==

Question A. (50 marks)
1) A database is being constructed to keep track of the teams and games of a football league:

*the league has many teams, each team has a name, a city, a coach, a captain, and a set of players.

*each player belongs to only one team, each player has an id, a name, a position (such as right back or center forward), a skill level, and a set of injury records.

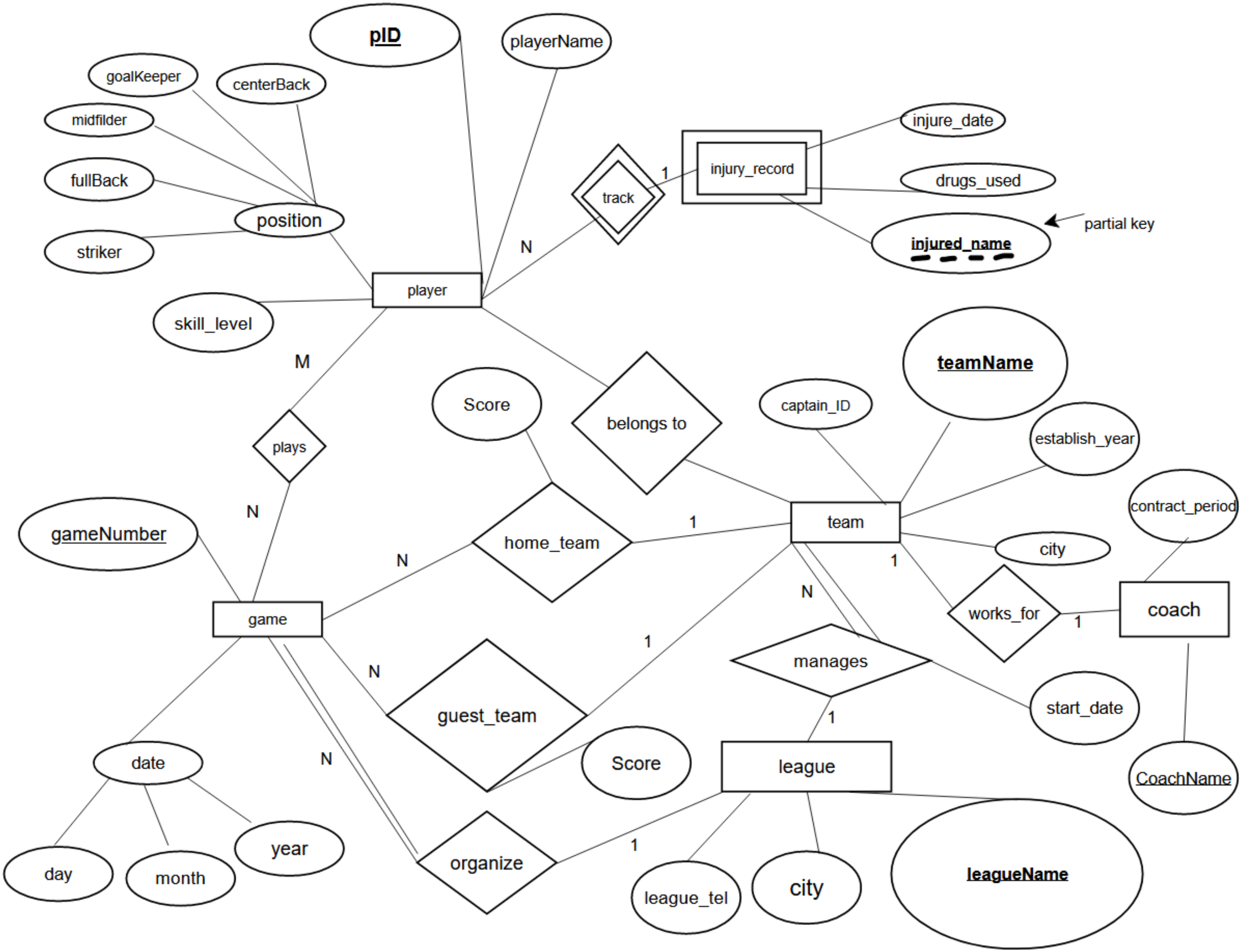
* a team captain is also a player

* each injury record keeps track of the payer who has been treated, the date of the treatment, several drugs used in the treatment.

* a game is played between two teams (referred to as host_team and guest_team) and has a date (such as June 11th, 2015) and a score (such as 4 to 2).

Design an ER diagram for this application, stating any assumptions you make. [30 marks]

2) For your ER diagram given above, convert it into relational schema using the mapping guidelines discussed in the lecture. For each relation (table) obtained, specify the name and its attributes, as well as its primary key. [20 marks]


pID
playerName
goalKeeper
centerBack
midfilder
fullBack
position
striker
skill_level
player
N
track
1
injury_record
injure_date
drugs_used
injured_name
partial key
M
plays
N
Score
belongs to
captain_ID
teamName
establish_year
gameNumber
N
home_team
1
team
contract_period
city
1
works_for
1
coach
N
game
N
guest_team
1
manages
start_date
1
Score
league
CoachName
date
N
organize
1
day
month
year
league_tel
city
leagueName

Assumptions:
Players has their pID for identification, which is the primary key.And they also have playerName, position, and skill_level as other attributes.

