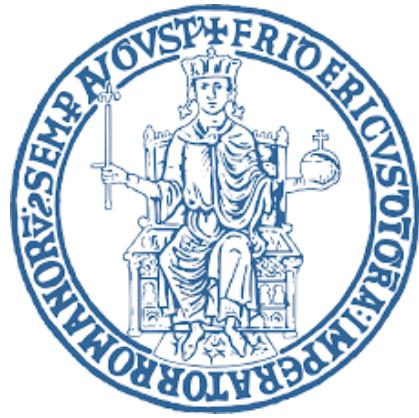


Progetto OO & BD

Milanese Roberto Rocco N86004554

Monteforte Valerio N86005290

Marzo 2025



*Università degli Studi di Napoli Federico II -
Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione
CORSO DI LAUREA IN INFORMATICA*

Indice

Indice	2
1 Database	4
1.1 Analisi dei requisiti	4
1.2 Progettazione Concettuale	7
1.3 Class Diagram non ristrutturato	9
1.4 Analisi della ristrutturazione del diagramma di classe	10
1.4.1 Analisi delle ridondanze	10
1.4.2 Rimozione degli attributi multivalore	10
1.4.3 Rimozione degli attributi composti	10
1.4.4 Partizione/Accorpamento delle associazioni	10
1.4.5 Rimozione delle gerarchie	10
1.4.6 Analisi degli identificativi	11
1.5 Class Diagram ristrutturato	12
1.6 Dizionario Delle Classi	13
1.7 Dizionario Delle Associazioni	17
1.8 Dizionario dei Vincoli	19
2 Traduzione modello logico	20
2.1 Mapping associazioni	20
2.2 Modello logico	22
3 Schema Fisico	24
3.1 Creazione del database	24
3.2 Creazione entità	24
3.3 Trigger Function	31
3.4 Popolamento del Database	50
4 Object Orientation	59
4.1 Introduzione	59
4.2 Diagramma del Problema	60
4.3 Diagramma della Soluzione	61

4.4 Controller Package	62
4.5 Model Package	63
4.6 DAO Package	64
4.7 Sequence Diagram 1 - AddNewPlayer	65
4.8 Sequence Diagram 2 - Login	66

1 Database

1.1 Analisi dei requisiti

Il sistema informativo richiesto dalla traccia mirava a gestire in modo efficiente le informazioni relative ai calciatori professionisti, includendo le loro interazioni con le squadre di appartenenza e i trofei vinti. In particolare, la traccia poneva come obiettivo la realizzazione di un database dettagliato, capace di rappresentare la carriera del singolo calciatore, nonché i trofei ottenuti sia dalle squadre che dai giocatori stessi.

A seguito di un'attenta analisi e di diversi confronti, è emersa la necessità di introdurre differenti categorie di informazioni al fine di creare un sistema concreto e coerente con i dati reali relativi ai calciatori e al mondo del calcio in generale. Alla base del progetto vi era l'intenzione di sviluppare un sistema informativo in grado di soddisfare tutti gli obiettivi delineati dalla traccia. Per questo motivo, e su suggerimento del corpo docente, abbiamo tratto ispirazione da sistemi consolidati quali **Transfermarkt** e **OneFootball**, che si distinguono per la dettagliata descrizione degli eventi e delle informazioni calcistiche.

Uno degli aspetti principali affrontati durante la progettazione del database è stata la gestione delle competizioni e delle relative tipologie. Considerata l'enorme varietà di competizioni esistenti nel panorama calcistico, abbiamo deciso di introdurre un **dominio dedicato** per l'attributo responsabile di identificare la tipologia della competizione salvata nel database. Questo dominio ha il compito di racchiudere tutte le competizioni ammesse dal sistema, fornendo così una struttura organizzata e facilmente interrogabile. Un altro attributo di rilievo è quello della **nazionalità**, che permette di consolidare il concetto di competizione calcistica a livello geografico.

Inoltre, uno degli aspetti di interesse analizzati è stato quello di registrare le competizioni disputate da una determinata squadra, includendo informazioni aggiuntive quali la posizione finale raggiunta. Per rispondere a tale esigenza, è stata introdotta un'entità specifica che tiene conto di questo attributo, consentendo così una gestione accurata dell'assegnazione dei premi in base alla classifica finale ottenuta in una determinata competizione. Per una gestione strutturata e precisa dei premi all'interno del database, abbiamo suddiviso l'entità **Trofeo** in due sottocategorie principali:

- **Trofeo Individuale**

- **Trofeo di Squadra**

Questa separazione ci permette di assegnare i diversi premi in modo distinto, in base al tipo di competizione e al soggetto premiato, garantendo una maggiore flessibilità nella modellazione e gestione delle informazioni.

Trofeo Individuale: rappresenta i premi assegnati ai singoli giocatori. Gli identificativi univoci di questa categoria sono stati contrassegnati con il prefisso IN, per una chiara distinzione.

Trofeo di Squadra: raccoglie i premi destinati alle squadre. Gli identificativi univoci per questi trofei sono stati invece contrassegnati con il prefisso DS.

Questa scelta ci consente di:

1. Organizzare e categorizzare in modo chiaro i dati relativi ai trofei, evitando ambiguità tra le tipologie di premi.
2. Facilitare l’interrogazione e la gestione del database, in quanto il prefisso consente di distinguere rapidamente tra trofei individuali e di squadra.

Per gestire la militanza di un calciatore, abbiamo deciso di creare un’entità chiamata **carriera**, che ci permette di tenere traccia delle informazioni riguardanti un calciatore in una determinata squadra. In particolare, abbiamo deciso di trattare la militanza in un arco di tempo definito da una **data di inizio** e una **data di fine** della carriera in quel club. Sempre per gestire la carriera di un calciatore e il suo eventuale passaggio a ruolo di allenatore o dirigente, abbiamo introdotto un attributo che definisce la fine della carriera da calciatore; attributo che risulta particolarmente importante, come accennato in precedenza, per il passaggio ad altri ruoli all’interno del club, poiché un allenatore o dirigente deve essere stato calciatore per poter essere salvato nel database.

In particolare, abbiamo deciso di integrare nella stessa entità anche le carriere da allenatore o dirigente degli ex calciatori, sia per una questione di praticità, sia per tenere tutte le carriere di una determinata persona raccolte in un’unica entità. Al fine di implementare questa logica, abbiamo creato un attributo chiamato **tipologia**, dove il valore è un elemento di un dominio definito in fase di creazione. Questo dominio ammette solo i valori come “calciatore”, “allenatore” o “dirigente”. Questa scelta ci consente di determinare in modo chiaro la tipologia di carriera, come da noi prefissato.

Le **statistiche dei giocatori**, invece, sono attributi che arricchiscono la carriera del calciatore. Queste informazioni vengono recuperate da un’entità chiamata **compete**, che

conserva le azioni di una data partita per un dato minutaggio. Successivamente, queste azioni vengono elaborate e salvate nell'entità **carriera** per avere le statistiche generali del calciatore aggiornate durante tutta la sua militanza in quel determinato club.

Inoltre, all'interno del database abbiamo dedicato una sezione alle **partite svolte dalle squadre**. In particolare, questa sezione ci aiuta a definire le statistiche generali sia dei singoli calciatori, che del club stesso. Per di più, ci permette di avere un quadro oggettivo per la classificazione e l'assegnazione dei premi e trofei. Per gestire gli scontri tra due club, abbiamo preferito un approccio mirato, utilizzando due **tuple** che condividono lo stesso identificativo della partita. Altre informazioni, come il risultato, sono espresse come i goal fatti dalla singola squadra, che diventano i goal subiti dalla squadra avversaria. Questa dualità risulta utile quando dobbiamo assegnare i goal subiti ai portieri che hanno giocato durante la partita.

