RELAZIONE
PROGETTO BASI DI DATI E SISTEMI INFORMATIVI
A.A 2019/2020
CAMPIONATO DI CALCIO

Autori:
Matricola: 7020875 Cognome: Catone Nome: Ilaria
Matricola: 7009348 Cognome: Iervolino Nome: Ludovico
Matricola: 7010856 Cognome: Canocchi Nome: Simone

Note importanti:
Per il corretto funzionamento del progetto è **necessario** cambiare il percorso nel file .sql con il percorso della cartella in cui sono posti i file di popolamento, questo è necessario poiché abbiamo usato il comando "*load data infile*" per popolare i vari file.

Testo del progetto:
L'Italia è nota per il suo seguito verso il calcio, per questo ogni anno è necessario creare campionati di calcio per tutta la nazione, dalle serie maggiori (A, B) fino alle serie Dilettantistiche(C, D).

Ogni squadra è identificata per Nome ed è caratterizzata da Data di Creazione, Città in cui gioca, la Serie in cui gioca nell'anno corrente, lo stadio, gli sponsor, i colori sociali e soprattutto dal Presidente.

Una squadra può essere gestita da un solo Presidente ma un Presidente può possedere più squadre nello stesso momento. Il presidente rende noto solo il codice fiscale, nome e cognome.

Una squadra non può giocare se non possiede in propri giocatori, i quali vengono registrati alla FIGC con il loro numero di tessera univoco e soprattutto che non può essere cambiato. Oltre al numero di tessera di un giocatore conosciamo il nome, cognome, luogo e data di nascita e il ruolo che ricopre. Una volta firmato il contratto tra un giocatore e la squadra, esso deve scegliere il suo numero tra quelli rimasti liberi nella società e concordare lo stipendio che dovrà percepire.

Allo stesso modo, una squadra deve avere un unico allenatore alla guida di essa. Ogni allenatore si identifica con il suo numero di tessera ed in più conosciamo nome, cognome, luogo e data di nascita ed il numero di squadre allenate in carriera(se allena più volte la solita squadra, essa viene contata una sola volta). Quando l'allenatore appone la firma sul contratto tra lui e la squadra, viene concordato lo stipendio fra i due.

