

## Progettazione

### Introduzione

Si vuole realizzare una base di dati dedicata alla gestione completa di un torneo maschile di basket, che prevede la partecipazione di 24 squadre, suddivise in quattro gironi da sei squadre ciascuno. La base di dati sarà progettata per gestire in modo efficiente e accurato tutte le informazioni riguardanti i giocatori, le squadre partecipanti, le partite in programma, gli arbitri coinvolti e la vendita dei biglietti per gli spettatori.

### Requisiti della base di dati

Al torneo partecipano 24 squadre. Per ogni squadra si vuole registrare il nome, la data di iscrizione, se la quota è già stata versata (la quota è fissa e uguale per ogni squadra, non dipende dal numero di giocatori) e il colore predominante della loro divisa. Inoltre, al momento dell'iscrizione, ad ogni squadra sarà associato un codice univoco.

Le squadre vengono raggruppate automaticamente in quattro gironi da sei sulla base dell'ordine di iscrizione (le prime 6 squadre nel primo girone e così via)

Ogni squadra è composta da un minimo di 6 e un massimo di 10 giocatori, che devono obbligatoriamente avere almeno 18 anni. Per ogni giocatore si vogliono registrare nome, cognome, numero di telefono, email, data di nascita, dove risiede (indirizzo e CAP), se è tesserato FIP o meno e la data del certificato medico (deve essere valido, cioè fatto nell'ultimo anno). Ogni giocatore è identificato dal codice fiscale e può partecipare al torneo con una sola squadra. Inoltre, ogni giocatore ha un numero di maglia che deve essere diverso da quelli degli altri componenti della stessa squadra.

Ogni squadra gioca 5 partite (contro tutte le altre squadre del proprio girone). Ogni partita è giocata da due squadre (una in casa e una in trasferta). Le partite vengono inizialmente inserite in calendario una volta chiuse le iscrizioni, mentre i punteggi di entrambe le squadre vengono inseriti una volta che la partita è finita. Della partita (che è identificata da un codice univoco) si vuole registrare la data e orario, il palazzetto in cui viene giocata, l'arbitro che dirigerà la partita e il numero di biglietti venduti. Inoltre, ogni partita assegna 2 punti in classifica alla squadra che vince (quella col punteggio maggiore). Le partite non possono terminare con un pareggio.

Per ogni partita si vogliono anche registrare le statistiche (principali) di ciascun giocatore che gioca tale partita, cioè minuti giocati, punti, assist e rimbalzi. Se un giocatore non si presenta alla partita o comunque non fa il suo ingresso in campo, le sue statistiche non vengono registrate per quella partita.

Degli arbitri si vogliono conoscere gli stessi dati anagrafici del giocatore, e in più si registra la data del patentino e il comitato arbitrale a cui appartiene.

Come detto, ogni partita si gioca in un palazzetto. Il palazzetto ha un nome (tutti diversi tra di loro), una capienza massima di spettatori, appartiene a una società, e ha una locazione (CAP+indirizzo).

Ad ogni partita possono assistere degli spettatori, che possono comprare biglietti anche per più partite. Dello spettatore si vuole conoscere soltanto nome, cognome, email e numero di telefono. Infine, ogni biglietto è identificato da un codice, è valido per una singola partita ed è associato ad un posto a sedere e ad un prezzo.

## Glossario dei termini:

Termine	Definizione	Collegamenti
Persona	Individuo che prende parte al torneo (come giocatore o arbitro)	Giocatore, Arbitro
Giocatore	Persona che partecipa al torneo come membro di una squadra	Persona, Squadra, Statistica
Arbitro	Persona che dirige una partita	Persona, Partita
Partita	Incontro tra due squadre dello stesso girone	Statistica, Palazzetto, Biglietto, Arbitro, Squadra
Squadra	Gruppo di giocatori che competono insieme nel torneo	Giocatore, Girone, Partita
Girone	Gruppo di 6 squadre che si affrontano tra di loro	Classifica
Statistica	Prestazione di un giocatore nella singola partita	Giocatore, Partita
Palazzetto	Luogo dove viene giocata la partita	Partita
Biglietto	Documento che concede l'accesso a uno spettatore a una specifica partita del torneo	Partita, Spettatore
Spettatore	Persona che assiste a una partita del torneo	Biglietto

## **Suddivisione dei requisiti in frasi omogenee**

### Frasi sui giocatori:

“Ogni squadra è composta da un minimo di 6 e un massimo di 10 giocatori, che devono obbligatoriamente avere almeno 18 anni”

“Per ogni giocatore si vogliono registrare nome, cognome, numero di telefono, email, data di nascita, dove risiedono (indirizzo e CAP), se è tesserato FIP o meno e la data del certificato medico valido (deve essere valido, cioè fatto nell'ultimo anno)”

“Ogni giocatore è identificato dal codice fiscale e può partecipare al torneo con una sola squadra.”

“Inoltre, ogni giocatore ha un numero di maglia che deve essere diverso da quelli degli altri componenti della stessa squadra.”

“Per ogni partita si vogliono anche registrare le statistiche (principali) di ciascun giocatore che gioca tale partita, cioè minuti giocati, punti, assist e rimbalzi.”

“Se un giocatore non si presenta alla partita o comunque non fa il suo ingresso in campo, le sue statistiche non vengono registrate”

### Frasi sulle squadre:

“Per ogni squadra si vuole registrare il nome, la data di iscrizione, se la quota è già stata versata (la quota è fissa e uguale per ogni squadra, non dipende dal numero di giocatori) e il colore predominante della loro divisa”

“Inoltre, al momento dell'iscrizione, ad ogni squadra sarà associato un codice univoco.”

“Le squadre vengono raggruppate automaticamente in quattro gironi da sei sulla base dell'ordine di iscrizione (le prime 6 squadre nel primo girone e così via)”

“Ogni squadra è composta da un minimo di 5 e un massimo di 10 giocatori”

“Ogni squadra gioca 5 partite (contro tutte le altre squadre del proprio girone).”

### Frasi sulle partite:

“Ogni partita è giocata da due squadre (una in casa e una in trasferta)”

“Le partite vengono inizialmente inserite in calendario una volta chiuse le iscrizioni, mentre i punteggi di entrambe le squadre vengono inseriti una volta che la partita è finita.”

“Della partita (che è identificata da un codice univoco) si vuole registrare la data e orario, il palazzetto in cui viene giocata, l'arbitro che dirigerà la partita e il numero di biglietti venduti. Inoltre, ogni partita assegna 2 punti in classifica alla squadra che vince (quella col punteggio maggiore)”

“Le partite non possono terminare con un pareggio.”

“Per ogni partita si vogliono anche registrare le statistiche (principali) di ciascun giocatore che gioca tale partita, cioè minuti giocati, punti, assist e rimbalzi”

“ogni partita si gioca in un palazzetto”

“Ad ogni partita possono assistere degli spettatori”

### Frasi sugli arbitri:

“Degli arbitri si vogliono conoscere gli stessi dati anagrafici del giocatore, e in più si registra la data del patentino e il comitato arbitrale a cui appartiene.”