Un'altra casistica discussa in questa entità riguarda le **partite giocate in casa**, poiché, in alcuni scontri, è necessario che queste partite vengano svolte in determinati stadi, come nel caso del campionato. Per questo motivo, abbiamo deciso di introdurre un attributo booleano dedicato nelle singole tuple delle squadre. Il funzionamento di quest'ultimo verifica che almeno una delle tuple abbia il valore impostato su `true`, ovvero che quella squadra stia giocando in casa. Ovviamente, se questa condizione è vera, è necessario un ulteriore controllo per verificare che lo stadio dove la partita si sta svolgendo appartenga alla squadra corretta.

A tal proposito, abbiamo deciso di creare un'entità che fornisca informazioni sullo stadio e che ci aiuti ad assegnare ad ogni club lo stadio ospitante. Inoltre, questa entità ci permette di definire tutti gli stadi dove si svolgono le partite, anche quando lo stadio ospitante non appartiene a nessuna delle due squadre (come nel caso dei mondiali, delle coppe internazionali, ecc.).

1.2 Progettazione Concettuale

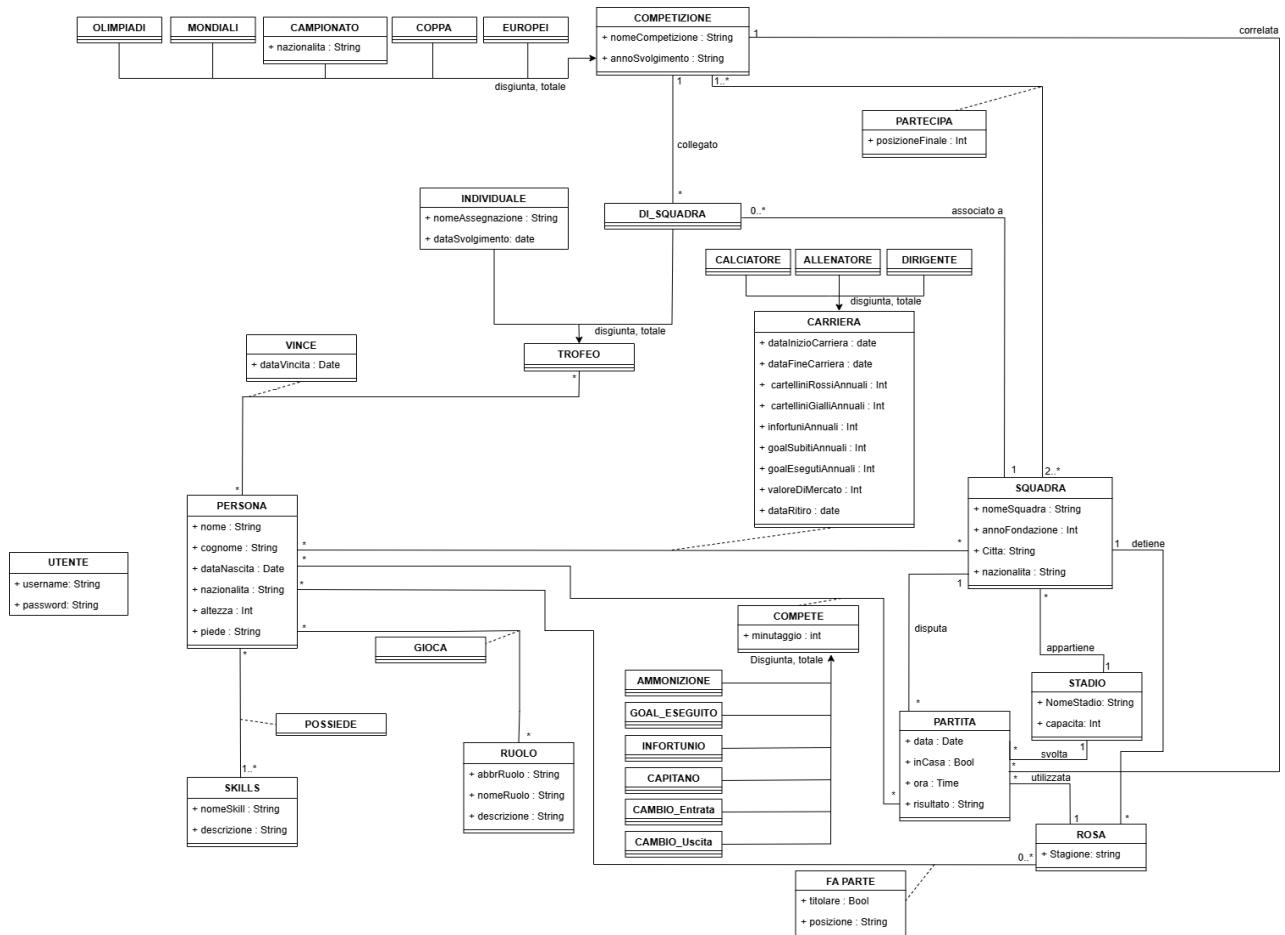
Alla base di quanto descritto dall’analisi dei requisiti, possiamo dedurre una modellazione del sistema nella maniera che segue:

- L’entità **Persona**: rappresenta le generalità di ogni calciatore all’interno del database.
- L’entità **Skills**: rappresenta tutte le abilità calcistiche che possono essere associate ad un calciatore.
- L’entità **Ruolo**: rappresenta le posizioni in cui un giocatore può essere schierato durante una partita.
- L’entità **Trofeo**: rappresenta tutti i trofei che possono essere vinti dalle squadre e dai calciatori.
 - **INDIVIDUALE**
 - **DI SQUADRA**
- L’entità **Competizione**: rappresenta tutte le competizioni a cui le squadre e i relativi calciatori possono partecipare.
 - **OLIMPIADI**
 - **MONDIALI**
 - **CAMPIONATO**
 - **COPPA**
 - **EUROPEI**
- L’entità **Squadra**: rappresenta tutte le caratteristiche associate a tutte le diverse squadre presenti nel database.
- L’entità **Stadio**: rappresenta sia lo stadio ove si disputa una data partita, sia lo stadio che ospita, durante le partite in casa, una determinata squadra.
- L’entità **Partita**: rappresenta tutte le informazioni riguardanti un dato scontro tra due squadre.
- L’entità **Rosa**: raccoglie tutte le rose utilizzate dalle squadre per le partite disputate.

- L'entità **Carriera**: raccoglie tutte le informazioni calcistiche associate ad un calciatore accumulate durante tutti i periodi di militanza.
 - **CALCIATORE**
 - **ALLENATORE**
 - **DIRIGENTE**
- L'entità **Compete**: definisce tutte le informazioni riguardanti i calciatori per la singola partita che si sta svolgendo.

Questa lista di entità va a definire a grandi linee quello che è il modello che abbiamo adottato per trattare questo argomento con il nostro database. Più avanti avremo modo di parlare dei singoli attributi e delle varie scelte adottate per garantire un approccio corretto ed efficiente.

1.3 Class Diagram non ristrutturato



1.4 Analisi della ristrutturazione del diagramma di classe

1.4.1 Analisi delle ridondanze

All'interno del nostro elaborato non sono presenti ridondanze.

1.4.2 Rimozione degli attributi multivalore

All'interno del nostro elaborato non troviamo attributi multivalore.

1.4.3 Rimozione degli attributi composti

All'interno del nostro progetto non sono presenti attributi composti.

1.4.4 Partizione/Accorpamento delle associazioni

Non sono presenti associazioni uno-a-uno.

1.4.5 Rimozione delle gerarchie

Procediamo all'eliminazione delle tre gerarchie.