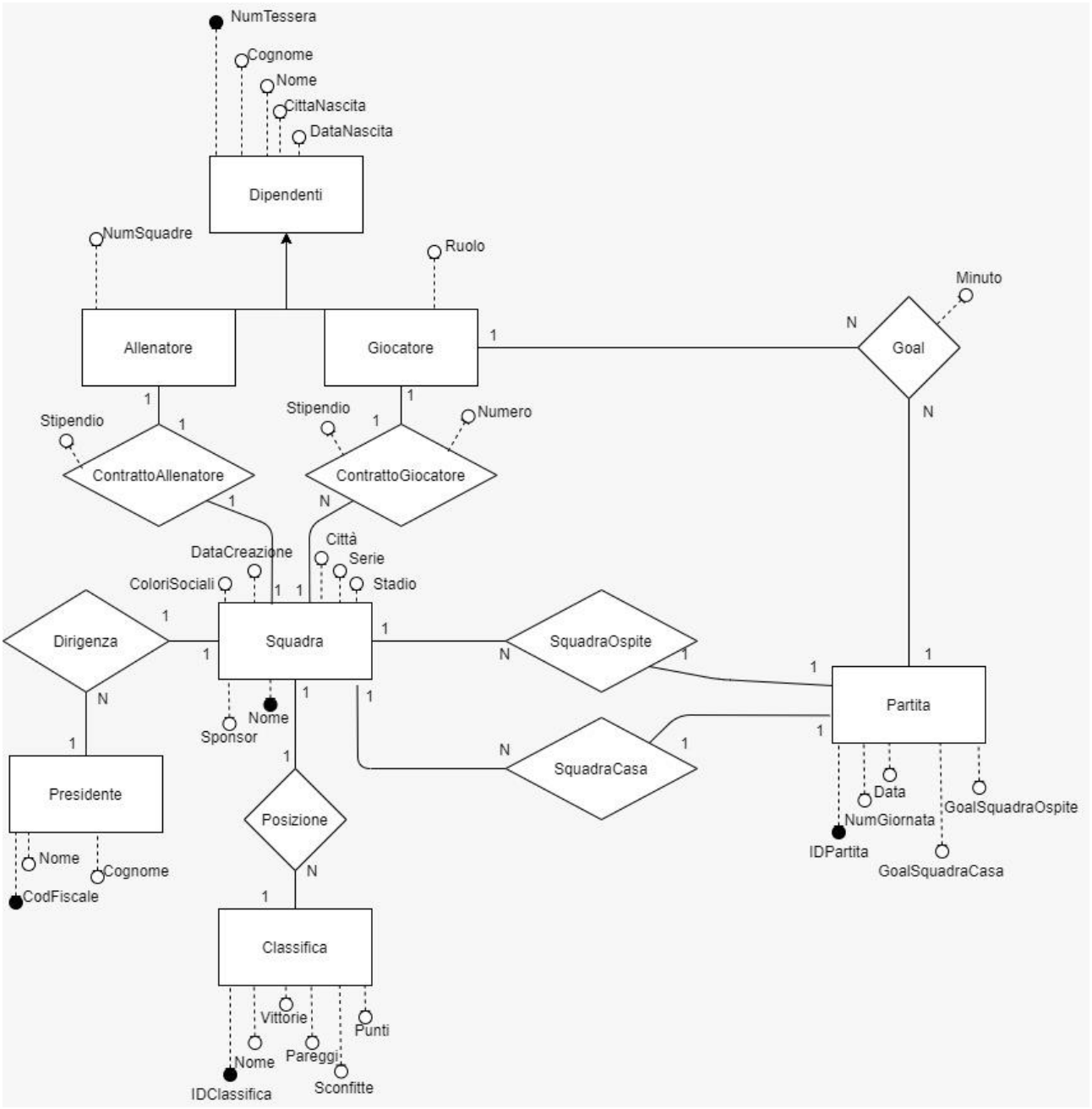
In un campionato di calcio, ogni anno si giocano delle partite, le quali vanno poi a comporre una classifica.
Di ogni partita conosciamo la giornata in cui essa viene giocata e quindi la data, la squadra di casa, la squadra ospite, i loro rispettivi goal segnati e/o subiti.

In una partita, solitamente vengono segnati dei goal, dei quali vogliamo sapere il marcatore e il minuto in cui è stato segnato.

Una volta finita la stagione è possibile stilare una classifica, una per ogni serie, con le varie squadre e i punti da loro totalizzati. In questo caso, noi stiliamo una classifica sullo

storico delle partite giocate dal campione di squadre che vogliamo analizzare.

Diagramma ER iniziale:



Dizionario dei Dati (entità):

ENTITA’	DESCRIZIONE	ATTRIBUTI	IDENTIFICATORE
Presidente	Persona che possiede la squadra	Nome Cognome CodFiscale	CodFiscale
Giocatore	Giocatore della squadra	NumTessera Cognome Nome DataNascita CittaNascita Ruolo	NumTessera
Allenatore	Allenatore della squadra	NumTessera Cognome Nome DataNascita CittaNascita NumSquadre	NumTessera
Squadra	Aggregazione di giocatori	Nome Serie Stadio DataCreazione Citta ColoriSociali Sponsor	Nome
Partita	Confronto tra due squadre	IDPartita NumGiornata GoalSquadraCasa GoalSquadraOspite data	IDPartita
Classifica	Classifica delle squadre (Creato per eventuali espansione del DataBase)	IDClassifica Vittorie Pareggi Sconfitte Punti	IDClassifica

Dizionario delle Relazioni:

RELAZIONI	DESCRIZIONE	COMPONENTI	ATTRIBUTI
Goal	Riepilogo dei dati sulla rete segnata	Giocatore, Partita	Minuto
Gioca	Squadra attuale del giocatore	Giocatore, Squadra	
Allenatore	Squadra attuale dell'allenatore	Allenatore, Squadra	
Posizione	Relativa classifica della squadra all'interno della classifica	Squadra, Classifica	
SquadraCasa	Squadra che gioca in casa	Squadra, Partita	
SquadraOspite	Squadra che gioca in trasferta	Squadra, Partita	