### Frasi sui palazzetti:

"Il palazzetto ha un nome (tutti diversi tra di loro), una capienza massima di spettatori, e ha una locazione (CAP+indirizzo)."

#### Frasi sui biglietti:

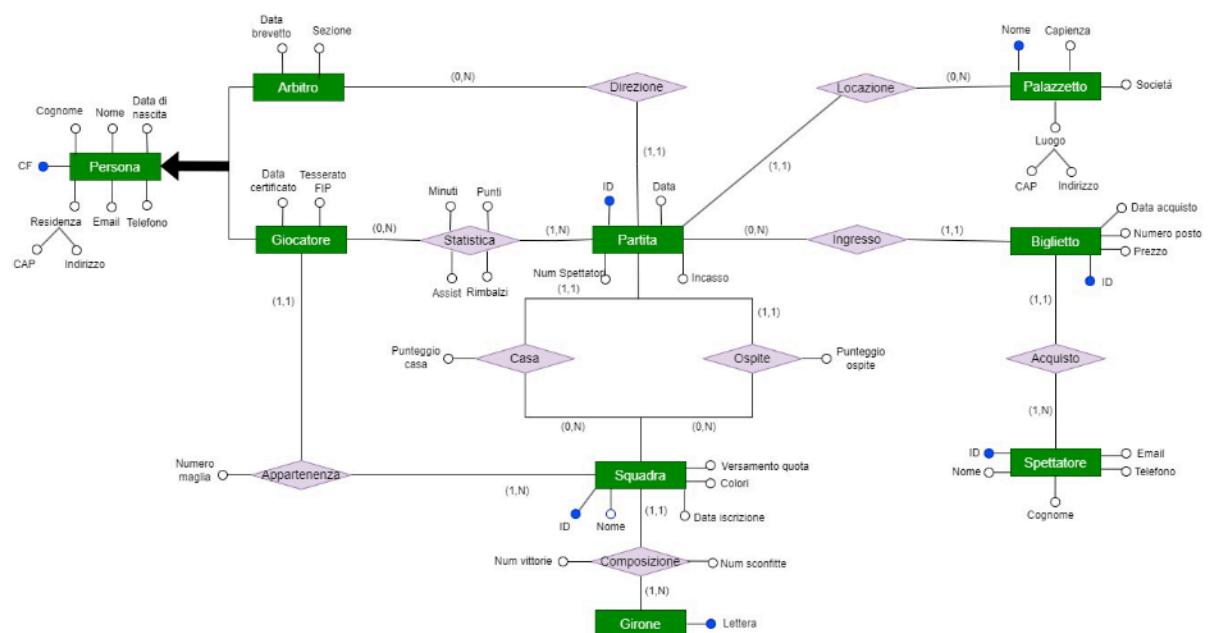
"Ad ogni partita possono assistere degli spettatori, che possono comprare biglietti anche per più partite"

"Infine, ogni biglietto è identificato da un codice, è valido per una singola partita ed è associato ad un posto a sedere e ad un prezzo."

#### Frasi sugli spettatori:

"Dello spettatore si vuole conoscere soltanto nome, cognome, email e numero di telefono"

## Diagramma Entity - Relationship



## Dizionario dei dati (Entità)

Entità	Descrizione	Attributi	Identificatore
Persona	Individuo che partecipa al torneo	CF, Nome, Cognome, Data di nascita, Residenza, Email, Telefono	CF
Giocatore	Membro della squadra	Data certificato, Tesserato FIP	-

Squadra	Squadra partecipante al torneo	ID, Nome, Data iscrizione, Versamento quota, Colori	ID
Partita	Incontro tra due squadre	ID, Data, Incasso, Num Spettatori	ID
Girone	Insieme di 6 squadre che si affrontano tra di loro	Lettera	Lettera
Arbitro	Persona che dirige una partita	Data patentino, Sezione	-
Palazzetto	Luogo dove si svolge una partita	Nome, Capienza, Luogo, Società	Nome
Biglietto	Strumento che permette allo spettatore di assistere alla partita	ID, Numero posto, Prezzo, Data acquisto	ID
Spettatore	Persona che assiste ad una partita	ID, Nome, Cognome, Email, Telefono	ID

## Dizionario dei dati (Relazioni)

Relazione	Descrizione	Composizione	Attributi
Appartenenza	Associa un giocatore alla squadra di cui fa parte	Giocatore, Squadra	Numero maglia
Statistica	Associa al giocatore la prestazione realizzata in una data partita	Giocatore, Partita	Minuti, Punti, Assist, Rimbalzi
Direzione	Associa alla partita il suo arbitro	Partita, Arbitro	-
Classifica	Associa una squadra al suo girone	Squadra, Girone	Numero vittorie, Numero sconfitte
Casa	Associa ad una partita la squadra di casa	Squadra, Partita	Punteggio casa
Trasferta	Associa ad una partita la squadra in trasferta	Squadra, Partita	Punteggio trasferta
Locazione	Associa ad una partita il palazzetto dove viene giocata	Partita, Palazzetto	-

Ingresso	Associa un biglietto alla relativa partita	Partita, Biglietto	-
Acquisto	Associa ad un biglietto i dati dell'acquirente	Biglietto, Spettatore	-

## Considerazioni generali (alcune motivazioni)

- Ho deciso di accorpare (almeno inizialmente) Giocatore e Arbitro sotto un'unica generalizzazione perché condividono molti attributi, procederò in un secondo momento all'eliminazione della generalizzazione.
- L'opzionalità in Direzione deriva dal fatto che all'inizio del torneo viene inserita una lista di arbitri che sono disponibili ad arbitrare, non è detto che tutti debbano arbitrare almeno una partita.
- La stessa cosa vale per i palazzetti, non è detto che in tutti i palazzetti venga giocata almeno una partita.
- L'opzionalità di Statistica invece deriva dal fatto che non tutti i giocatori giocano almeno una partita: se un giocatore non si presenta o gioca 0 minuti, la sua statistica per quella partita non viene inserita.
- Avrei potuto eliminare l'entità Biglietto e usarla come relazione che collega le entità Partita e Spettatore, valuterò in seguito quale strategia è migliore.
- La non opzionalità di Acquisto dipende dal fatto che uno Spettatore per essere tale deve aver acquistato almeno un biglietto.
- Ogni partita prevede che si affrontino 2 squadre. Ho valutato di utilizzare due relazioni distinte Casa e Ospite per chiarezza. Inoltre, punteggio casa e punteggio ospite li ho messi come attributi delle relazioni e non di Partita in quanto sono concettualmente "collegati" anche a Squadra.
- La lista delle partite viene inserita del tutto all'inizio del torneo, specificando per ciascuna partita le due squadre che si affrontano, la data e il palazzetto. I risultati delle partite e i relativi incassi verranno invece aggiunti man mano che si gioca il torneo, andando a modificare le relative istanze di Partita.