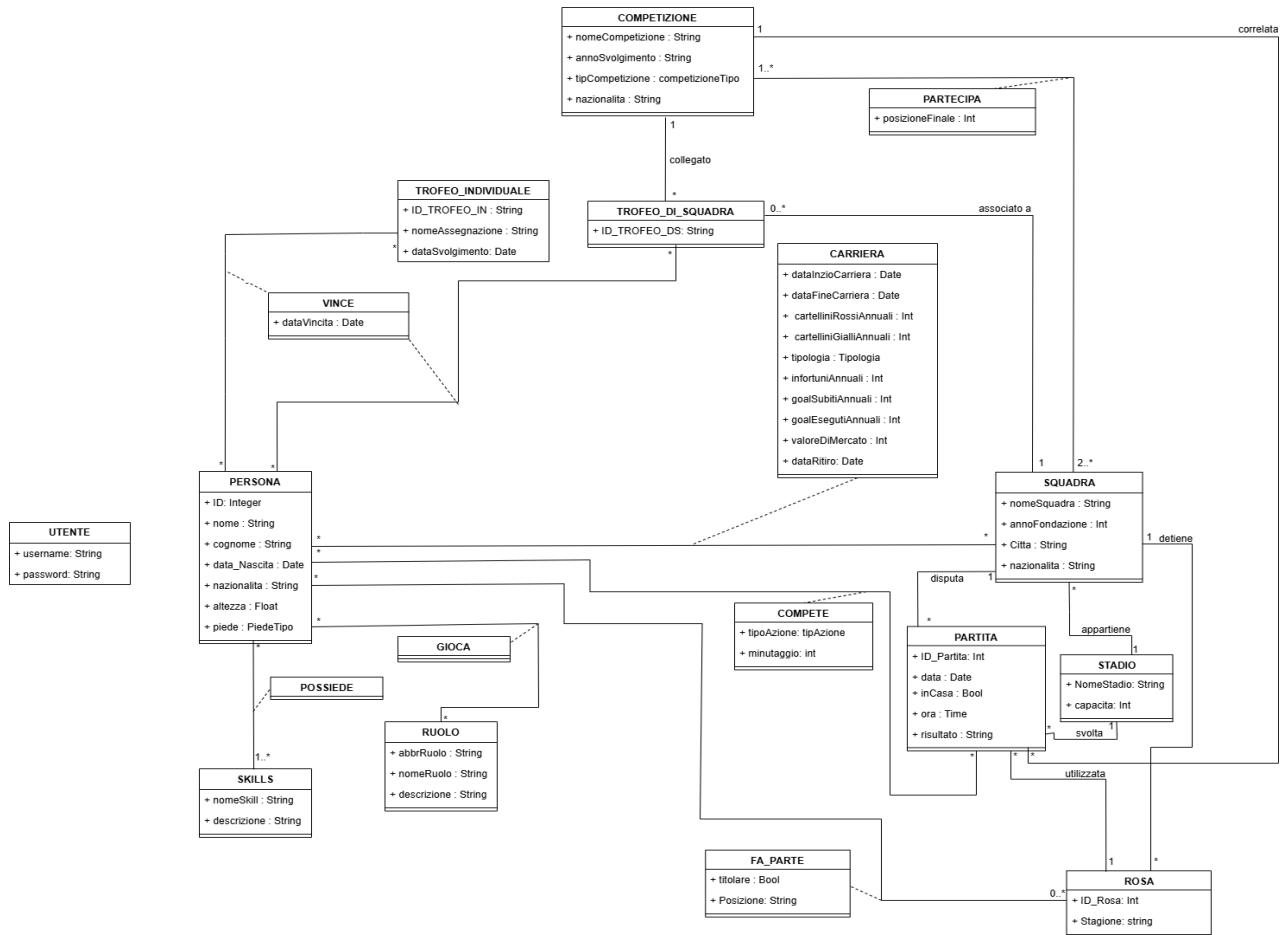
- Abbiamo accorpato *OLIMPIADI*, *MONDIALI*, *CAMPIONATO*, *COPPA* ed *EUROPEI* all'interno dell'entità **COMPETIZIONE**, utilizzando l'attributo *tipCompetizione* per indicarne la tipologia. Inoltre, abbiamo aggiunto all'entità **COMPETIZIONE** l'attributo *nazionalita*, che risulta obbligatorio esclusivamente per la tipologia *campionato*; per tutte le altre tipologie, questo campo non è necessario. Si tratta di una specializzazione **DISGIUNTA TOTALE**, poiché una competizione non può appartenere contemporaneamente a più tipologie diverse, e deve obbligatoriamente rientrare in una delle categorie indicate.
- Abbiamo accorpato *CALCIATORE*, *ALLENATORE* e *DIRIGENTE* nell'entità **CARRIERA**, utilizzando l'attributo *tipologia* per indicarne il ruolo. Si tratta di una specializzazione **DISGIUNTA TOTALE**, poiché un individuo non può ricoprire più ruoli contemporaneamente nella stessa carriera, ma deve concluderne uno prima di intraprenderne un altro. È **totale** perché ogni carriera deve necessariamente essere associata a uno dei ruoli indicati.

- Abbiamo accorpato gli attributi dell’entità *TROFEO* nelle entità **TROFEO_INDIVIDUALE** e **TROFEO_DI_SQUADRA**. Questa specializzazione è **DISGIUNTA TOTALE**, poiché ogni trofeo assegnato deve appartenere almeno a uno dei due tipi, ma non può essere contemporaneamente di entrambi. Questa scelta è stata adottata per gestire separatamente i due tipi di trofeo, che però convergono in una stessa **classe di associazione** denominata *VINCE*.
Per le tipologie di azioni, invece, che comprendono *AMMONIZIONE*, *CARTELLINO GIALLO*, *CARTELLINO ROSSO*, *CAMBIO_ENTRATA*, *CAMBIO_USCITA*, *INFORTUNIO*, *GOAL_ESEGUITO*, *CAPITANO*, sono state accorpate nell’entità compete sotto la denominazione di **TIPOAZIONE**. Per questa specializzazione si parla di **DISGIUNTA TOTALE**, perché ogni azione che viene inserita all’interno della tabella deve essere di almeno una di queste *OBBLIGATORIAMENTE*.

1.4.6 Analisi degli identificativi

Per diverse entità all’interno del sistema, abbiamo deciso di utilizzare chiavi surrogate (ovvero attributi con il prefisso ”ID_”) per gestirle in maniera efficiente. Per altre entità, invece, abbiamo utilizzato gli attributi delle entità stesse, in quanto soddisfacevano tutti i requisiti necessari per essere impiegati come chiavi primarie.

1.5 Class Diagram ristrutturato



1.6 Dizionario Delle Classi

Entità	Descrizione	Attributi
PERSONA	Corrisponde al giocatore e/o all allenatore e/o dirigente che fa parte del popolamento del Database	ID (Int): Identificativo della persona. Nome (String): identifica il nome della persona. Cognome (String): identifica il cognome della persona. DataNascita (Date): corrisponde alla data di nascita della persona. Nazionalita (String): indica la nazionalità della persona. Altezza (Int): Indica l'altezza della persona. Piede (String): Indica quale piede viene utilizzato dal giocatore.
POSSIEDE	Contiene tutte le skills possedute da una persona	
SKILLS	Describe tutte le skills, ovvero particolari abilità/trick, appartenenti alla persona	NomeSkill (String): Nome con il quale viene identificata la skill. Descrizione (String): breve sintesi della skill.
GIOCA	Contiene tutti i ruoli che il giocatore può giocare	
RUOLO	Corrisponde al ruolo/posizione in campo giocata dalla persona	AbbrRuolo (String): Nome del ruolo abbreviato. NomeRuolo (String): Nome completo del ruolo. descrizione (String): Breve descrizione del ruolo.
VINCE	raccoglie tutti i trofei vinti da una determinata persona	dataVincita (Date): Indica quando è stato vinto il trofeo.
TROFEO INDIVIDUALE	Describe tutti i trofei individuali vinti durante la carriera della persona	ID_Trofeo_IN (int): Identificativo del trofeo. nomeAssegnazione (String): Nome proprio del trofeo. dataSvolgimento (Date): Data in cui si è partecipati e si è vinto il trofeo.