Injury_record is an week entity, it needs the primary key of player to help identify the injured player and it will be the foreign key. So injured_name is partial key. And drugs_used and injure_date are the other attributes. One player only has one record, but record has data belongs to many players.

The game has primary key of gameNumber, date. And M players can plays N games.

Home_team is a relationship connects game and team. The primary key should be the gameNumber and teamName. And there is also a attribute of score.

Guest_team is close to home_team, also include gameNumber and teamName to be primary keys, and an attribute of score.

A league has an primary key of leagueName, and other attributes like city and league_tel.

The league manages many teams, the primary key are teamName and leagueName.
And there is an attribute of start_date.

An the league organize many games. The primary key are leagueName and gameNumber.
There are many teams, each player belongs to one team. TeamName is the primary key. And there are establish_year, coach, city, and captain_ID

Each coach works for a team. And coach is also player, so they can be identified with their coachName.

game(gameNumber, date)

player(PID, position, skill, level)

injury_record(injured_name, injure_date, drugs_used)

team (teamName, coach, city, captain_ID)

organize (leagueName, teamName, GameNumber, location, city, league_tel)

home_team(gameNumber, teamNumber, score)

guest_team(gameNUmber, teamNumber, score)

track(PID, injured_name, drugs_used, injure_date) , where injured_name is partial key

manages(teamName, leagueName, start_date)

works_for(coachName, teamName, contract_period)

Question B. [50 marks]
Consider the relation R = {A, B, C, D, E, F, G, H, I,J} and the set of functional dependencies F = { {A, B} -> {C}, {A} -> {D, E}, {B} -> {F}, {F} ->{G, H}, {D} -> {I, J} }.

(a) Proof {A}->{E,J} holds by using inference rules. (10 marks)
{A }->{ D, E } (given)
so {A } ->{ D } , {A }- > {E} (by decomposition rule)
And {D} -> {I, J } (given)
since {A} ->{D} ,
{ D }->{ I }and {D }-> { J }, (by decomposition)
so {A }- >{ J } (by transitive rule)
Since { A} - > {E} and {A }-> { J } , we can get {A} -> {E, J } (by union rule)

(b) Whether {A,B,C} is a super key? Whether {A,B,C} is a candidate key? Why? (10 marks)
{A, B, C} is a superkey.
A - > D,
A ->E (by decomposition rule)
A,B -> C, (given)
A->C,
and B->C (by decomposition rule)
B - > F, (given)
F-> G, H
F -> G ,
and F -> H (by decomposition rule)
A -> D(given)
D- > I , J (given)
A -> I , J (by transitive rule)
A-> I
and A -> J (by decompostion rule)
A-> A, B-> B, C-> C (trivial)
Therefore, we can say that {A B C} is a super key because they can determine all other attributes of the relation R.

However, {A,B,C }is not a candidate key.
Because they are not the minimal. They are not the minimal subsets that can
functionally determines all other attributes of R. The minimal subset should be { A , B}.

(c) Whether {A,F} is a super key? Whether {A,F} is a candidate key? Why? (10 marks)

A - > D, A ->E (by decomposition rule)
A,B -> C, (given)
A->C (by decomposition rule)
F-> G, H
F -> G ,
and F -> H (by decomposition rule)
A -> D(given)
D- > I , J (given)
A -> I , J (by transitive rule)
A-> I
and A -> J (by decompostion rule)
B -> F, but it does not mean F- > B.
A-> A, B-> B, C-> C (trivial)
So the closure of {AF }={ A, C, D, E, F , G}

It is not able to functionally determines all other attributes in the R.
So{A, F} is not a super key, and then it would not be the candidate key.
The candidate key requires to satisfy the requirements of superkey first.
Additionally, it has to be the minimal key of the relation R. Failing the superkey requirment
means it is not a superkey and not a candidate key.

(d) Decompose R into 2NF. (10 marks)
F = { {A, B} -> {C}, {A} -> {D, E}, {B} -> {F}, {F} ->{G, H}, {D} -> {I, J} }

To satisfy 2NF, we need to eliminate all partial functional dependency and only keep those Full functional dependency,

A - > D,
A ->E (by decomposition rule)
A,B -> C, (given)
A->C,
and B->C (by decomposition rule)
B - > F, (given)
F-> G, H
F -> G ,
and F -> H (by decomposition rule)
A -> D(given)
D- > I , J (given)
A -> I , J (by transitive rule)
A-> I
and A -> J (by decompostion rule)
A-> A, B-> B, C-> C (trivial)

And then the following 2 relation can satisfy the 2NF:
R1 (A , D, E, C, I , J)
R2 (B, F , G, H)

(e) Decompose R into BCNF. (10 marks)
As we have obtain relations that satisfy the 2NF, we futher reduce the transitive dependency to obtain 3NF.

F = { {A, B} -> {C}, {A} -> {D, E}, {B} -> {F}, {F} ->{G, H}, {D} -> {I, J} }

2NF:
R1 (A , D, E, C, I , J)
R2 (B, F , G, H)

3NF:
R1 (A , D, E, C)
R2 (B, F)
R3(D, I , J)
R4(F, G, H)

To obtain the BCNF, we want every attributes are not functional dependent to primary key.

BCNF:
R1 (A , D, E, C)
R2 (B, F)
R3(D, I , J)
R4(F, G, H)

Since primary keys are single attribute, it also satisfy BCNF.