## Vincoli non esprimibili graficamente

- Il numero di biglietti venduti per una partita non può essere maggiore della capienza del relativo palazzetto.
- In un singolo palazzetto non possono essere giocate più di 3 partite al giorno.
- Un arbitro non può arbitrare più di 2 partite nello stesso giorno.
- I giocatori devono essere tutti maggiorenne e con un certificato medico valido (fatto al massimo un anno prima).
- Nella stessa squadra non possono esserci giocatori con lo stesso numero di maglia
- Non possono affrontarsi squadre di gironi diversi.
- La somma dei punti messi a referto dai singoli giocatori in una determinata partita deve essere uguale al punteggio fatto dalla propria squadra in quella partita (non si possono verificare "auto canestri").

## Tavola dei volumi

Il torneo ha un numero predefinito di squadre e di conseguenza di partite.

Partecipano 24 squadre suddivise in 4 gironi da 6.

Ogni squadra ha almeno 6 e massimo 10 giocatori, supponendo in media 8 giocatori per squadra, il numero totale di giocatori registrati è  $24 \times 8 = 192$

Ogni squadra gioca 5 partite, quindi in totale ci sono  $(24 \times 5) / 2 = 60$  partite

Per ogni partita suppongo che in media mettano piede in campo 7 giocatori per squadra, perciò ci sono  $60 \times 7 \times 2 = 840$  "prestazioni".

I palazzetti a disposizione sono 6.

Gli arbitri disponibili ad arbitrare gli incontri sono circa 20.

Per ogni partita si stima una media di 100 spettatori, quindi i biglietti totali venduti saranno  $60 \times 100 = 6000$  circa.

Di questi 6000 biglietti, si stima che in media un singolo spettatore assiste a 3 partite nel corso del torneo, perciò gli spettatori unici sono 2000.

Concetto	Tipo	Volume
Giocatore	E	$24 \times 8 = 192$
Squadra	E	24
Arbitro	E	20

Girone	E	4
Partita	E	$24*5/2=60$
Biglietto	E	$60*100 = 6000$
Spettatore	E	$6000/3 = 2000$
Palazzetto	E	6
Statistica	R	$60*7*2 = 840$
Appartenenza	R	192
Composizione	R	24
Casa	R	60
Ospite	R	60
Direzione	R	60
Locazione	R	60
Ingresso	R	6000
Acquisto	R	6000

## Operazioni d'interesse

- Descrizione:** Visualizzazione della classifica provvisoria di ciascuno dei 4 gironi, con le squadre ordinate per percentuale di vittorie in modo decrescente.  
**Tipo:** Interattivo  
**Frequenza:** 50/giorno
- Descrizione:** Per ogni partita già giocata, visualizzazione dei dettagli (squadre che si affrontano, risultato finale, top marcatore di ogni squadra, data, luogo, arbitro, numero biglietti venduti).  
**Tipo:** Interattivo  
**Frequenza:** 100/giorno
- Descrizione:** Per una determinata squadra scelta dall'utente, per ciascun giocatore si visualizzano nome, cognome, data di nascita e le statistiche individuali (media minuti, media punti, media rimbalzi, media assist, numero partite giocate).  
**Tipo:** Interattivo  
**Frequenza:** 100/giorno

4. **Descrizione:** Visualizzazione del calendario delle partite di un determinato palazzetto.  
**Tipo:** Interattivo  
**Frequenza:** 100/giorno
  
5. **Descrizione:** Si vuole calcolare l'incasso totale di ciascun palazzetto.  
**Tipo:** Batch  
**Frequenza:** 1 a fine torneo
  
6. **Descrizione:** L'ente vuole determinare i 5 spettatori più "affezionati" (che hanno assistito a più partite e a parità di partite viste che hanno speso più soldi) per inviare loro email promozionali e gadget.  
**Tipo:** Batch  
**Frequenza:** 1 a fine torneo
  
7. **Descrizione:** L'ente vuole visualizzare le date dei brevetti degli arbitri che hanno arbitrato partite con più di un determinato numero di spettatori.  
**Tipo:** Batch  
**Frequenza:** 3/giorno
  
8. **Descrizione:** L'ente vuole determinare, per ogni palazzetto, quante prestazioni da "tripla doppia" ci sono state nelle partite giocate in quel palazzetto (tripla doppia = un giocatore che mette a referto almeno 10 punti, 10 assist e 10 rimbalzi nella stessa partita).  
**Tipo:** Batch  
**Frequenza:** 1 a fine torneo
  
9. **Descrizione:** L'ente vuole determinare il giocatore più performante del torneo, ossia il giocatore col maggior numero di doppie doppie ( $\geq 10$  in due delle tre statistiche). A parità di questo fattore prevale il giocatore con la minor media minuti.  
**Tipo:** Batch  
**Frequenza:** 1 a fine torneo

### **Altre operazioni più "standard"**

10. **Descrizione:** Inserimento dei dati di giocatori, squadre e partite  
**Tipo:** Batch  
**Frequenza:** 1 prima dell'inizio del torneo
  
11. **Descrizione:** Inserimento dei dati di una partita: modifica del risultato, inserimento statistiche individuali dei giocatori, calcolo dell'incasso.  
**Tipo:** Batch  
**Frequenza:** 12/giorno
  
12. **Descrizione:** Vendita di un biglietto a uno spettatore con conseguente incremento del numero di spettatori per quella partita.  
**Tipo:** Interattiva  
**Frequenza:** 1200/giorno

## Eliminazione generalizzazioni

L'unica generalizzazione presente è quella di Persona (padre) a cui fanno riferimento Giocatore e Arbitro (figli). Questa decisione era stata presa inizialmente in quanto queste due entità condividono molti attributi tra di loro.

Tuttavia, esse differiscono per due attributi ciascuno e, inoltre, non siamo interessati ad avere giocatori e arbitri nella stessa entità. Infatti, come si può notare dalle operazioni, queste due entità vengono sempre trattate separatamente e non c'è alcun bisogno di tenerle insieme.

Di conseguenza, in questo caso la scelta più sensata è senz'altro accorpare la generalizzazione nelle entità figlie.

## Analisi ridondanze

### **CALCOLO DELL'INCASSO PER PARTITA:**

In questo caso abbiamo due opzioni: includere o meno nell'entità Partita l'attributo "Incasso", che rappresenta la somma dei prezzi dei biglietti venduti per quella determinata partita. Dal momento che si può anche pagare in loco, l'organizzazione decide di calcolare l'incasso dopo la partita (per questo motivo, non posso semplicemente sommare i prezzi dei biglietti man mano che li vendo).

Quindi, nel caso si abbia questo attributo "Incasso", gli organizzatori del torneo devono provvedere ad aggiornare il database inserendo l'incasso relativo ad ogni partita.

L'aggiunta di questo attributo ovviamente facilita notevolmente l'operazione n.5, in quanto invece di sommare i prezzi dei singoli biglietti per ogni partita posso semplicemente tenere traccia del totale.

Tuttavia, in questo modo viene rallentata l'operazione di inserimento (operazione n.11) dei dati relativi ad ogni partita, in quanto per inserire il totale nell'attributo "incasso" devo comunque aver fatto la somma dei prezzi dei biglietti venduti.

