EECS339 ~~HW~~ PS1 San Lee
slw579

1). An entity is any object in the system that we want to model and store information about. Entities could be ~~either~~ both concrete or abstract recognizable concepts.

2) A weak entity is an entity that cannot be uniquely identified by its attribute alone. It needs a foreign key along with its attribute to create a primary key. An entity is a standalone concept with its own primary key and attribute.

3) Schema is a template that comprises the common attribute of the fields. It provides a blue print of how the database is constructed.

4) A: Atomicity is an indivisible and irreducible series of operation so that either all occur or none occur. This is important because it enables a grmp of operation closely related to take place in order to ensure consistency and maintain correct relationship in relational DB.

C: Consistency requires that any database transaction to be allowed in a certain way to change data. This is like integrity constraint to maintain how the database is managed.

I: Isolation determines how transaction is visible to users and system. This lets different users have different access/read-ability to certain parts of the database.

D: Durability guarantees that transaction that have been committed will survive permanently. I.E the saved data must be consistent even in case of system crash.

5) In DBMS, key is a field used to sort data. One type of key is primary key which must hold a unique value for each record. Another type of key is called foreign key which identifies records in different table.

~~(6)~~

2. (a)

Entity sets:

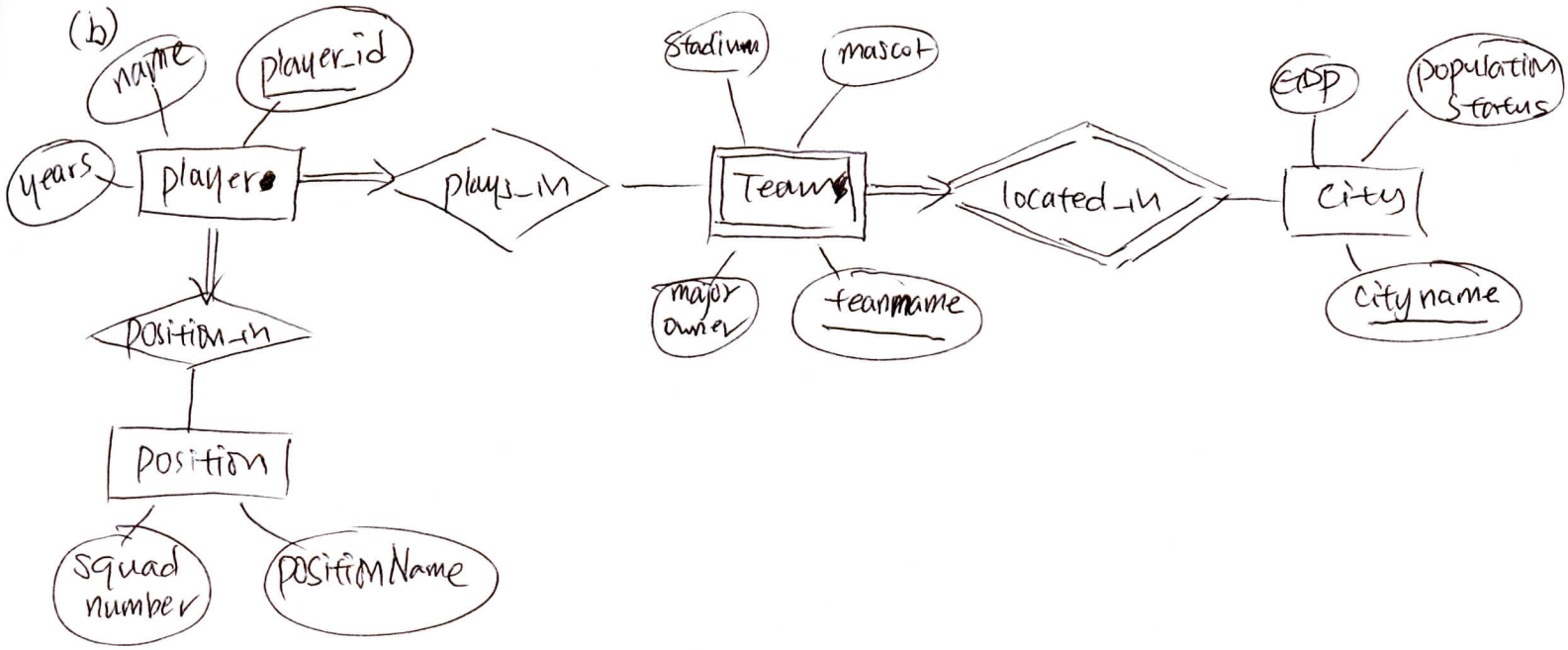
1. Players
2. Team
3. City
4. Position

Relationship sets:

1. plays_in (player, team)
2. located_in (team, city)
3. ~~posit~~ position_in (player, position)

- City needs a cityname as an additional primary identifier.

(b)



(c)

```
CREATE TABLE plays_in (
    player_id    INTEGER
    teamName     CHAR(10)
    cityName     CHAR(10)
    PRIMARY KEY (player_id, teamName)
    FOREIGN KEY (player_id) REFERENCES Players
    FOREIGN KEY (teamName, cityName) REFERENCES Teams )
```