Descrizione dei Vincoli:

VINCOLI INTEGRITA' SUI DATI
(1)La squadra di una determinata classifica può appartenere solo a tale classifica
(2)La squadra di una determinata serie può appartenere solo a tale serie
(3)Una squadra senza allenatore o senza giocatori non può partecipare ad un campionato
(4)Una squadra può essere allenata da solamente un allenatore
(5)Un goal, per aspetti di progettazione, non può essere segnato dopo il 95

PROGETTAZIONE LOGICA:

Ristrutturazione dello Schema E/R

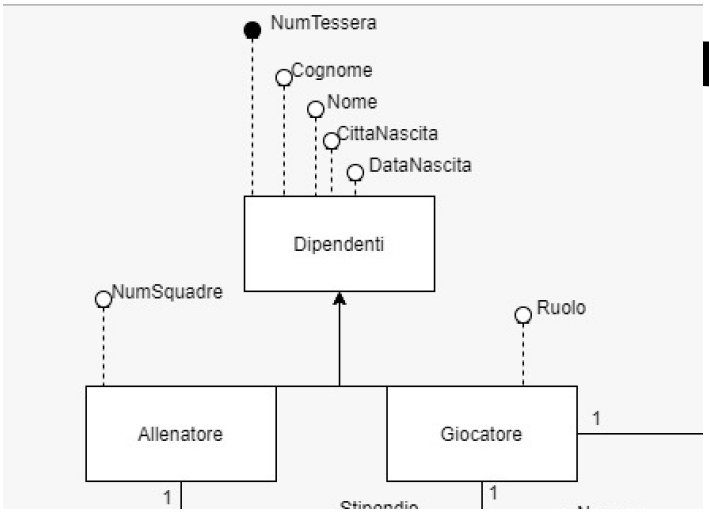
La fase di ristrutturazione di uno schema E/R si può suddividere in una serie di passi da effettuare in sequenza:

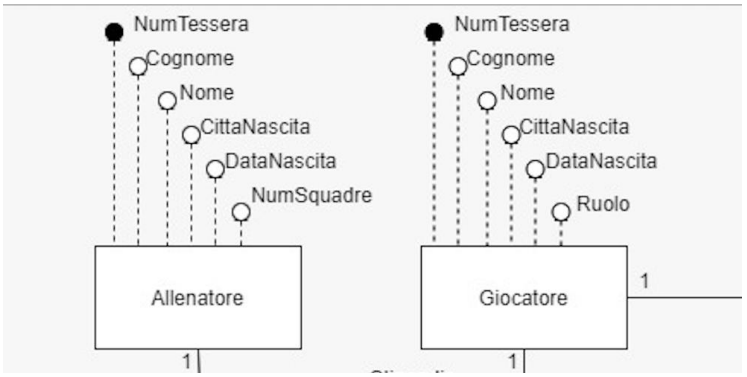
1. Analisi delle ridondanze:

L'attributo punti nell'entità classifica può essere derivato effettuando delle operazioni su sconfitte, vittorie, pareggi. Tuttavia, attraverso un'analisi dei costi, l'occupazione di memoria di "Punti" è più conveniente rispetto all'eliminazione della ridondanza, dato che le operazioni renderebbero meno efficiente il programma dovendo gestire più byte.

2. Eliminazione delle generalizzazioni:

La generalizzazione tra Persona, Giocatore e Allenatore è stata eliminata accorpando il padre nell'entità figlie ereditando tutte le associazioni e gli attributi.