Concetto	Costrutto	Accessi	Tipo
Operazione n.5			
Partita	E	60	L
Operazione n.11			
Biglietto	E	100	L
Partita	E	1	L
Partita	E	1	S

**In presenza di ridondanza,** l'operazione n.5 viene semplificata perché devo semplicemente accedere all'attributo Incasso, per ciascuna delle 60 partite. Invece l'operazione n.11 diventa più dispendiosa perché per scrivere l'incasso totale devo sommare i prezzi dei biglietti (avevo stimato 100 per partita) per 12 volte al giorno, andando poi a modificare l'istanza nell'entità Partita.

Il numero di accessi giornalieri è invece dato da  $(100+1+1)*12 = 1224$  accessi.

Quindi, essendoci 5 giorni di partite, il numero totale di accessi nel corso del torneo è  $1224*5+60 = 6180$  accessi (l'operazione 5 viene effettuata una sola volta a fine torneo).

Concetto	Costrutto	Accessi	Tipo
Operazione n.5			
Biglietto	E	6000	L

volta alla fine del torneo).

Di conseguenza, senza la ridondanza non ho accessi giornalieri ma soltanto **6000** accessi alla fine del torneo, perciò concludo che in questo caso va **eliminata la ridondanza**.

#### **CALCOLO DELLA CLASSIFICA DEI GIRONI:**

Come nel caso precedente, anche qui si tratta di esaminare se è conveniente o meno aggiungere un attributo per semplificare le operazioni di lettura. In questo caso, si tratta di tenere o meno gli attributi "Numero vittorie" e "Numero sconfitte" alla relazione Composizione che collega le entità Squadra e Girone.

Le operazioni interessate sono la n.11, poiché quando si inserisce il risultato di una partita si va a modificare anche tali attributi (se in presenza di ridondanza) e la n.1, che consente agli utenti/organizzatori di visualizzare le classifiche dei vari gironi.

Concetto	Costrutto	Accessi	Tipo
Operazione 1			
Composizione	R	24	L
Operazione 11			
Partita	E	1	L
Partita	E	1	S
Composizione	R	2	L
Composizione	R	2	S

**In presenza di ridondanza,**  
l'operazione n.1 viene notevolmente semplificata (devo solo accedere agli attributi del numero di vittorie e sconfitte per ciascuna delle 24 squadre). Invece, l'operazione 11 diventa più dispendiosa perché devo aggiornare tali attributi (2 accessi perché per entrambe le squadre coinvolte), dopo aver inserito il risultato della partita.

Dunque, il numero totale di accessi giornalieri è dato da:  $24*50+12*2+12*4 = \mathbf{1272\ accessi}$

Concetto	Costrutto	Accessi	Tipo

Operazione n.11			
Partita	E	1	L
Partita	E	1	S
Operazione n.1			
Partita	E	60	L

In **assenza di ridondanza**, risparmio le scritture sull'attributo Somma punti, ma diventa più dispendiosa l'operazione n.1, in quanto per calcolare i punti di ogni squadra devo leggere tutte le istanze dell'entità Partita (ci sono 60 partite in totale).

Il numero totale è quindi dato da  $2*12 + 60*50 = \mathbf{3024}$  accessi giornalieri

Di conseguenza, in questo caso è più conveniente **mantenere la ridondanza**.

#### **CALCOLO DELLE STATISTICHE INDIVIDUALI DEI GIOCATORI:**

In questo caso si ragiona sull'operazione n.3, che prevede di visualizzare le medie statistiche dei giocatori di una determinata squadra (minuti, punti, rimbalzi, assist). Ci sono altre operazioni che riguardano statistiche individuali, ma nelle singole partite, non le medie. In particolare, dobbiamo analizzare se sia conveniente o meno aggiungere un attributo composto "Medie" all'entità Giocatore, che tiene conto direttamente delle medie delle statistiche realizzate.

Ovviamente, l'aggiunta di questo attributo renderebbe più efficiente la lettura per l'operazione n.3 in sé, ma rallenterebbe il processo di inserimento delle statistiche individuali (operazione n.11).

Concetto	Costrutto	Accessi	Tipo
Operazione n.3			
Giocatore	E	8	L
Operazione n.11			
Statistica	R	14	S
Giocatore	E	14	L
Giocatore	E	14	S

In **presenza di ridondanza**, l'operazione n.3 diventa una semplice lettura (ci sono 8 giocatori in media per squadra). Invece, l'operazione n.11 si complica perché devo: inserire la statistica (avevo stimato 7 giocatori per squadra a referto), leggere le medie precedenti, sommarvi le nuove statistiche e ricalcolare le nuove medie ottenute (sempre 14, vanno modificate le statistiche solo dei giocatori che hanno effettivamente giocato quella partita).

Di conseguenza, in presenza di ridondanza il numero totale è dato da  $8*100+(14+14+14)*12 = \mathbf{1304}$  accessi giornalieri.

Concetto	Costrutto	Accessi	Tipo
Operazione n.11			
Statistica	R	14	S
Operazione n.3			
Statistica	R	18	L

In **assenza di ridondanza**, si risparmia sul costo dell'operazione n.11 perché non bisogna aggiornare nessun dato nell'entità Giocatore. Tuttavia, aumenta il costo dell'operazione n.3 perché per calcolare le medie di ciascun giocatore della squadra X devo leggere tutte le istanze di Statistica relative a giocatori della squadra X ( $7*5 = 35$  a fine torneo,

tengo 18 come media perché gli accessi vengono fatti in modo uniforme nelle varie giornate).

Quindi, il numero totale in assenza di ridondanza è dato da  $14*12 + 18*100 = 1968$  accessi giornalieri.

Per cui, è più conveniente **avere una ridondanza**, aggiungendo all'entità Giocatore l'attributo composto "Medie" e l'attributo "Numero partite", che mi consente di calcolare ed aggiornare le medie.

### **CALCOLO DEL NUMERO DI SPETTATORI PER PARTITA**

In questo caso, voglio determinare se sia conveniente o meno avere un attributo "Num Spettatori" nell'entità Partita.

Avere tale attributo rallenta l'operazione di vendita biglietti, in quanto dopo ogni acquisto (in media 1200 al giorno) devo andare a modificare il numero di biglietti venduti nell'entità Partita (sia lettura che scrittura, quindi 2400 accessi giornalieri). Dall'altra parte, tuttavia, questa scelta facilita notevolmente l'operazione n.7, in quanto devo semplicemente leggere tutte le istanze di Partita ( $60 * 3 = 180$  accessi giornalieri).

Invece, non avere questo attributo complicherebbe tale operazione, in quanto per ogni partita dovrei contare il numero di biglietti venduti (6000 istanze in totale = 18000 accessi giornalieri), oltre a rendere più difficile anche il controllo del vincolo di non superamento della capienza del palazzetto.

Di conseguenza, in questo caso risulta evidente che la scelta più conveniente sia quella di mantenere la ridondanza con l'attributo "Num Spettatori".