Tabella 1: Dizionario delle classi (parte 1)

Entità	Descrizione	Attributi
TROFEO DI SQUADRA	Describe tutti i trofei di squadra vinti dalla persona	ID_Trofeo_DS(Int) : Identificativo del trofeo.
PARTECIPA	contiene tutte le competizioni svolte da una determinata squadra	posizioneFinale(int) : indica il riscontro della competizione per una data squadra.
COMPETIZIONE	Corrisponde alla competizione/torneo giocato dalla persona/squadra	nomeCompetizione(String) : identificativo della competizione. annoSvolgimento(String) : Anno in cui è stata svolta la competizione. tipCompetizione(String) : indica la tipologia della competizione giocata. nazionalita(String) : Indica la nazionalità dove si svolge la competizione.
SQUADRA	Contiene informazioni inerenti alle squadre presenti nel database	nomeSquadra(String) : Identifica il nome della squadra. Citta(String) : Rappresenta la Citta di appartenenza della squadra. nazionalita(String) : Identifica la nazionalità della squadra. annoFondazione(Int) : Anno in cui è stata fondata la squadra.

Tabella 2: Dizionario delle classi (parte 2)

Entità	Descrizione	Attributi
STADIO	Identifica lo stadio che ospita la squadra per le partite in casa	NomeStadio (String): Nome assegnato allo stadio. Capacità (int): Valore con cui si determina la capienza dello stadio.
CARRIERA	Identifica la carriera di una persona in una determinata stagione	dataInizioCarriera (Date): Corrisponde alla data di inizio della carriera di quel giocatore in una squadra. dataFineCarriera (Date): Corrisponde alla data di fine della carriera di quel giocatore in una squadra. cartelliniRossiAnnuali (int): Totale dei cartellini rossi ricevuti durante la stagione. cartelliniGialliAnnuali (int): Totale dei cartellini gialli ricevuti durante la stagione. tipologia (Tipologia): Tipologia di mansione svolta all'interno della squadra che varia da: giocatore, allenatore o dirigente. infortuniAnnuali (int): Totale degli infortuni subiti durante la stagione. goalSubitiAnnuali (int): Variabile che determina se per quella stagione il giocatore ha subito, o meno goal. goalEseguitiAnnuali (int): Variabile che determina se per quella stagione il giocatore ha eseguito, o meno goal. ValoreDiMercato (int): Valore del giocatore per quella stagione. dataRitiro (Date): Data in cui l'utente ha concluso la sua carriera da calciatore/alleantore/dirigente.
COMPETE	Contiene tutte le persone che hanno partecipato ad una specifica partita	tipoAzione (tipAzione): Identifica che tipologia di azione e' stata eseguita. minutaggio (int): Identifica il minuto in cui e' stata eseguita l'azione.

Tabella 3: Dizionario delle classi (parte 3)

Entità	Descrizione	Attributi
PARTITA	Contiene le partite svolte da ogni squadra	ID_Partita(int) : Identifica una determinata partita. Data(Date) : identifica la data di svolgimento di una data partita. inCasa(Boolean) :Definisce se la partita viene disputata in casa o meno. Ora(Time) : identifica l'ora di svolgimento di una data partita. risultato(String) : Indica l'esito della partita svolta.
ROSA	identifica la rosa di una squadra per una partita	ID_Rosa(int) : Identifica una determinata rosa. stagione(Int) : Indica in quale stagione è stata creata ed utilizzata.
FA PARTE	identifica la rosa di cui fa parte una determinata persona	Titolare(Bool) : Describe se una persona è o meno titolare. Posizione(String) : Posizione in cui gioca una determinata persona in quella rosa.

Tabella 4: Dizionario delle classi (parte 4)

1.7 Dizionario Delle Associazioni

Associazione	Descrizione
POSSIEDE	Associazione molti-a-molti: tra skills e persona. Una persona può possedere N skills, e una skill può essere abbinata a N persone diverse.
GIOCA	Associazione molti-a-molti: tra ruolo e persona. Un giocatore può giocare più ruoli, lo stesso ruolo può essere associato a più giocatori.
CARRIERA	Associazione molti-a-molti: tra persona e squadra. Una persona può aver militato in diverse squadre e in una squadra possono aver militato diverse persone.
VINCE	Associazione molti-a-molti: tra persona e Trofeo_Individuale. Una persona può aver vinto più trofei individuali e un trofeo individuale può esser stato vinto da più persone nel corso del tempo.
VINCE	Associazione molti-a-molti: tra persona e Trofeo_di_squadra. Una persona può aver vinto più trofei di squadra e lo stesso trofeo può essere associato a più giocatori.
PARTECIPA	Associazione molti-a-molti: tra squadra e competizione. Una squadra può partecipare da 1 a N competizioni e una competizione può essere composta da 2 o più squadre in gioco.
FA PARTE	Associazione molti-a-molti tra persona e rosa: Un calciatore nella sua carriera può aver giocato in più rose. In una rosa ci sono N giocatori.
DISPUTA	Associazione uno-a-molti tra squadra e partita: Una squadra può aver disputato più partite in casa, ma per ogni tupla di partita troviamo una sola squadra.
CORRELATA	Associazione uno-a-molti tra competizione e partita utilizzata per identificare per ogni partita a quale competizione appartiene.

Tabella 5: Associazioni e descrizioni (parte 1)

DETIENE	Associazione uno-a-molti tra squadra e rosa: Una squadra può avere diverse rose, ma una rosa appartiene esclusivamente ad una squadra.
UTILIZZATA	Associazione uno-a-molti tra rosa e partita: Una rosa può essere utilizzata in più partite. Però in una singola partita la squadra può utilizzare solo una rosa.
ASSOCIATO A	Associazione uno-a-molti tra squadra e Trofeo_di_squadra: Una squadra può esser associata ad uno o più trofei; mentre un trofeo di squadra può essere associato a una sola squadra.
COLLEGATO	Associazione uno-a-molti tra competizione e Trofeo_di_squadra: Una competizione può aver dato vita a più trofei di squadra negli anni, mentre un unico trofeo di squadra è associato a un'unica competizione.
APPARTIENE	Associazione uno-a-molti tra squadra e stadio: Una squadra può avere un solo stadio associato, mentre uno stadio può essere condiviso da più squadre (esempio stadio San Siro, condiviso sia dal Milan che dall'Inter).
SVOLTA	Associazione uno-a-molti tra stadio e partita: Una partita viene svolta in uno stadio, mentre uno stadio può ospitare più partite nel corso del tempo.
COMPETE	Associazione molti-a-molti tra persona e partita: Una persona può competere in più partite e in una stessa partita possono competere N persone.

Tabella 6: Associazioni e descrizioni (parte 2)

1.8 Dizionario dei Vincoli

Nome Vincolo	Tipo	Descrizione
DOMtipAzione	Dominio	Può essere uno dei seguenti valori: ('Ammunizione', 'Goal', 'Infortunio', 'Capitano', 'Cambio_Entrata', 'Cambio_Uscita').
DOMPiedeTipo	Dominio	Può essere uno dei seguenti valori: ('Destro', 'Sinistro', 'Ambidestro').
DOMTipologia	Dominio	Può essere uno dei seguenti valori: ('CALCIATORE', 'ALLENATORE', 'DIRIGENTE').
DOMTipCompetizione	Dominio	Può essere uno dei seguenti valori: ('coppa', 'campionato', 'mondiale', 'europei', 'olimpiadi').
uqTrofeoPersona	Interrelazionali	Gli attributi idTrofeoIN, idTrofeoDS e id devono essere univoci.
ck_trofeo_individuale	Intrarelazionali	Controlla che i trofei individuali inseriti inizino con la sigla IN.
ckMaxCartellini	Intrarelazionale	Controlla che il numero di cartellini gialli sia minore o uguale a 2.

