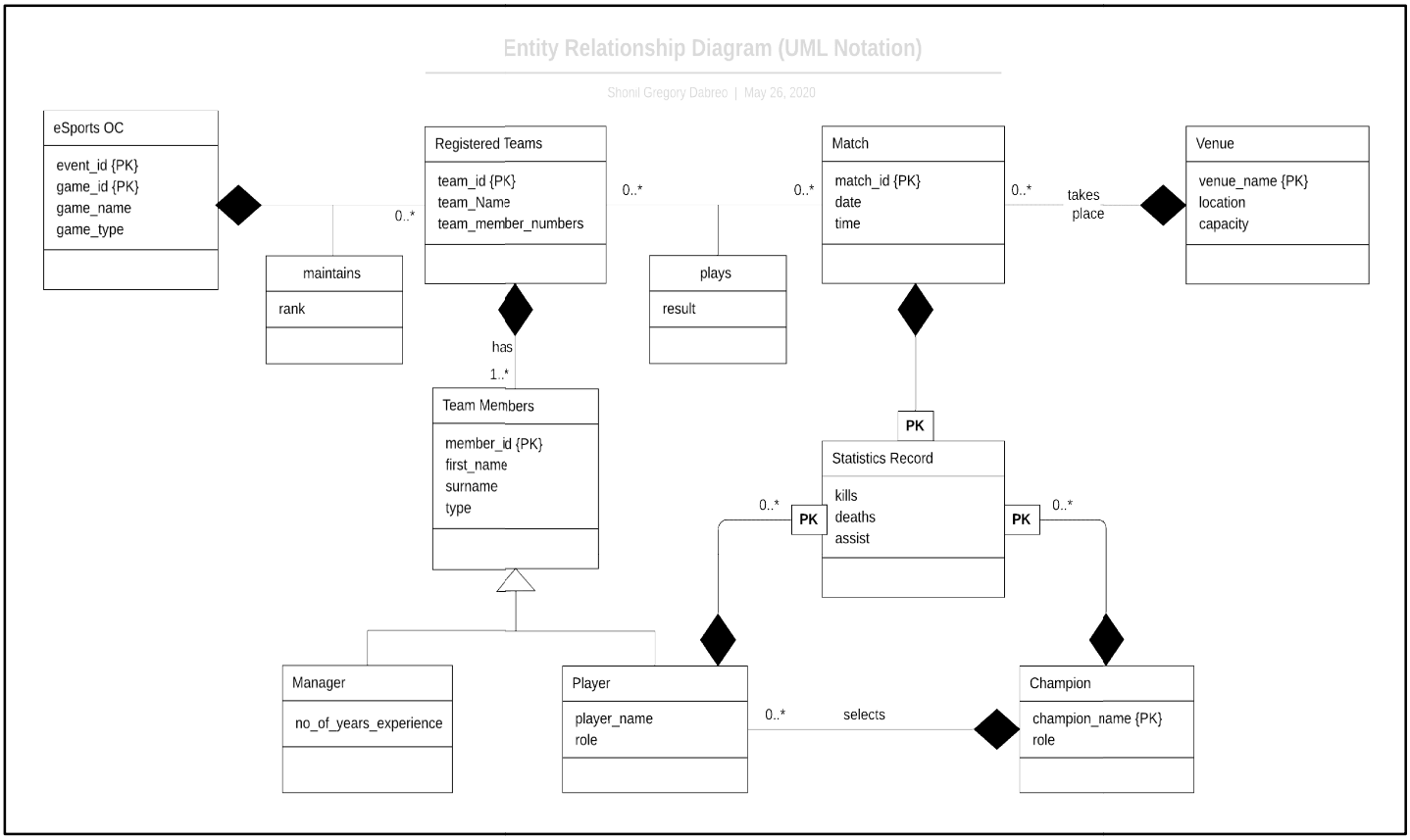
3.3. The APP relation is not in BCNF or 3NF. Decompose the relation into BCNF/3NF relations. Your decomposition must keep all functional dependencies and must be lossless. For each resultant relation, discuss if it is in BCNF or 3NF and indicate the primary key (underline) and any foreign keys (*).

- **Relations from the functional dependencies**

  DoctorInfo (docID, docName)
  PatientInfo (patID, patName, patDOB)
  Appointment (patID*, appDate, appTime, roomNo, docID*)

  The resulting relations are in 3NF, and indeed also in BCNF.

Q4. ER Model


Entity Relationship Diagram (UML Notation)
Shonil Gregory Dabreo | May 26, 2020
eSports OC
event_id {PK}
game_id {PK}
game_name
game_type
0..*
maintains
rank
Registered Teams
team_id {PK}
team_Name
team_member_numbers
0..*
plays
result
0..*
Match
match_id {PK}
date
time
0..*
takes place
Venue
venue_name {PK}
location
capacity
has
1..*
Team Members
member_id {PK}
first_name
surname
type
PK
Statistics Record
kills
deaths
assist
PK
PK
0..*
0..*
Manager
no_of_years_experience
Player
player_name
role
0..*
selects
Champion
champion_name {PK}
role

Q5. ER to Relational Schema Mapping

Class	(groupNo, day, time, roomNo, type, eno*, cno*)
Staff	(eno, givename, surname)
Tutor	(t_eno, t_givename, t_surname, contract)
Student	(sno, givename, surname, DOB, address)
Take	(groupNo*, sno*)
Course	(cno, title)
Enrol	(sno*, cno*, grade)

Note: t_eno, t_givename, t_surname are renamed to avoid confusion.