### **ANALISI DEL CICLO GIOCATORE - PARTITA - SQUADRA**

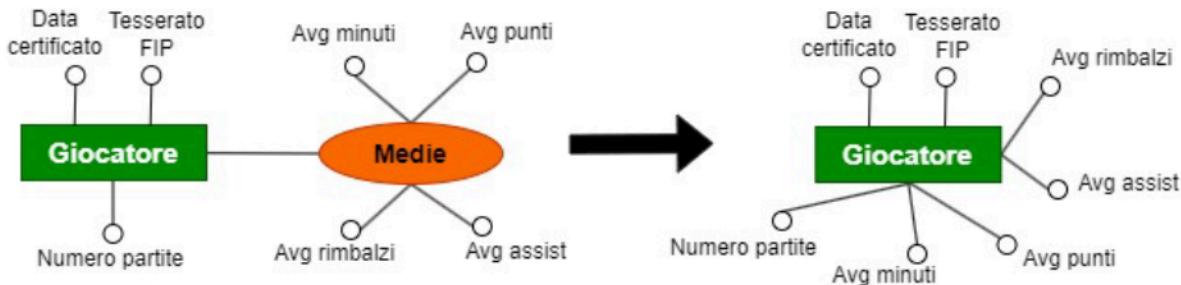
Si può notare che si forma un ciclo tra le entità Giocatore - Squadra - Partita e le relazioni Appartenenza - Casa/Ospite - Statistica. Si potrebbe pensare di eliminare una delle relazioni in modo da "spezzare" questo ciclo.

Tuttavia, tutte queste relazioni sono indispensabili in quante vengono tutte coinvolte dalle operazioni, perciò in questo caso decidiamo a prescindere di mantenere tutte le relazioni. Infatti, la relazione "Appartenenza" è indispensabile per identificare in modo agevole la squadra di ogni giocatore (fondamentale in molte operazioni), così come è necessario collegare le entità Squadra e Partita. Infine, anche la relazione Statistica non può essere eliminata perché è indispensabile per tenere traccia delle prestazioni dei giocatori.

## Partizionamento attributi composti

Come abbiamo visto analizzando le ridondanze, dobbiamo aggiungere un attributo composto "Medie" che tenga traccia delle prestazioni medie dei giocatori nelle varie partite.

Decidiamo di partizionare l'attributo composto in più attributi singoli in quanto in alcune operazioni siamo interessati ad ordinare i risultati dell'operazione per uno dei 4 attributi (es. nell'operazione n.9 ordino anche per media minuti):



Al contrario, le entità Giocatore, Arbitro e Palazzetto hanno gli attributi Residenza / Luogo, che sono entrambi composti da CAP e Indirizzo. In questo caso, non essendo interessati ai singoli attributi, racchiudiamo tutto in un solo attributo.

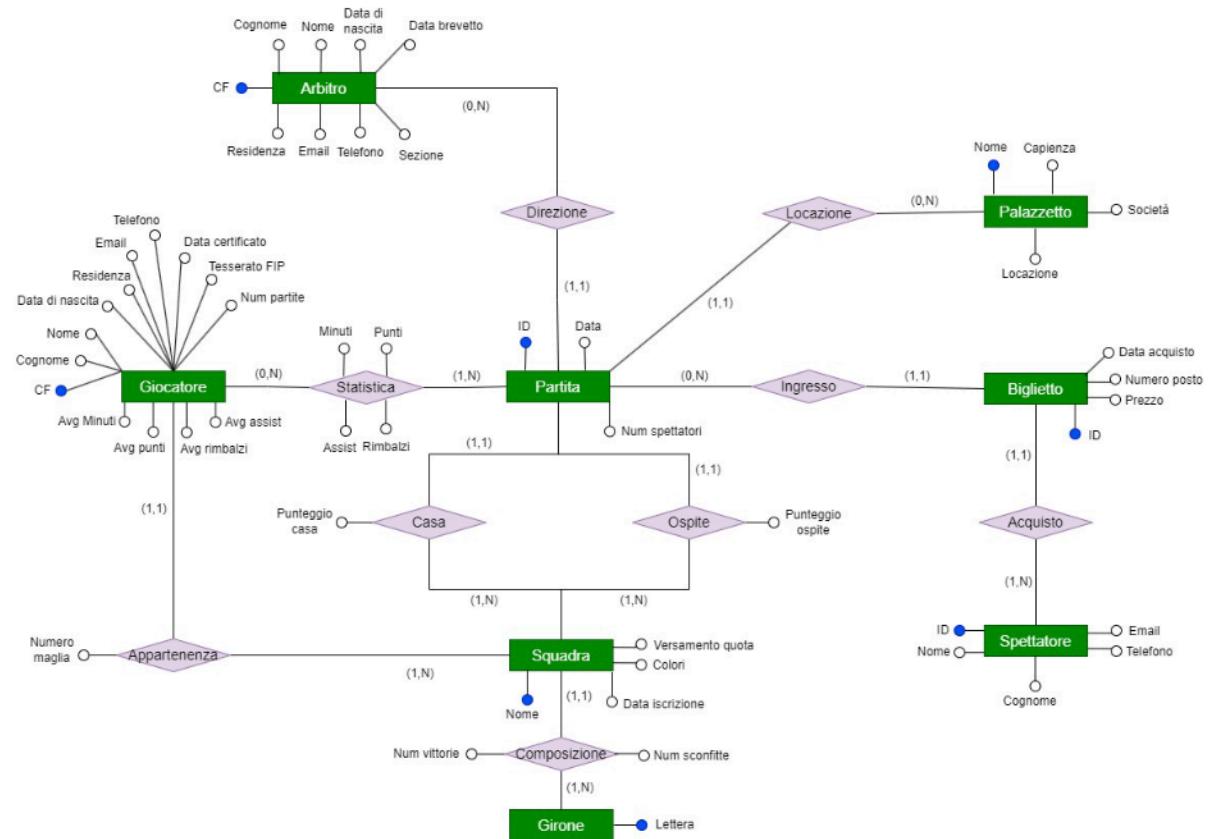
## Accorpamento/partizionamento di E/R

Non è necessario effettuare alcun accorpamento/partizionamento di entità o relazioni