Tabella 7: Dizionario dei vincoli

2 Traduzione modello logico

2.1 Mapping associazioni

Associazioni 1-N

Associazione	Descrizione
Trofeo_di_squadra - Competizione	Inseriamo la chiave primaria di <i>Competizione</i> nell'entità <i>Trofeo_di_squadra</i> come chiave esterna.
Trofeo_di_squadra - Squadra	Inseriamo <i>nomeSquadra</i> dall'entità <i>Squadra</i> , nell'entità <i>Trofeo_di_squadra</i> come chiave esterna.
Partita - Stadio	Inseriamo <i>NomeStadio</i> all'interno dell'entità <i>Partita</i> .
Partita - Rosa	Inseriamo <i>ID_Rosa</i> di <i>Rosa</i> come chiave esterna in <i>Partita</i> (identifica la rosa della squadra che gioca la partita).
Partita - Competizione	Inseriamo in <i>Partita nomeCompetizione</i> .
Squadra - Partita	Inseriamo il <i>nomeSquadra</i> e l'attributo <i>stadio</i> (dell'entità <i>Squadra</i>) nell'entità <i>Partita</i> come chiavi esterne.
Squadra - Stadio	Inseriamo la chiave primaria <i>NomeStadio</i> all'interno dell'entità <i>Squadra</i> .
Squadra - Rosa	Inseriamo il <i>nomeSquadra</i> all'interno dell'entità <i>Rosa</i> come chiave esterna.

Tabella 8: Associazioni 1-N

Associazioni N-N

Associazione	Descrizione
Persona - Skills	<i>POSSIEDE(ID, nomeSkill)</i>
Persona - Ruolo	<i>GIOCA(ID, abbrRuolo)</i>
Persona - Rosa	<i>FA_PARTE(ID, ID_Rosa, titolare, Posizione)</i>
Persona - Partita	<i>COMPETE(ID, ID_Partita, cartellinoRosso, cartelliniGialli, infortunato, goalEseguiti, goalSubiti, capitano)</i>
Persona - Trofeo_Individuale	<i>VINCE(ID_Trofeo_IN, ID, dataVincita)</i>
Persona - Trofeo_Di_Squadra	<i>VINCE(ID_Trofeo_DS, ID, dataVincita)</i>
Persona - Squadra	<i>CARRIERA(ID, nomeSquadra, DataInizioCarriera, DataFineCarriera, cartelliniRossiAnnuali, cartelliniGialliAnnuali, tipologia, infortuniAnnuali, goalSubitiAnnuali, goalEseguitiAnnuali, valoreDiMercato, dataRitiro)</i>
Competizione - Squadra	<i>PARTECIPA(nomeCompetizione, annoSvolgimento, nomeSquadra, posizioneFinale)</i>

Tabella 9: Associazioni N-N

2.2 Modello logico

Entità	Attributi
PERSONA	<u>ID</u> , nome, cognome, dataNascita, nazionalita, altezza, piede <u>ID</u> ↑, <u>nomeSkill</u> ↑
POSSIEDE	POSSIEDE.ID → PERSONA.ID POSSIEDE.nomeSkill → SKILLS.nomeSkill
SKILLS	<u>nomeSkill</u> , descrizione <u>ID</u> ↑, <u>abbrRuolo</u> ↑
GIOCA	GIOCA.ID → PERSONA.ID GIOCA.abbrRuolo → RUOLO.abbrRuolo
RUOLO	<u>abbrRuolo</u> , nomeRuolo, descrizione
TROFEO_INDIVIDUALE	<u>ID_TROFEO_IN</u> , nomeAssegnazione, dataSvolgimento <u>ID_TROFEO_DS</u> , <u>nomeCompetizione</u> ↑, <u>annoSvolgimento</u> ↑, <u>nomeSquadra</u> ↑
TROFEO_DI_SQUADRA	TROFEO_DL_SQUADRA.nomeCompetizione → COMPETIZIONE.nomeCompetizione TROFEO_DL_SQUADRA.annoSvolgimento → COMPETIZIONE.annoSvolgimento TROFEO_DL_SQUADRA.nomeSquadra → SQUADRA.nomeSquadra
COMPETIZIONE	<u>nomeCompetizione</u> , <u>annoSvolgimento</u> , tipCompetizione, nazionalita
STADIO	<u>nomeStadio</u> , capacita
SQUADRA	<u>nomeSquadra</u> , annoFondazione, citta, nazionalita, <u>nomeStadio</u> ↑ SQUADRA.nomeStadio → STADIO.nomeStadio
VINCE	<u>ID_TROFEO_IN</u> ↑, <u>ID_TROFEO_DS</u> ↑, <u>ID</u> ↑, dataVincita VINCE.ID_TROFEO_IN → TROFEO_INDIVIDUALE.ID_TROFEO_IN VINCE.ID_TROFEO_DS → TROFEO_DL_SQUADRA.ID_TROFEO_DS VINCE.ID → PERSONA.ID
PARTECIPA	<u>nomeCompetizione</u> ↑, <u>annoSvolgimento</u> ↑, <u>nomeSquadra</u> ↑, posizioneFinale PARTECIPA.nomeCompetizione → COMPETIZIONE.nomeCompetizione PARTECIPA.annoSvolgimento → COMPETIZIONE.annoSvolgimento PARTECIPA.nomeSquadra → SQUADRA.nomeSquadra

Tabella 10: Modello Logico (parte 1)

Entità	Attributi
CARRIERA	<u>ID</u> ↑, <u>nomeSquadra</u> ↑, <u>dataInizioCarriera</u> , dataFineCarriera, cartelliniRossiAnnuali, cartelliniGialliAnnuali, tipologia, infortuniAnnuali, goalSubitiAnnuali, goalEseguitiAnnuali, valoreDiMercato, dataRitiro CARRIERA.nomeSquadra → SQUADRA.nomeSquadra CARRIERA.ID → PERSONA.ID
ROSA	<u>ID_Rosa</u> , <u>nomeSquadra</u> ↑, stagione ROSA.nomeSquadra → SQUADRA.nomeSquadra
PARTITA	<u>ID_Partita</u> , <u>nomeSquadra</u> ↑, <u>nomeStadio</u> ↑, <u>nomeCompetizione</u> ↑, <u>ID_Rosa</u> ↑, inCasa, data, ora, risultato PARTITA.nomeSquadra → SQUADRA.nomeSquadra PARTITA.ID_Rosa → ROSA.ID_Rosa PARTITA.nomeCompetizione → COMPETIZIONE.nomeCompetizione PARTITA.annoSvolgimento → COMPETIZIONE.annoSvolgimento PARTITA.nomeStadio → COMPETIZIONE.nomeStadio
FA_PARTE	<u>ID_Rosa</u> ↑, <u>ID_Calciatore</u> ↑, titolare, posizione FA_PARTE.ID_Rosa → ROSA.ID_Rosa FA_PARTE.ID_Calciatore → PERSONA.ID
COMPETE	<u>ID</u> ↑, <u>ID_Partita</u> ↑, minutaggio, <u>nomeSquadra</u> ↑, tipoAzione, COMPETE.ID → PERSONA.ID COMPETE.ID_Partita → PARTITA.ID_Partita COMPETE.nomeSquadra → PARTITA.nomeSquadra

Tabella 11: Modello Logico (parte 2)

