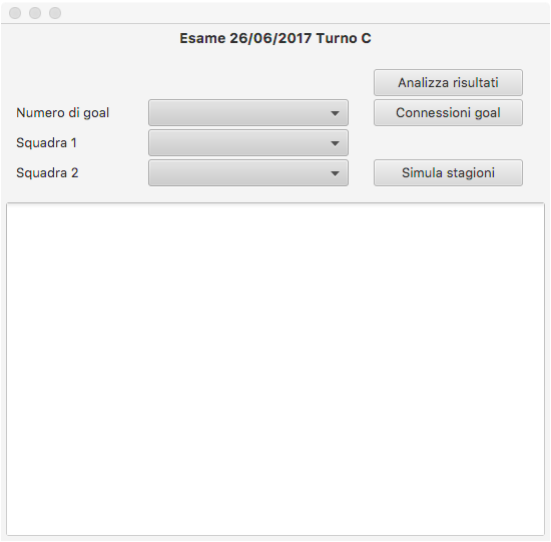
Prova d'esame del 26/06/2017 – Turno C

Si consideri il database "serie_a", contenente i risultati di tutte le partite di calcio della Serie A italiana, tra la stagione 2002/2003 e la stagione 2016/2017. Il database (tratto dal sito http://www.football-data.co.uk/italym.php) è strutturato secondo il diagramma ER della pagina seguente.

Si intende costruire un'applicazione JavaFX che permetta di interrogare tale base dati, e calcolare informazioni a proposito delle gare disputate. L'applicazione dovrà svolgere le seguenti funzioni:



PUNTO 1

a. Permettere all'utente di selezionare il bottone "Analizza risultati".
b. Alla pressione del bottone, creare un grafo che rappresenti i risultati (in termini di punteggio) delle partite giocate da ogni coppia di squadre, in tutte le stagioni disponibili. Il grafo dovrà essere orientato e pesato, con i vertici che rappresentino i numeri di *Goal* fatti (es: 0, 1, 2, 3, …), e gli archi il fatto le partite siano terminate con il risultato dato dai due vertici adiacenti. Ad esempio, l'arco tra il vertice 3 ed il vertice 1 rappresenta tutte le partite che sono terminate con il risultato di 3 ad 1. Il peso dell'arco deve rappresentare il *numero di partite* che avevano tale risultato (in ogni stagione).
c. Permettere all'utente di selezionare, da un menu a tendina, il numero di goal corrispondente ad uno dei nodi presenti nel grafo, e premere il bottone "Connessioni Goal".
d. Per il numero di Goal selezionato nel menu a tendina, stampare l'elenco dei risultati in cui tali goal sono stati segnati in casa (ed il numero di partite corrispondenti), in ordine decrescente di numero di partite.

PUNTO 2

a. Permettere all'utente di selezionare due squadre dai rispettivi menu a tendina, e di premere il bottone "Simula stagioni". Si vuole simulare l'evoluzione del numero di tifosi delle due squadre al passare delle diverse stagioni.
b. Alla pressione del bottone, si simuli il *numero di tifosi* appartenenti alle due squadre. Il simulatore deve considerare le partite giocate in tutte le stagioni, in ordine crescente di data, nelle quali abbiano giocato *tutte e due* le squadre selezionate (tendenzialmente vi saranno due partite per stagione, una per l'andata ed una per il ritorno). (Nota: il grafo creato al primo punto non risulta utile ai fini della simulazione e non è necessario costruirne un altro).
c. All'inizio della simulazione, si ipotizzi che ciascuna squadra abbia lo stesso numero di tifosi T, con T=1000.
d. Considerare solo le partite in cui giocano tutte e due le squadre insieme. Il risultato di ogni partita "simulata" dipenderà dal risultato "storico", ossia quello presente nel database, corretto da un fattore che tiene conto del numero di tifosi. Infatti, in presenza di un maggior numero di tifosi è più probabile vincere. In particolare, se la squadra che ha più tifosi ha vinto la partita "storica", allora non ci sarà alcun cambiamento. Invece, se la squadra che ha più tifosi ha perso la partita "storica", allora con probabilità del 50% il risultato sarà *ribaltato* (es da 3-2 diventerà 2-3). Nel restante 50% dei casi il risultato rimane invariato.
e. Al termine di ogni partita, alcuni dei tifosi della squadra che ha perso decidono di passare alla squadra vincente. La percentuale di tifosi che cambierà squadra è proporzionale allo scarto-reti moltiplicato per un fattore P (esempio: P=10). Ad esempio, se una il risultato fosse A-B=3-1, lo scarto reti è pari a 2, ed il 2*10=20% dei tifosi della squadra B passerà alla squadra A. La squadra che vince la partita non perde alcun tifoso (ma ne guadagnerà). In caso di pareggio, nessun tifoso si sposta.

f. Al termine della simulazione, si stampi il numero di tifosi di ciascuna squadra ed il numero di partire il cui risultato è stato ribaltato.