## Scelta degli identificatori

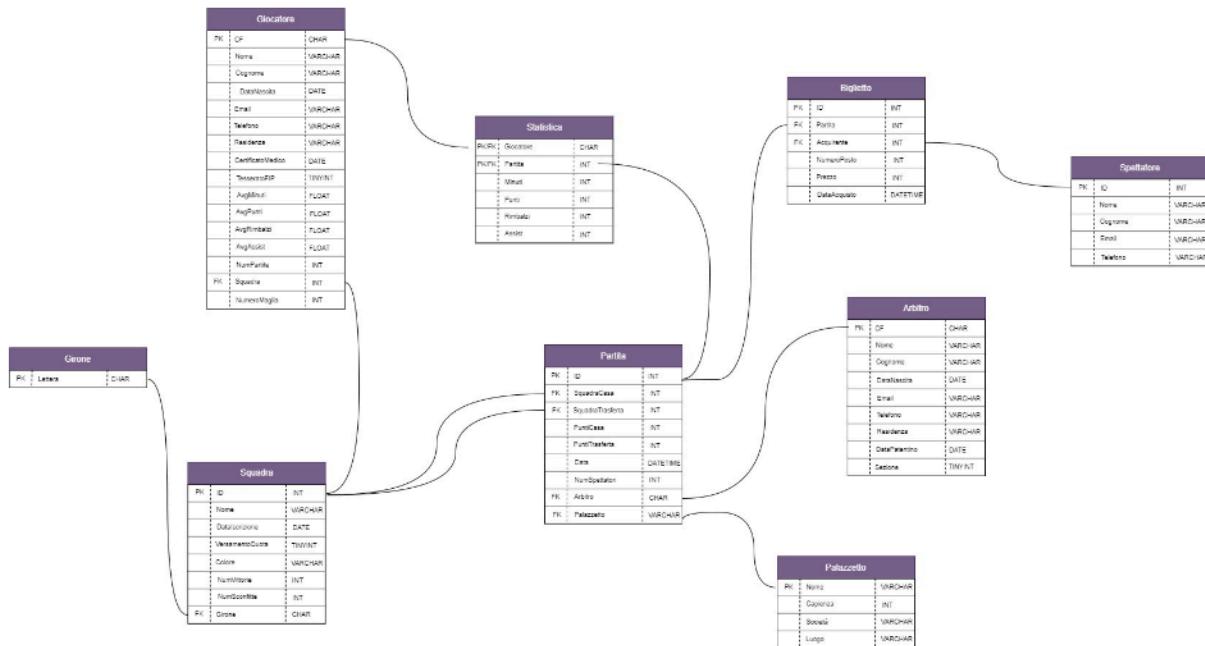
- Giocatore: CF
- Partita: ID
- Arbitro: CF
- Squadra: ID
- Statistica: (Giocatore, Partita)
- Girone: Lettera
- Biglietto: ID
- Spettatore: ID
- Palazzetto: Nome

## Schema ER ristrutturato



## Passaggio allo schema logico

- **Giocatore**: (CF, Nome, Cognome, DataNascita, Email, Telefono, Residenza, CertificatoMedico, TesseratoFIP, AvgMinuti, AvgPunti, AvgRimbalzi, AvgAssist, NumPartite, Squadra, NumeroMaglia).
- **Partita**: (ID, SquadraCasa, SquadraTrasferta, PuntiCasa, PuntiTrasferta, Data, NumSpettatori, Arbitro, Palazzetto)
- **Arbitro**: (CF, Nome, Cognome, DataNascita, Email, Telefono, Residenza, DataPatentino, Sezione).
- **Statistica**: (Giocatore, Partita, Minuti, Punti, Rimbalzi, Assist)
- **Squadra**: (Nome, DataIsrizione, VersamentoQuota, Colore, NumVittorie, NumSconfitte, Girone)
- **Girone**: (Lettera)
- **Biglietto**: (ID, Partita, Acquirente, NumeroPosto, Prezzo, DataAcquisto)
- **Spettatore**: (ID, Nome, Cognome, Email, Telefono)
- **Palazzetto**: (Nome, Capienza, Società, Luogo)



## Normalizzazione

La 1NF è rispettata poiché tutti gli attributi nelle varie tabelle sono atomici.

Per quanto riguarda la 2NF, l'unica tabella che presenta una chiave primaria composta è Statistica. Tuttavia, si può verificare che tutti gli attributi dipendono dalla chiave composta, e dunque la 2NF è rispettata.

Infine, la 3NF non viene rispettata perché sono presenti vari campi calcolati che si possono ricavare da attributi di altre tabelle. Tuttavia, come abbiamo visto analizzando le ridondanze, questa scelta è conveniente poiché permette di diminuire il numero di accessi e quindi velocizzare le varie operazioni.

## SQL

### TRIGGER (di controllo)

- Trigger che blocca l'inserimento di un'istanza nella tabella Giocatore. Questo può avvenire in presenza di 4 condizioni:
  - il giocatore è minorenne
  - il certificato medico del giocatore è scaduto
  - ci sono due giocatori con lo stesso numero di maglia nella stessa squadra
  - ci sono già 10 giocatori in quella squadra

```

DELIMITER $$

CREATE TRIGGER controlloGiocatore
BEFORE INSERT ON Giocatore
FOR EACH ROW
BEGIN

    DECLARE num_giocatori INT;
    DECLARE num_pl_maglia INT;

    IF NEW.DataNascita > CURRENT_DATE - INTERVAL 18 YEAR THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Il giocatore è
minorenne. Inserimento non consentito.';
    END IF;

    IF NEW.CertificatoMedico < CURRENT_DATE - INTERVAL 1 YEAR THEN
        SIGNAL SQLSTATE '45001' SET MESSAGE_TEXT = 'Il certificato
medico è scaduto. Inserimento non consentito.';
    END IF;

    SELECT COUNT(*) INTO num_pl_maglia FROM Giocatore WHERE Squadra =
NEW.Squadra AND NumeroMaglia = NEW.NumeroMaglia;
    IF num_pl_maglia > 0 THEN
        SIGNAL SQLSTATE '45002' SET MESSAGE_TEXT = 'Esiste già un altro
giocatore della stessa squadra con lo stesso numero di maglia.
Inserimento non consentito.';
    END IF;

    SELECT COUNT(*) INTO num_giocatori FROM Giocatore WHERE Squadra =
NEW.Squadra;
    IF num_giocatori > 9 THEN
        SIGNAL SQLSTATE '45003' SET MESSAGE_TEXT = 'Ci sono già 10
giocatori in questa squadra. Inserimento non consentito.';
    END IF;
END$$
DELIMITER ;

```

2. Trigger che blocca l'inserimento di un'istanza nella tabella Partita. Questo può avvenire in presenza di 3 condizioni:
- le due squadre sfidanti appartengono a gironi diversi
  - ci sono più di 3 partite nello stesso giorno e nello stesso palazzetto
  - un arbitro dirige più di 2 partite nello stesso giorno

```

DELIMITER $$

CREATE TRIGGER controlloPartita

```

```

BEFORE INSERT ON Partita
FOR EACH ROW
BEGIN
    DECLARE num_partite INT;
    DECLARE num_arbitraggi INT;

    DECLARE GironeCasa CHAR(1);
    DECLARE GironeTrasferta CHAR(1);
    SELECT Girone INTO GironeCasa FROM Squadra WHERE ID =
NEW.SquadraCasa;
    SELECT Girone INTO GironeTrasferta FROM Squadra WHERE ID =
NEW.SquadraTrasferta;
    IF GironeCasa <> GironeTrasferta THEN
        SIGNAL SQLSTATE '45004' SET MESSAGE_TEXT = 'Le due squadre
appartengono a gironi diversi. Inserimento non consentito.';
    END IF;

    SELECT COUNT(*) INTO num_partite FROM Partita WHERE
DATE(DataPartita) = DATE(NEW.DataPartita) AND Palazzetto =
NEW.Palazzetto;
    IF num_partite >= 3 THEN
        SIGNAL SQLSTATE '45005' SET MESSAGE_TEXT = 'Ci sono già 3
partite nello stesso giorno e nello stesso palazzetto. Inserimento non
consentito.';
    END IF;

    SELECT COUNT(*) INTO num_arbitraggi FROM Partita WHERE
DATE(DataPartita) = DATE(NEW.DataPartita) AND Arbitro = NEW.Arbitro;
    IF num_arbitraggi >= 2 THEN
        SIGNAL SQLSTATE '45006' SET MESSAGE_TEXT = 'L\'arbitro ha già
arbitrato 2 partite nello stesso giorno. Inserimento non consentito.';
    END IF;

END$$
DELIMITER ;