3 Schema Fisico

3.1 Creazione del database

```
1 Create database Unina Soccer
```

3.2 Creazione entità

Entità Persona

```
1 CREATE DOMAIN DOMPiedeTipo AS VARCHAR(15)
2 CHECK (VALUE IN ('Destro', 'Sinistro', 'Ambidestro'));
```

```
1 CREATE TABLE Persona (
2     ID SERIAL PRIMARY KEY,
3     nome VARCHAR(25) NOT NULL,
4     cognome VARCHAR(25) NOT NULL,
5     data_Nascita DATE NOT NULL,
6     nazionalita VARCHAR(30) NOT NULL,
7     altezza FLOAT NOT NULL,
8     piede DOMPiedeTipo
9 );
```

Entità Ruolo

```
1 CREATE TABLE RUOLO(
2     abbrRuolo VARCHAR(5) PRIMARY KEY,
3     nomeRuolo VARCHAR(40) NOT NULL,
4     descrizione TEXT
5 );
```

Entità Skills

```
1 CREATE TABLE Skills(
2     nomeSkill VARCHAR(30) PRIMARY KEY,
3     descrizione TEXT
4 );
```

Entità Gioca

```
1 CREATE TABLE GIOCA(
2     abbrRuolo VARCHAR(5),
3     ID SERIAL,
4     PRIMARY KEY (ID, abbrRuolo),
5     FOREIGN KEY (abbrRuolo) REFERENCES RUOLO(abbrRuolo),
6     FOREIGN KEY (ID) REFERENCES PERSONA(ID)
7 );
```

Entità Trofeo_Individuale

```
1 CREATE TABLE Trofeo_Individuale(
2     ID_Trofeo_IN VARCHAR(8) PRIMARY KEY,
3     nomeAssegnazione VARCHAR(20) NOT NULL,
4     dataSvolgimento DATE NOT NULL
5     CONSTRAINT ck_trofeo_individuale CHECK (ID_Trofeo_IN LIKE 'IN%')
6 );
```

Entità Competizione

```
1 CREATE DOMAIN DOMTipCompetizione AS VARCHAR(30)
2     CHECK (VALUE IN (
3             'Campionato',
4             'Coppa',
5             'Mondiale',
6             'Olimpiadi',
7             'Europei'
8         ));
```

```
1 CREATE TABLE COMPETIZIONE (
2     nomeCompetizione VARCHAR(30),
3     annoSvolgimento VARCHAR(20),
4     tipCompetizione DOMTipCompetizione,
5     nazionalita VARCHAR(30),
6     PRIMARY KEY (nomeCompetizione, annoSvolgimento)
7 );
```

Entità Stadio

```
1 CREATE TABLE STADIO (
2     nomeStadio VARCHAR(30) PRIMARY KEY,
3     capacita INTEGER
4 );
```

Entità Squadra

```
1 CREATE TABLE SQUADRA (
2     nomeSquadra VARCHAR(30),
3     annoFondazione INTEGER NOT NULL,
4     campAppartenenza VARCHAR(30),
5     nazionalita VARCHAR(30) NOT NULL,
6     nomeStadio VARCHAR(30) NOT NULL,
7     PRIMARY KEY (nomeSquadra)
8     FOREIGN KEY (nomeStadio) REFERENCES STADIO(nomeStadio)
9 );
```

Entità Trofeo_Di_Squadra

```
1 CREATE TABLE TROFEO_DI_SQUADRA(
2     ID_Trofeo_DS VARCHAR(8) PRIMARY KEY,
3     nomeCompetizione VARCHAR(30) NOT NULL,
4     annoSvolgimento VARCHAR(20),
5     nomeSquadra VARCHAR(30) NOT NULL,
6     FOREIGN KEY (nomeCompetizione,annoSvolgimento) REFERENCES
7         → COMPETIZIONE(nomeCompetizione,annoSvolgimento),
8     FOREIGN KEY (nomeSquadra) REFERENCES SQUADRA(nomeSquadra)
9 );
10 ;
```

Entità Vince

```
1 CREATE TABLE VINCE (
2     ID_TROFEO_IN VARCHAR(8),
3     ID_TROFEO_DS VARCHAR(8),
4     ID SERIAL,
5     dataVincita DATE NOT NULL,
6     FOREIGN KEY (ID_TROFEO_IN) REFERENCES
7         → TROFEO_INDIVIDUALE(ID_TROFEO_IN),
8     FOREIGN KEY (ID_TROFEO_DS) REFERENCES
9         → TROFEO_DI_SQUADRA(ID_TROFEO_DS),
10    FOREIGN KEY (ID) REFERENCES PERSONA(ID),
11    CONSTRAINT unique_trofeo_persona UNIQUE (ID_TROFEO_IN, ID_TROFEO_DS,
12        → ID)
13 );
14 ;
```

Entità Partecipa

```
1 CREATE TABLE PARTECIPA (
2     nomeCompetizione VARCHAR(30),
3     annoSvolgimento VARCHAR(20),
4     nomeSquadra VARCHAR(30),
5     posizioneFinale INTEGER,
6     FOREIGN KEY (nomeCompetizione, annoSvolgimento) REFERENCES
7         → COMPETIZIONE(nomeCompetizione, annoSvolgimento),
8     FOREIGN KEY (nomeSquadra) REFERENCES SQUADRA(nomeSquadra)
9 );
10
```

Entità Carriera

```
1 CREATE DOMAIN DOMTipologia VARCHAR(15)
2 CHECK (VALUE IN ('Allenatore', 'Calciatore', 'Dirigente'));
3
```

```
1 CREATE TABLE CARRIERA (
2     ID SERIAL,
3     nomeSquadra VARCHAR(30),
4     dataInizioCarriera DATE NOT NULL,
5     dataFineCarriera DATE,
6     cartelliniRossiAnnuali INTEGER,
7     cartelliniGialliAnnuali INTEGER,
8     tipologia DOMTipologia NOT NULL,
9     infortuniAnnuali INTEGER,
10    goalSubitiAnnuali INTEGER,
11    goalEseguitiAnnuali INTEGER,
12    valoreDiMercato INTEGER,
13    dataRitiro DATE,
14    PRIMARY KEY (ID, nomeSquadra, dataInizioCarriera),
15    FOREIGN KEY (nomeSquadra) REFERENCES SQUADRA(nomeSquadra),
16    FOREIGN KEY (ID) REFERENCES PERSONA(ID)
17 );
18
```

Entità Rosa

```
1 CREATE TABLE ROSA (
2   ID_Rosa SERIAL PRIMARY KEY,
3   nomeSquadra VARCHAR(30),
4   stagione VARCHAR(20),
5   FOREIGN KEY (nomeSquadra) REFERENCES SQUADRA(nomeSquadra)
6 );
```

Entità Partita

```
1 CREATE TABLE PARTITA (
2   ID_Partita INTEGER,
3   nomeSquadra VARCHAR(30) NOT NULL,
4   nomeStadio VARCHAR(30) NOT NULL,
5   nomeCompetizione VARCHAR(30) NOT NULL,
6   annoSvolgimento VARCHAR(30) NOT NULL,
7   ID_Rosa INTEGER NOT NULL,
8   inCasa BOOL NOT NULL,
9   data DATE NOT NULL,
10  ora TIME NOT NULL,
11  risultato VARCHAR(30) NOT NULL,
12  PRIMARY KEY (ID_Partita,nomeSquadra),
13  FOREIGN KEY (nomeSquadra) REFERENCES SQUADRA(nomeSquadra),
14  FOREIGN KEY (ID_Rosa) REFERENCES ROSA(ID_Rosa),
15  FOREIGN KEY (nomeStadio) REFERENCES STADIO(nomeStadio),
16  FOREIGN KEY (nomeCompetizione, annoSvolgimento) REFERENCES
17    → COMPETIZIONE(nomeCompetizione, annoSvolgimento)
);
```