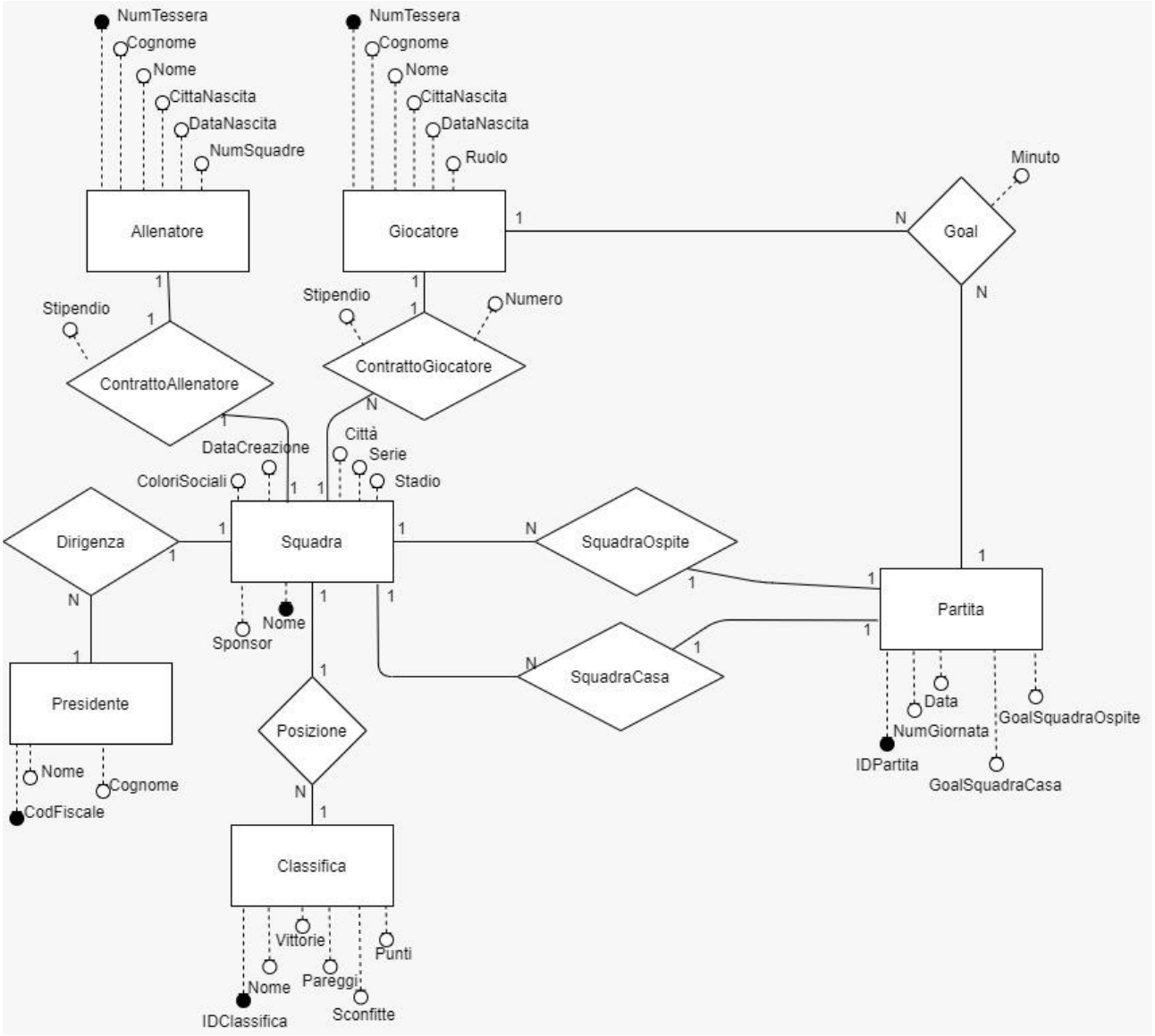
3. Partizione/accorpamento di entità e associazioni:

Il modello E/R realizzato non presenta concetti che debbano essere partizionati o accorpati.

4. Scelta degli identificatori primari:

Le chiavi primarie sono già tutte ben definite e non potranno assumere valori nulli, sono tutte chiavi primarie semplici (costituite da un solo attributo) e tutte costituite da attributi interni.