```

3. Trigger che blocca la modifica di una partita (inserimento risultati) se la somma dei punti dei giocatori di una delle due squadre è diversa dai punti che voglio inserire per quella squadra:

```

DELIMITER $$
CREATE TRIGGER controlloStatistica
BEFORE UPDATE ON Partita

```

```

FOR EACH ROW
BEGIN

    DECLARE puntiGiocatoriCasa INT;
    DECLARE puntiGiocatoriTrasferta INT;

    SELECT SUM(Punti) INTO puntiGiocatoriCasa
    FROM Statistica
    WHERE Partita = NEW.ID AND Giocatore IN (SELECT CF FROM Giocatore
    WHERE Squadra = NEW.SquadraCasa);

    SELECT SUM(Punti) INTO puntiGiocatoriTrasferta
    FROM Statistica
    WHERE Partita = NEW.ID AND Giocatore IN (SELECT CF FROM Giocatore
    WHERE Squadra = NEW.SquadraTrasferta);

    IF puntiGiocatoriCasa <> NEW.PuntiCasa THEN
        SIGNAL SQLSTATE '45007' SET MESSAGE_TEXT = 'La somma dei punti
        dei giocatori della squadra di casa non corrisponde ai punti totali
        della squadra.';
    END IF;

    IF puntiGiocatoriTrasferta <> NEW.PuntiTrasferta THEN
        SIGNAL SQLSTATE '45008' SET MESSAGE_TEXT = 'La somma dei punti
        dei giocatori della squadra in trasferta non corrisponde ai punti totali
        della squadra.';
    END IF;

END$$
DELIMITER ;

```

4. Trigger che blocca l'inserimento di un biglietto se questo eccede la capienza massima del palazzetto:

```

DELIMITER $$

CREATE TRIGGER controlloCapienzaPalazzetto
BEFORE INSERT ON Biglietto
FOR EACH ROW
BEGIN

    DECLARE numBiglietti INT;
    DECLARE capienzaPalazzetto INT;

    SELECT COUNT(*) INTO numBiglietti FROM Biglietto WHERE Partita =

```

```

NEW.Partita;

    SELECT Capienza INTO capienzaPalazzetto FROM Palazzetto WHERE Nome =
(SELECT Palazzetto FROM Partita WHERE ID = NEW.Partita);

    IF numBiglietti >= capienzaPalazzetto THEN
        SIGNAL SQLSTATE '45009'
        SET MESSAGE_TEXT = 'Il numero di biglietti venduti per questa
partita supera la capienza del palazzetto.';
    END IF;

END$$

DELIMITER ;

```

## TRIGGER (di modifica)

- Trigger che, quando inserisco un biglietto, modifica l'attributo NumSpettatori per la relativa partita:

```

DELIMITER $$

CREATE TRIGGER modificaNumSpettatori
BEFORE INSERT ON Biglietto
FOR EACH ROW
BEGIN
    UPDATE Partita SET NumSpettatori = NumSpettatori + 1 WHERE ID =
NEW.Partita;

END$$

DELIMITER ;

```

- Trigger che, quando inserisco il risultato di una partita (modificando la tabella Partita), aggiorna anche il numero di vittorie e sconfitte delle due squadre in base a chi ha vinto:

```

DELIMITER $$

CREATE TRIGGER modificaNumVittorie
AFTER UPDATE ON Partita
FOR EACH ROW

```

```

BEGIN
    IF NEW.PuntiCasa > NEW.PuntiTrasferta THEN
        UPDATE Squadra SET NumVittorie = NumVittorie + 1 WHERE ID =
NEW.SquadraCasa;
        UPDATE Squadra SET NumSconfitte = NumSconfitte + 1 WHERE ID =
NEW.SquadraTrasferta;
    ELSE
        UPDATE Squadra SET NumVittorie = NumVittorie + 1 WHERE ID =
NEW.SquadraTrasferta;
        UPDATE Squadra SET NumSconfitte = NumSconfitte + 1 WHERE ID =
NEW.SquadraCasa;
    END IF;
END$$

DELIMITER ;

```

3. Trigger che, quando inserisco le statistiche dei giocatori per una determinata partita, modifica le medie delle prestazioni di quei giocatori:

```

DELIMITER $$

CREATE TRIGGER modificaMedieGiocatore
AFTER INSERT ON Statistica
FOR EACH ROW
BEGIN
    DECLARE vecchiMinuti FLOAT;
    DECLARE vecchiPunti FLOAT;
    DECLARE vecchiRimbalzi FLOAT;
    DECLARE vecchiAssist FLOAT;
    DECLARE numeroPartite INT;

    DECLARE nuoviMinuti FLOAT;
    DECLARE nuoviPunti FLOAT;
    DECLARE nuoviRimbalzi FLOAT;
    DECLARE nuoviAssist FLOAT;

    SELECT AvgMinuti INTO vecchiMinuti FROM Giocatore WHERE CF =
NEW.Giocatore;
    SELECT AvgPunti INTO vecchiPunti FROM Giocatore WHERE CF =
NEW.Giocatore;
    SELECT AvgRimbalzi INTO vecchiRimbalzi FROM Giocatore WHERE CF =
NEW.Giocatore;
    SELECT AvgAssist INTO vecchiAssist FROM Giocatore WHERE CF =
NEW.Giocatore;

```

```

    SELECT NumPartite INTO numeroPartite FROM Giocatore WHERE CF =
NEW.Giocatore;

    SET nuoviMinuti = ((vecchiMinuti * numeroPartite) + NEW.Minuti) /
(numeroPartite + 1);
    SET nuoviPunti = ((vecchiPunti * numeroPartite) + NEW.Punti) /
(numeroPartite + 1);
    SET nuoviRimbalzi = ((vecchiRimbalzi * numeroPartite) +
NEW.Rimbalzi) / (numeroPartite + 1);
    SET nuoviAssist = ((vecchiAssist * numeroPartite) + NEW.Assist) /
(numeroPartite + 1);

    UPDATE Giocatore
    SET AvgMinuti = nuoviMinuti,
        AvgPunti = nuoviPunti,
        AvgRimbalzi = nuoviRimbalzi,
        AvgAssist = nuoviAssist,
        NumPartite = (numeroPartite + 1)
    WHERE CF = NEW.Giocatore;

END$$

DELIMITER ;

```

## OPERAZIONI

- Visualizzazione della classifica provvisoria di ciascuno dei 4 gironi, con le squadre ordinate per percentuale di vittorie in modo decrescente.

```

DELIMITER $$

CREATE PROCEDURE VisualizzaClassificaGironi()
BEGIN
    SELECT
        S.Girone,
        S.Nome,
        S.NumVittorie,
        S.NumSconfitte,
        (S.NumVittorie / (S.NumVittorie + S.NumSconfitte)) AS
PercentualeVittorie,
        S.NumVittorie * 2 AS Punti
    FROM Squadra S
    ORDER BY S.Girone, PercentualeVittorie DESC;
END$$