~~CREATE TABLE located_in (teamName CHAR(10) cityName CHAR(10) PRIMARY KEY (teamName, cityName) FOREIGN KEY (teamName) REFERENCES team FOREIGN KEY (cityName) REFERENCES city)~~

```
CREATE TABLE position_in (
    player_id    INTEGER
    position_num INTEGER
    positionName CHAR(10)
    PRIMARY KEY (player_id, positionName)
    FOREIGN KEY (player_id) REFERENCES player)
```

```
CREATE TABLE player (
    player_id    INTEGER
    name         CHAR
    years        INTEGER
    PRIMARY KEY (player_id));
```

```
CREATE TABLE TEAM (
    teamName     CHAR
    Stadium      CHAR
    major_owner  CHAR
    Mascot       CHAR
    CityName     CHAR  NOT NULL
    PRIMARY KEY (teamName, cityName)
    FOREIGN KEY (cityName) REFERENCES city
        ON DELETE CASCADE )
```

```
CREATE TABLE city (
    cityName     CHAR
    GDP          REAL
    population status  INTEGER
    primary KEY (cityName));
```

```
CREATE TABLE Position (
    positionName   CHAR
    SquadNumber    INTEGER
    PRIMARY KEY (positionName))
```

3.1

1) update anomaly

When we have to update Favorite Record of an employee, we have to update the ~~ea~~ Fav. artist of that employee. Also, when updating the service time we have to update the entire row for it.

2) Insert anomaly:

When a new employee is hired, but he/she doesn't have a favorite album, favorite artist has to be NULL.

3) Delete anomaly.

Since favorite record artist is ~~a~~ dependent on favorite ~~artist~~ record as functional dependency, if ~~the~~ a employee doesn't like the ~~album~~ record and deletes it, favorite record needs to be deleted too or else it would cause ~~discrep~~ inconsistency.

3.2)

1. $R(A,B,C)$, $F \Rightarrow \{A \to B, B \to C\}$

⇒ False. Dependent entity (B) is serving as a key in the ~~2~~ second functional dependency.

2. $R_1(A,B)$. $R_2(B,C)$. $F \Rightarrow \{A \to B, B \to C\}$

⇒ True. No redundancy.

3.3)

(a) No. Functional dependency is not seperated from the given table.

(b)

Functional Dependency:

(Fav. Track → Fav. Artist)

In BCNF, we seperate the relationship into two parts.

$R_1 : X \cup Y$

$R_2 : R - Y$

$\therefore R_1$: (Fav. Track, Fav. Artist)

R_2 : (FID, Fav. Track, Location, distance)

by using the functional dependency of (Track → Artist) mentioned above.