Schema ER dopo la ristrutturazione:



Modello relazionale:

SCHEMA RELAZIONALE
Presidente(Nome, Cognome, CodFiscale)
Squadra (Nome, DataCreazione, Citta, Serie, Stadio, Sponsor, ColoriSociali, Presidente)
Giocatore (NumTessera, Cognome, Nome, CittaNascita, DataNascita, Ruolo)
ContrattoGiocatore(NumTessera, NomeSquadra, Numero, Stipendio)
Allenatore (NumTessera, Cognome, Nome, CittaNascita, DataNascita, NumSquadre)
ContrattoAllenatore(NumTessera, NomeSquadra, Stipendio)
Partita (IDPartita, NumGiornata, SquadraCasa, GoalSquadraCasa, SquadraOspite, GoalSquadraOspite, data)

Goal (IDPartita, Minuto, Marcatore)
Classifica (IDClassifica, Nome, Vittorie, Pareggi, Sconfitte, Punti)

Vincoli:

Squadra.Presidente->Presidente.CodFiscale
ContrattoGiocatore.NumTessera -> Giocatore.NumTessera
ContrattoGiocatore.NomeSquadra -> Squadra.Nome
ContrattoAllenatore.NumTessera -> Allenatore.NumTessera
ContrattoAllenatore.NomeSquadra -> Squadra.Nome
Partita.SquadraCasa -> Squadra.Nome
Partita.SquadraOspite -> Squadra.Nome
Goal.IDPartita -> Partita.IDPartita
Goal.Marcatore -> Giocatore.NumTessera
Classifica.Nome -> Squadra.Nome

Progettazione Fisica:

Squadra	
Nome	char(15)
DataCreazione	date
Citta	char(15)
Serie	Enum
Stadio	char(20)
Sponsor	char(20)
ColoriSociali	char(20)
Presidente	char(16)

Presidente	
Nome	char(15)
Cognome	char(15)
CodFiscale	char(16)

Giocatore	
NumTessera	int
Cognome	char(20)
Nome	char(15)
CittaNascita	char(15)
DataNascita	date

Ruolo	char(20)

ContrattoGiocatore	
NumTessera	int
NomeSquadra	char(15)
Stipendio	Decimal(8,2)
Numero	int

ContrattoAllenatore	
NumTessera	int
NomeSquadra	char(15)
Stipendio	Decimal(8,2)

Allenatore	
NumTessera	int
Cognome	char(20)
Nome	char(15)
CittaNascita	char(15)
DataNascita	date
NumSquadre	int

Partita	
IDPartita	int
NumGiornata	int
SquadraCasa	char(15)
GoalSquadraCasa	int
SquadraOspite	char(15)
GoalSquadraOspite	int
data	date

Goal	
IDPartita	int
Minuto	int
Marcatore	int

Classifica	
IDClassifica	int
Nome	Char(15)
Vittorie	tinyInt
Pareggi	tinyInt
Sconfitte	tinyInt
Punti	int

Indici:

TABELLA	INDICE
Squadra	Nome
Giocatore	NumTessera
Allenatore	NumTessera
Partita	IDPartita
Goal	IDPartita
Classifica	IDClassifica
ContrattoGiocatore	NumTessera
ContrattoAllenatore	NumTessera

Implementazione SQL:

•Creazione e popolamento delle tabelle

```
DROP DATABASE IF EXISTS Campionato;
create database if not exists Campionato;
use Campionato;

create table if not exists Presidente(
  Nome char(15),
  Cognome char(15),
  CodFiscale char(16) primary key
) ENGINE=INNODB;

create table if not exists Squadra (
    Nome char(15) UNIQUE PRIMARY KEY,
    DataCreazione date,
    Citta char(15) NOT NULL,
    Serie ENUM('A' , 'B', 'C', 'D'),
    Stadio char(20) NOT NULL,
    Sponsor char(20) NOT NULL,
    ColoriSociali char(20) NOT NULL,
  Presidente char(16) NOT NULL,
  foreign key (Presidente)
        references Presidente (CodFiscale)
)  ENGINE=INNODB;

create table if not exists Giocatore (
    NumTessera INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,
    Cognome char(20) NOT NULL,
    Nome char(15) NOT NULL,
    CittaNascita char(15) NOT NULL,
    DataNascita date NOT NULL,
    Ruolo char(20) NOT NULL
)  ENGINE=INNODB;

create table if not exists Allenatore (
    NumTessera INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,
    Cognome char(20) NOT NULL,
    Nome char(15) NOT NULL,
    CittaNascita char(15) NOT NULL,
    DataNascita date NOT NULL,
    NumSquadre int DEFAULT '0'
)  ENGINE=INNODB;
```

```sql
CREATE TABLE IF NOT EXISTS ContrattoGiocatore(
  NumTessera int primary key,
  NomeSquadra char(15),
  FOREIGN KEY (NumTessera) REFERENCES Giocatore(NumTessera),
  FOREIGN KEY (NomeSquadra) REFERENCES Squadra(Nome),
  Numero int CHECK (Numero BETWEEN '1' AND '99'),
  stipendio decimal(8, 2)
)ENGINE=INNODB;

CREATE TABLE IF NOT EXISTS ContrattoAllenatore(
  NumTessera int primary key,
  NomeSquadra char(15),
  FOREIGN KEY (NumTessera) REFERENCES Allenatore(NumTessera),
  FOREIGN KEY (NomeSquadra) REFERENCES Squadra(Nome),
  stipendio decimal(8, 2)
)ENGINE=INNODB;

create table if not exists Partita (
    IDPartita INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,
    NumGiornata int NOT NULL,
    SquadraCasa char(15) NOT NULL,
    foreign key (SquadraCasa)
        references Squadra (Nome),
    GoalSquadraCasa int NOT NULL DEFAULT '0',
    SquadraOspite char(15) NOT NULL,
    foreign key (SquadraOspite)
        references Squadra (Nome),
    GoalSquadraOspite int NOT NULL DEFAULT '0',
    data date
)  ENGINE=INNODB;

create table if not exists Goal (
    IDPartita int NOT NULL,
    Minuto int NOT NULL,
    Marcatore int,
  primary key(IDPartita),
    foreign key (IDPartita) references Partita (IDPartita),
    foreign key (Marcatore) references Giocatore (NumTessera)
)  ENGINE=INNODB;
```

```sql
create table if not exists Classifica (
  IDClassifica INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  Nome char(15) NOT NULL,
  Vittorie TINYINT DEFAULT '0',
  Pareggi TINYINT DEFAULT '0',
  Sconfitte TINYINT DEFAULT '0',
  Punti TINYINT NOT NULL
) ENGINE= INNODB;
ALTER TABLE Classifica ADD FOREIGN KEY (Nome) REFERENCES Squadra (Nome) ON DELETE CASCADE ON UPDATE CASCADE;
```

```
################  Creazione istanza: popolamento database  ################
###########################################################################
load data local infile 'C:/Users/simonecanocchi/Desktop/Presidente.csv'
into table Presidente
fields terminated by ','
ignore 2 lines
(Nome, Cognome, CodFiscale);

load data local infile 'C:/Users/simonecanocchi/Desktop/Squadra.csv'
into table Squadra
fields terminated by ','
ignore 2 lines
(Nome, DataCreazione, Citta, Serie, Stadio, Sponsor, ColoriSociali, Presidente);

load data local infile 'C:/Users/simonecanocchi/Desktop/Giocatore.csv'
into table Giocatore
fields terminated by ','
ignore 2 lines
(NumTessera, Cognome, Nome, CittaNascita, DataNascita, Ruolo);

load data local infile 'C:/Users/simonecanocchi/Desktop/Allenatore.csv'
into table Allenatore
fields terminated by ','
ignore 2 lines
(NumTessera, Cognome, Nome, CittaNascita, DataNascita, NumSquadre);

load data local infile 'C:/Users/simonecanocchi/Desktop/Partita.csv'
into table Partita
fields terminated by ','
ignore 2 lines
(IDPartita, NumGiornata, SquadraCasa, GoalSquadraCasa, SquadraOspite, GoalSquadraOspite, data);

load data local infile 'C:/Users/simonecanocchi/Desktop/Goal.csv'
into table Goal
fields terminated by ','
ignore 2 lines
(IDPartita, Minuto, Marcatore);
```

```
load data local infile 'C:/Users/simonecanocchi/Desktop/ContrattoGiocatore.csv'
into table ContrattoGiocatore
fields terminated by ','
ignore 2 lines
(NumTessera, NomeSquadra, Stipendio, Numero);

load data local infile 'C:/Users/simonecanocchi/Desktop/ContrattoAllenatore.csv'
into table ContrattoAllenatore
fields terminated by ','
ignore 2 lines
(NumTessera, NomeSquadra, Stipendio);

INSERT INTO Partita
SET IDPartita='18',NumGiornata='22', SquadraCasa='Fiorentina', GoalSquadraCasa='1', SquadraOspite='Torino', GoalSquadraOspite='0', data='2019/03/22';

INSERT INTO Goal
SET IDPartita='18', Minuto='12',Marcatore='1';

INSERT INTO Presidente(Nome, Cognome, CodFiscale)
VALUES ('Mario', 'Maresca', 'PLXCYR55L21A866Q');

INSERT INTO Squadra(Nome, DataCreazione, Citta, Serie, Stadio, Sponsor, ColoriSociali, Presidente)
VALUES ('Antella', '1999/06/27', 'Antella', 'D', 'Pulicciano', 'MBA', 'Bianco Azzurro', 'PLXCYR55L21A866Q');
```