Entità Fa_Parte

```
1 CREATE TABLE FA_PARTE (
2   ID_Rosa SERIAL,
3   ID_Calciatore SERIAL,
4   titolare BOOL NOT NULL,
5   Posizione VARCHAR(40) NOT NULL,
6   PRIMARY KEY (ID_Rosa, ID_Calciatore),
7   FOREIGN KEY (ID_Calciatore) REFERENCES PERSONA(ID),
8   FOREIGN KEY (ID_Rosa) REFERENCES ROSA(ID_Rosa)
9 );
```

Entità Compete

```
1 CREATE TABLE COMPETE (
2   CREATE DOMAIN DOMtipAzione AS VARCHAR(15)
3   CHECK (VALUE IN ('Ammonizione', 'Goal', 'Infortunio', 'Cambio_Entrata',
4   ↳ 'Cambio_Uscita', 'Capitano'));
```

```
1 CREATE TABLE COMPETE (
2   ID SERIAL,
3   ID_Partita INTEGER,
4   nomeSquadra VARCHAR(30) NOT NULL,
5   tipoAzione DOMtipAzione,
6   minutaggio INTEGER,
7   FOREIGN KEY (ID) REFERENCES PERSONA(ID),
8   FOREIGN KEY (ID_Partita, nomeSquadra) REFERENCES
9   ↳ PARTITA(ID_Partita, nomeSquadra)
);
```

3.3 Trigger Function

checkSquadraCamp - Trigger Function dell'entità SQUADRA

Questa funzione verifica se la squadra è associata al campionato selezionato, basandosi sulla nazionalità della squadra indicata nella tupla dell'entità. L'obiettivo della funzione è garantire che tutte le squadre appartenenti allo stesso campionato abbiano la stessa nazionalità.

```
1 CREATE OR REPLACE FUNCTION checkCamp()
2 RETURNS TRIGGER AS $$ 
3 DECLARE
4     CampAnalizzato int;
5 BEGIN
6     SELECT COUNT(*) INTO CampAnalizzato
7     FROM Competizione
8     WHERE Competizione.nomeCompetizione = NEW.campAppartenenza
9     AND Competizione.nazionalita = NEW.nazionalita;
10
11    IF CampAnalizzato < 1 THEN
12        RAISE EXCEPTION 'Un eccezione è stata violata: Il campionato
13            → associato non può essere assegnato alla squadra selezionata o
14            → non esiste';
15    END IF;
16
17    RETURN NEW;
18
19 END;
20 $$ LANGUAGE plpgsql;
21
22 CREATE TRIGGER checkSquadraCamp
23 BEFORE INSERT ON SQUADRA
FOR EACH ROW
EXECUTE FUNCTION checkCamp();
)
```

CheckPosition - Trigger Function dell'entità PARTECIPA

Questa funzione controlla che all'interno di una stessa competizione per uno stesso anno di svolgimento, due squadre non possono avere la stessa posizione finale.

```
1 CREATE OR REPLACE FUNCTION SearchPosition()
2 RETURNS TRIGGER AS $$ 
3 DECLARE
4     CheckPosNEXT int;
5 BEGIN
6     SELECT COUNT(*)
7     INTO CheckPosNEXT
8     FROM Partecipa
9     WHERE nomeCompetizione = NEW.nomeCompetizione
10    AND posizioneFinale = NEW.posizioneFinale
11    AND nomeSquadra <> NEW.nomeSquadra;
12
13    IF CheckPosNEXT > 0 THEN
14        RAISE EXCEPTION 'Un eccezione è stata violata: due squadre non
15        → possono avere la stessa posizione finale!';
16    END IF;
17
18    RETURN NEW;
19 END;
20 $$ LANGUAGE plpgsql;
21
22 CREATE TRIGGER CheckPosition
23 BEFORE INSERT ON Partecipa
24 FOR EACH ROW
25 EXECUTE FUNCTION SearchPosition();
)
```

InserimentoTrofeoDiSquadra - Trigger Function dell'entità PARTECIPA

Questa funzione, generano automaticamente il trofeo per la squadra vincitrice della competizione. In particolare, in abbinamento con questo trigger function, abbiamo allegato una funziona che sarà quella che ci aiutera nel generare gli ID di trofeo di squadra seguendo il pattern deciso da analisi dei requisiti.

```
1 CREATE SEQUENCE trofeo_di_squadra_seq START 1;
2 CREATE OR REPLACE FUNCTION genera_id_trofeo()
3 RETURNS VARCHAR(8) AS $$ 
4 DECLARE
5     nuovo_id INT;
6     id_formattato VARCHAR(8);
7 BEGIN
8
9     nuovo_id = nextval('trofeo_di_squadra_seq');
10
11    id_formattato = 'DS' || lpad(nuovo_id::text, 6, '0');
12
13    RETURN id_formattato;
14 END;
15 $$ LANGUAGE plpgsql;
16
17
18
19
20 CREATE OR REPLACE FUNCTION InserisciTrofeoDiSquadra()
21 RETURNS TRIGGER AS $$ 
22 BEGIN
23
24     IF NEW.posizioneFinale = 1 THEN
25
26         INSERT INTO TROFEO_DI_SQUADRA (ID_Trofeo_DS, nomeCompetizione,
27             annoSvolgimento, nomeSquadra)
28         VALUES (genera_id_trofeo(), NEW.nomeCompetizione,
29             NEW.annoSvolgimento, NEW.nomeSquadra);
30
31     END IF;
32
33     RETURN NEW;
34 END;
35 $$ LANGUAGE plpgsql;
36
37
38 CREATE TRIGGER InserimentoTrofeoDiSquadra
```

```
33  AFTER INSERT ON PARTECIPA
34  FOR EACH ROW
35  EXECUTE FUNCTION InserisciTrofeoDiSquadra();
36 ) ;
```

assegna_premio - Trigger Function dell'entità TROFEO_DS_SQUADRA

Questa funzione consente di assegnare il trofeo vinto dalla a tutti i giocatori che ne facevano parte in quel determinato momento storico.

```
1 CREATE OR REPLACE FUNCTION assegna_premio_calciatori()
2 RETURNS TRIGGER AS $$ 
3 DECLARE
4     DataFinale DATE;
5     ID_Calciatore INTEGER;
6
7     scansionaGiocatore CURSOR FOR
8         SELECT ID
9             FROM Carriera
10            WHERE nomeSquadra = NEW.nomeSquadra
11              AND CAST(EXTRACT(YEAR FROM dataInizioCarriera) AS INTEGER)
12            <= CAST(SPLIT_PART(NEW.annoSvolgimento, '/', 1) AS INTEGER);
13 BEGIN
14     -- Seleziona la data massima della partita
15     SELECT MAX(data) INTO DataFinale
16         FROM PARTITA
17        WHERE nomeSquadra = NEW.nomeSquadra
18          AND annoSvolgimento = NEW.annoSvolgimento
19          AND nomeCompetizione = NEW.nomeCompetizione;
20
21     -- Apri il cursore
22     OPEN scansionaGiocatore;
23
24     -- Ciclo per ogni calciatore nel cursore
25     LOOP
26         FETCH scansionaGiocatore INTO ID_Calciatore;
27         EXIT WHEN NOT FOUND;
28         -- Inserisce il premio nella tabella VINCE
29         INSERT INTO VINCE (ID_TROFEO_IN, ID_TROFEO_DS, ID, dataVincita)
30             VALUES (NULL, NEW.ID_TROFEO_DS, ID_Calciatore, DataFinale);
31     END LOOP;
32
33     -- Chiudi il cursore
34     CLOSE scansionaGiocatore;
35
36     RETURN NEW;
```

```
37 END;
38 $$ LANGUAGE plpgsql;
39 -- Creazione della trigger associata alla tabella TROFEO_DI_SQUADRA
40 CREATE TRIGGER assegna_premio
41 AFTER INSERT ON TROFEO_DI_SQUADRA
42 FOR EACH ROW
43 EXECUTE FUNCTION assegna_premio_calciatori();
44 ) ;
```