```

```
DELIMITER ;
```

2. Per ogni partita, visualizzazione dei dettagli (squadre che si affrontano, risultato finale, top marcatore della partita, data, luogo, arbitro, numero spettatori paganti).

```
DELIMITER $$
```

```
CREATE PROCEDURE DettagliPartita()
BEGIN
    SELECT
        P.ID,
        SC.Nome AS SquadraCasa,
        ST.Nome AS SquadraTrasferta,
        P.PuntiCasa,
        P.PuntiTrasferta,
        P.DataPartita,
        Pal.Nome AS Palazzetto,
        CONCAT(Arb.Nome, ' ', Arb.Cognome) AS Arbitro,
        P.NumSpettatori,
        (
            SELECT CONCAT(G.Nome, ' ', G.Cognome)
            FROM Statistica S
            JOIN Giocatore G ON S.Giocatore = G.CF
            WHERE S.Partita = P.ID
            ORDER BY S.Punti DESC
            LIMIT 1
        ) AS MigliorMarcatore
    FROM Partita P
    JOIN Squadra SC ON P.SquadraCasa = SC.ID
    JOIN Squadra ST ON P.SquadraTrasferta = ST.ID
    JOIN Palazzetto Pal ON P.Palazzetto = Pal.Nome
    JOIN Arbitro Arb ON P.Arbitro = Arb.CF
    ORDER BY P.DataPartita;
END$$
```

```
DELIMITER ;
```

3. Per una determinata squadra, per ciascun giocatore si visualizzano nome, cognome, data di nascita e le statistiche individuali (media minuti, media punti, media rimbalzi, media assist). Ordino i giocatori per media punti.

```
DELIMITER $$
```

```

CREATE PROCEDURE StatisticheGiocatoriPerSquadra(IN squadra_param
VARCHAR(50))
BEGIN
    SELECT
        S.Nome AS Squadra,
        G.Nome,
        G.Cognome,
        G.DataNascita,
        G.AvgMinuti,
        G.AvgPunti,
        G.AvgRimbalzi,
        G.AvgAssist,
        G.NumPartite
    FROM Giocatore G
    JOIN Squadra S ON G.Squadra = S.ID
    WHERE S.Nome = squadra_param
    ORDER BY G.AvgPunti DESC;
END$$

DELIMITER ;

```

#### 4. Visualizzazione del calendario delle partite di un determinato palazzetto.

```

DELIMITER $$

CREATE PROCEDURE CalendarioPartitePalazzetto(IN palazzettoNome
VARCHAR(100))
BEGIN
    SELECT
        P.ID,
        SC.Nome AS SquadraCasa,
        ST.Nome AS SquadraTrasferta,
        P.DataPartita,
        P.NumSpettatori
    FROM Partita P
    JOIN Squadra SC ON P.SquadraCasa = SC.ID
    JOIN Squadra ST ON P.SquadraTrasferta = ST.ID
    WHERE P.Palazzetto = palazzettoNome
    ORDER BY P.DataPartita;
END$$

DELIMITER ;

```

5. Si vuole calcolare la media di spettatori e l'incasso totale di ciascun palazzetto.

```
DELIMITER $$

CREATE PROCEDURE MediaSpettatoriIncassoPalazzetti()
BEGIN
    SELECT
        P.Nome AS Palazzetto,
        AVG(Pt.NumSpettatori) AS MediaSpettatori,
        SUM(B.Prezzo) AS IncassoTotale
    FROM Palazzetto P
    JOIN Partita Pt ON P.Nome = Pt.Palazzetto
    JOIN Biglietto B ON Pt.ID = B.Partita
    GROUP BY P.Nome;
END$$

DELIMITER ;
```

6. L'ente vuole determinare i 5 spettatori più “affezionati” (che hanno assistito a più partite e a parità di partite viste che hanno speso più soldi) per inviare loro email promozionali e gadget.

```
DELIMITER $$

CREATE PROCEDURE SpettatoriAffezionati()
BEGIN
    SELECT
        S.Nome,
        S.Cognome,
        S.Email,
        COUNT(B.ID) AS NumPartite,
        SUM(B.Prezzo) AS TotaleSpeso
    FROM Spettatore S
    JOIN Biglietto B ON S.ID = B.Acquirente
    GROUP BY S.ID
    ORDER BY NumPartite DESC, TotaleSpeso DESC
    LIMIT 5;
END$$

DELIMITER ;
```

7. L'ente vuole visualizzare le date dei brevetti degli arbitri che hanno arbitrato partite con più di un determinato numero di spettatori.

```
DELIMITER $$

CREATE PROCEDURE DatePatentiniArbitri(IN minSpettatori INT)
BEGIN
    SELECT
        CONCAT(A.Nome, ' ', A.Cognome) AS Arbitro,
        A.DataPatentino
    FROM Arbitro A
    JOIN Partita P ON A.CF = P.Arbitro
    WHERE P.NumSpettatori >= minSpettatori
    ORDER BY A.DataPatentino;
END$$

DELIMITER ;
```

8. L'ente vuole determinare, per ogni palazzetto, quante prestazioni da "tripla doppia" ci sono state nelle partite giocate in quel palazzetto (tripla doppia = un giocatore che mette a referto almeno 10 punti, 10 assist e 10 rimbalzi nella stessa partita).

```
DELIMITER $$

CREATE PROCEDURE TriplaDoppiaPerPalazzetto()
BEGIN
    SELECT
        P.Nome AS Palazzetto,
        COUNT(*) AS NumTripleDoppi
    FROM Partita Pt
    JOIN Statistica S ON Pt.ID = S.Partita
    JOIN Palazzetto P ON Pt.Palazzetto = P.Nome
    WHERE S.Punti >= 10 AND S.Assist >= 10 AND S.Rimbalzi >= 10
    GROUP BY P.Nome;
END$$

DELIMITER ;
```

9. L'ente vuole determinare i TOT giocatori più performanti del torneo, ossia i giocatori col maggior numero di doppie doppi (>=10 in due delle tre statistiche). A parità di questo fattore prevale il giocatore con la minor media minuti.

```
DELIMITER $$
```

```
CREATE PROCEDURE GiocatorePerformante(IN num_giocatori INT)
BEGIN
    SELECT
        G.CF,
        G.Nome,
        G.Cognome,
        COUNT(*) AS NumDoppieDoppie,
        G.AvgMinuti
    FROM Giocatore G
    JOIN Statistica S ON G.CF = S.Giocatore
    WHERE (S.Punti >= 10 AND S.Rimbalzi >= 10)
        OR (S.Punti >= 10 AND S.Assist >= 10)
        OR (S.Rimbalzi >= 10 AND S.Assist >= 10)
    GROUP BY G.CF
    ORDER BY NumDoppieDoppie DESC, G.AvgMinuti ASC
    LIMIT num_giocatori;
END$$

DELIMITER ;
```

## DEMO

Ho inserito il codice per la demo con Python nel mio Github:  
<https://github.com/TommasoMoro03/Esame-Basi-Di-Dati/tree/main>  
Nel Readme si può vedere un breve video della demo.