•Trigger: Dalla sua attivazione permette di controllare in maniera autonoma che ogni nuovo inserimento tra i cannonieri sia compreso tra il minuto 0 e 95. Quelli inseriti al di fuori di questo inervallo vengono memorizzati rispettivamente come valore minimo e massimo.

```
###############  Ulteriori vincoli tramite viste e/o trigger ###############
########################################################################
DELIMITER $$
CREATE TRIGGER ControlloMinuto
  BEFORE INSERT ON Goal
  FOR EACH ROW
BEGIN
  IF NEW.Minuto < 1 THEN
    SET NEW.Minuto = 1;
  ELSEIF NEW.Minuto > 95 THEN
    SET NEW.Minuto = 95;
  END IF;
END $$
DELIMITER ;

create view Cannonieri(Nome,Cognome,NumTessera,Squadra,NumGoal) as
select Nome, Cognome, Giocatore.NumTessera, ContrattoGiocatore.NomeSquadra, count(*)
from Giocatore join ContrattoGiocatore on (Giocatore.NumTessera=ContrattoGiocatore.NumTessera) join Goal on Giocatore.NumTessera=Goal.Marcatore
group by NomeSquadra, Marcatore;

create view Vittorie(Squadra) as
select SquadraCasa as squadra
from Partita
where GoalSquadraCasa>GoalSquadraOspite
union all
select SquadraOspite as squadra
from Partita
where GoalSquadraCasa<GoalSquadraOspite;
```

•Viste:

1. Rappresento in cannonieri tutti i giocatori che hanno fatto almeno un goal con i rispettivi dati passati attraverso l'operatore count.

```
create view Cannonieri(Nome,Cognome,NumTessera,Squadra,NumGoal) as
select Nome, Cognome, Giocatore.NumTessera, ContrattoGiocatore.NomeSquadra, count(*)
from Giocatore join ContrattoGiocatore on (Giocatore.NumTessera=ContrattoGiocatore.NumTessera) join Goal on Giocatore.NumTessera=Goal.Marcatore
group by NomeSquadra, Marcatore;
```

2. Seleziono tutte le squadre che hanno vinto almeno una partita.

```
create view Vittorie(Squadra) as
select SquadraCasa as squadra
from Partita
where GoalSquadraCasa>GoalSquadraOspite
union all
select SquadraOspite as squadra
from Partita
where GoalSquadraCasa<GoalSquadraOspite;
```

•Procedure:

1. Creazione della classifica dichiarando due variabili locali:
“lastTeam” con cui verifichiamo se la squadra è l’ultima della classifica, attraverso l’utilizzo di un cursore o meno e “_teamName” per il nome della squadra. Utilizzo un cursore per scorrere la classifica i cui valori sono dati dal ciclo all’interno della procedura.

```
DELIMITER $$
CREATE PROCEDURE SetupClassifica()
BEGIN

  DECLARE lastTeam BOOLEAN DEFAULT FALSE;
  DECLARE _teamName char(15);

  DECLARE teamCursor CURSOR FOR
    SELECT Squadra.Nome
    FROM   Squadra;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET lastTeam = TRUE;

  OPEN teamCursor;
  teamLoop: LOOP

    FETCH teamCursor INTO _teamName;
    IF (lastTeam) THEN
      LEAVE teamLoop;
    END IF;

    INSERT INTO Classifica(Nome, vittorie, Pareggi, Sconfitte, Punti) VALUES
      (_teamName, 0, 0, 0, 0);
  END LOOP teamLoop;
  CLOSE teamCursor;
END $$
```