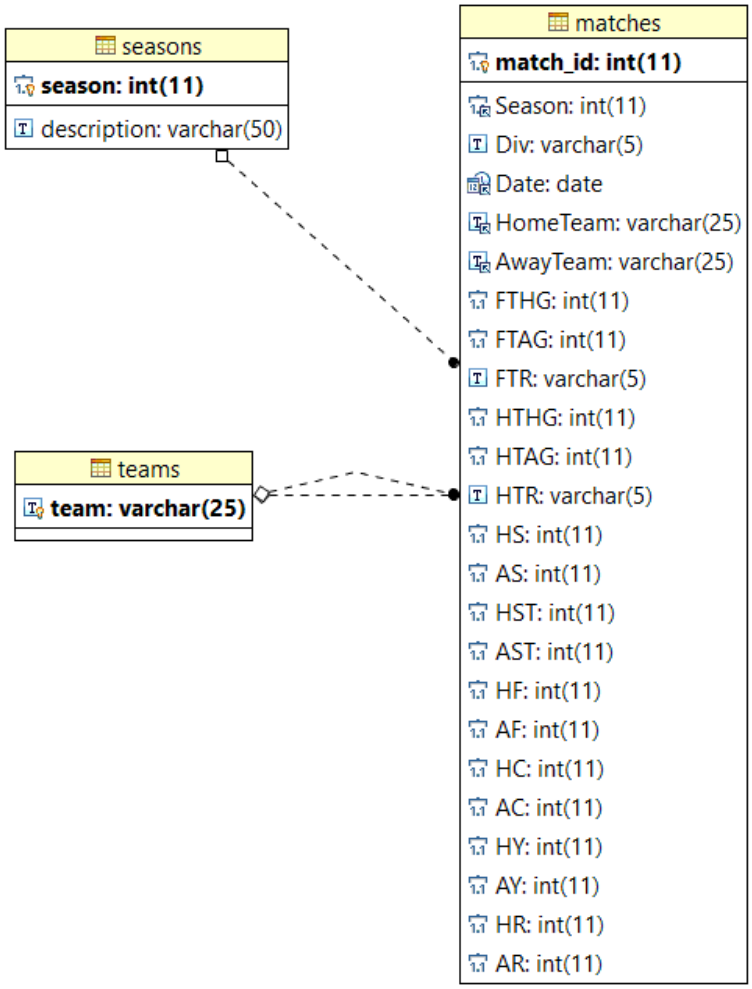
Nella realizzazione del codice, si lavori a partire dalle classi (Bean e DAO, FXML) e dal database contenuti nel progetto di base. È ovviamente permesso aggiungere o modificare classi e metodi.

Tutti i possibili errori di immissione, validazione dati, accesso al database, ed algoritmici devono essere gestiti, non sono ammesse eccezioni generate dal programma.

Legenda (tabella `matches`):

- `Season` = Season year (foreign key to table `seasons`)
- `Div` = League Division
- `Date` = Match Date
- `HomeTeam` = Home Team (foreign key to table `teams`)
- `AwayTeam` = Away Team (foreign key to table `teams`)
- `FTHG` = Full Time Home Team Goals
- `FTAG` = Full Time Away Team Goals
- `FTR` = Full Time Result (**H**=Home Win, **D**=Draw, **A**=Away Win)
- `HTHG` = Half Time(*) Home Team Goals
- `HTAG` = Half Time(*) Away Team Goals
- `HTR` = Half Time(*) Result (**H**=Home Win, **D**=Draw, **A**=Away Win)

(*) tranne quanto la partita è assegnata a tavolino



Parametri aggiuntivi (non presenti in tutte le stagioni):

- HS = Home Team Shots
- AS = Away Team Shots
- HST = Home Team Shots on Target
- AST = Away Team Shots on Target
- HHW = Home Team Hit Woodwork
- AHW = Away Team Hit Woodwork
- HC = Home Team Corners
- AC = Away Team Corners
- HF = Home Team Fouls Committed
- AF = Away Team Fouls Committed
- HO = Home Team Offsides
- AO = Away Team Offsides
- HY = Home Team Yellow Cards
- AY = Away Team Yellow Cards
- HR = Home Team Red Cards
- AR = Away Team Red Cards