2. Stampa di una classifica e di una giornata.

```
CREATE PROCEDURE StampaClassifica()
BEGIN
  SELECT   Squadra.Nome, Classifica.Vittorie AS V, Classifica.Pareggi AS P, Classifica.Sconfitte AS S, Classifica.Punti AS TP
  FROM     Classifica INNER JOIN Squadra ON Classifica.Nome = Squadra.Nome
  WHERE    Serie= 'B'
  ORDER BY Classifica.Punti DESC, Classifica.Vittorie DESC, Classifica.Pareggi DESC;
END $$

CREATE PROCEDURE StampaGiornata(IN Giornata INT)
BEGIN
  SELECT   NumGiornata, SquadraCasa, GoalSquadraCasa, SquadraOspite, GoalSquadraOspite
  FROM     Partita
  WHERE    Partita.NumGiornata = Giornata
  ORDER BY NumGiornata ASC;
END $$
```

3. Richiama tutte le procedure in modo da avere la classifica vuota e se ne effettua una stampa per verificare il corretto svolgimento della procedura. Successivamente, viene giocata la giornata di campionato e ne viene effettuata la stampa, stampa a sua volta fatta sulla classifica aggiornata dopo la partita.

```
CREATE PROCEDURE SetClassifica(IN _day TINYINT UNSIGNED)
BEGIN
  -- Dichiara le variabili utilizzate.
  DECLARE lastMatch BOOLEAN DEFAULT FALSE;
  DECLARE _matchId INTEGER UNSIGNED;
   DECLARE _team1Id, _team2Id CHAR(15);
  DECLARE _goals1, _goals2 TINYINT UNSIGNED;
  -- Dichiara i cursori e gli handler utilizzati.
  DECLARE matchCursor CURSOR FOR
    SELECT IDPartita, SquadraCasa, SquadraOspite,GoalSquadraCasa, GoalSquadraOspite
    FROM   Partita
    WHERE  NumGiornata = _day;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET lastMatch = TRUE;
  -- Per ogni partita...
  OPEN matchCursor;
  matchLoop: LOOP
    -- ...prova a recuperare la chiave primaria della partita e delle squadre in
    -- casa e fuori casa, uscendo dal loop se era l'ultimo record;
    FETCH matchCursor INTO _matchId, _team1Id, _team2Id, _goals1, _goals2;
    IF (lastMatch) THEN
      LEAVE matchLoop;
    END IF;

    IF (_goals1 > _goals2) THEN
      UPDATE Classifica
      SET    Vittorie = Vittorie + 1,
             Punti = Punti + 3
      WHERE  Classifica.Nome = _team1Id;
      UPDATE Classifica
      SET    Sconfitte = Sconfitte + 1
      WHERE  Classifica.Nome = _team2Id;
    ELSEIF (_goals1 = _goals2) THEN
      UPDATE Classifica
      SET    Pareggi = Pareggi + 1,
             Punti = Punti + 1
      WHERE  Classifica.Nome = _team1Id;
      UPDATE Classifica
      SET    Pareggi = Pareggi + 1,
             Punti = Punti + 1
      WHERE  Classifica.Nome = _team2Id;
```

```
    ELSE
      UPDATE Classifica
      SET    Sconfitte = Sconfitte + 1
      WHERE  Classifica.Nome = _team1Id;
      UPDATE Classifica
      SET    Vittorie = Vittorie + 1,
             Punti = Punti + 3
      WHERE  Classifica.Nome = _team2Id;
    END IF;
  END LOOP matchLoop;
  CLOSE matchCursor;
END $$

CREATE PROCEDURE Gioca()
BEGIN
  CALL SetupClassifica();
  CALL StampaClassifica();
  CALL StampaGiornata(31);
  CALL SetClassifica(31);
  CALL StampaClassifica();
END $$
DELIMITER ;

CALL Gioca();
```