CheckID_Partita - Trigger Function dell'entità PARTITA

Controlla che solo due squadre abbiano lo stesso ID Partita, verificando che le due squadre si stiano effettivamente sfidando.

```
1 CREATE OR REPLACE FUNCTION CountID_Partita()
2 RETURNS TRIGGER AS $$ 
3 DECLARE
4     CheckTotalID int;
5 BEGIN
6     -- Conta il numero di squadre con lo stesso ID_partita
7     SELECT COUNT(*) INTO CheckTotalID
8     FROM Partita
9     WHERE ID = NEW.ID_partita;
10
11    -- non possono esserci più di due squadre nella stessa partita
12    IF CheckTotalID >= 2 THEN
13        RAISE EXCEPTION 'Solo due squadre possono partecipare alla stessa
14        → partita!';
15    END IF;
16
17    RETURN NEW;
18
19
20    LANGUAGE plpgsql;
21
22
23    CREATE TRIGGER CheckID_Partita
24    BEFORE INSERT ON Partita
25    FOR EACH ROW
26    EXECUTE FUNCTION CountID_Partita();
```

Check_inCasa - Trigger Function dell'entità PARTITA

La funzione, quando una tupla imposta il valore di inCasa su TRUE, verifica innanzitutto che esista una sola tupla con inCasa impostato su TRUE. Se non ci sono conflitti, procede a controllare che lo stadio della partita corrisponda a quello della squadra che ha inCasa impostato su TRUE.

```
1 CREATE OR REPLACE FUNCTION Check_inCasaCorrect()
2 RETURNS TRIGGER AS $$
3 DECLARE
4     CheckTotal_inCasa int;
5     StadioSquadra varchar(100);
6 BEGIN
7
8
9
10    IF NEW.inCasa = TRUE THEN
11
12        SELECT COUNT(*) INTO CheckTotal_inCasa
13        FROM Partita
14        WHERE inCasa = TRUE
15        AND ID_partita = NEW.ID_Partita;
16
17
18    IF CheckTotal_inCasa > 0 THEN
19        RAISE EXCEPTION 'Solo una squadra può giocare in casa per la
20                           → stessa partita!';
21
22
23    SELECT S.Stadio INTO StadioSquadra
24    FROM SQUADRA AS S
25    WHERE S.nomeSquadra = NEW.nomeSquadra;
26
27
28    IF NEW.nomeStadio <> StadioSquadra THEN
29        RAISE EXCEPTION 'Lo stadio della partita non è associato alla
30                           → squadra che gioca in casa';
31
32    END IF;
33
34    END IF;
35
```

```
33      RETURN NEW;
34
35  $$ LANGUAGE plpgsql;
36
37
38 CREATE TRIGGER Check_inCasa
39 BEFORE INSERT ON Partita
40 FOR EACH ROW
41 EXECUTE FUNCTION Check_inCasaCorrect();
```

check_Convocati_Titolari - Trigger Function dell'entità FA_PARTE

La funzione gestisce i calciatori convocati per una determinata partita, assicurandosi che il numero sia conforme a un quantitativo realistico. Inoltre, include l'allenatore, poiché anche quest'ultimo può essere soggetto a sanzioni.

```
1  -- Funzione per il trigger
2  CREATE OR REPLACE FUNCTION CheckTotalConvocati_Titolari()
3  RETURNS TRIGGER AS $$
4  DECLARE
5      TotalConvocati INT;
6      CheckAllenatore INT;
7      TotalTitolari INT;
8  BEGIN
9
10     SELECT count(*) INTO TotalConvocati
11     FROM FA_PARTE
12     WHERE ID_Rosa = New.ID_Rosa;
13
14     SELECT count(*) INTO CheckAllenatore
15     FROM FA_PARTE
16     WHERE ID_Rosa = New.ID_Rosa AND posizione = 'Allenatore';
17
18     IF TotalConvocati > 21 THEN
19         RAISE EXCEPTION 'Numero massimo di giocatori convocati
20                         → consentito';
21     END IF;
22
23     IF TotalConvocati = 21 AND CheckAllenatore = 0 THEN
24         RAISE EXCEPTION 'Allenatore non convocato e numero
25                         → massimo di giocatori convocabili consentito';
26     END IF;
27
28     SELECT count(*) INTO TotalTitolari
29     FROM FA_PARTE
30     WHERE ID_Rosa = New.ID_Rosa
31     AND Titolare = TRUE;
32
33     IF TotalTitolari > 11 THEN
34         RAISE EXCEPTION 'Numero massimo di giocatori titolari
35                         → consentito';
```

```
33      END IF;
34
35      RETURN NEW;
36
37  $$ LANGUAGE plpgsql;
38
39 -- Definizione del trigger
40 CREATE TRIGGER Check_Convocati_Titolari
41 AFTER INSERT ON FA_PARTE
42 FOR EACH ROW
43 EXECUTE FUNCTION CheckTotalConvocati_Titolari();
```

CheckPlayer - Trigger Function dell'entità FA_PARTE

Questa funzione controlla se effettivamente il ruolo associato a quel giocatore per quella determinata rosa, risulta essere un ruolo giocato dal medesimo calciatore.

```
1 CREATE OR REPLACE FUNCTION ExistRoleAndPlay()
2 RETURNS TRIGGER AS $$ 
3 DECLARE
4     PosizioneDaControllare VARCHAR(30);
5     RuoloConfermato VARCHAR(10);
6 BEGIN
7     IF NEW.posizione <> 'Allenatore' THEN
8         IF NOT EXISTS (SELECT 1 FROM RUOLO WHERE nomeRuolo =
9             → NEW.posizione) THEN
10            RAISE EXCEPTION 'Il ruolo inserito non esiste!';
11        END IF;
12
13        SELECT G.abbrRuolo INTO RuoloConfermato
14        FROM GIOCA AS G
15        JOIN RUOLO AS R ON G.abbrRuolo = R.abbrRuolo
16        WHERE G.ID = NEW.ID_Calciatore AND R.nomeRuolo =
17            → NEW.posizione;
18
19        IF RuoloConfermato IS NULL THEN
20            RAISE EXCEPTION 'Il giocatore non gioca quel
21            → ruolo!';
22        END IF;
23    END IF;
24    RETURN NEW;
25
26
27 END;
28 $$ LANGUAGE plpgsql;
29
30
31 CREATE TRIGGER CheckPlayer
32 BEFORE INSERT ON FA_PARTE
33 FOR EACH ROW
34 EXECUTE FUNCTION ExistRoleAndPlay();
```

updateCareers - Trigger Function dell'entità COMPETE

Questa funzione permette di aggiornare tutti i valori della carriera di un determinato calciatore a seguito di una partita svolta.

```
1 CREATE OR REPLACE FUNCTION UpdatePlayerCareer()
2 RETURNS TRIGGER AS $$ 
3 DECLARE
4     CountCartelliniGialli INTEGER;
5     avversarioPortiereID INTEGER;
6     squadraAvversaria VARCHAR(30);
7     secondoPortiereID INTEGER;
8     ControlloCambioRiuscito INTEGER;
9 BEGIN
10
11     IF NEW.tipoAzione = 'Cambio_Uscita' THEN
12         SELECT 1 INTO ControlloCambioRiuscito
13         FROM COMPETE
14         WHERE nomeSquadra = NEW.nomeSquadra
15         AND minutaggio = NEW.minutaggio
16         AND tipoAzione = 'Cambio_Eintrata'
17         AND ID_Partita = NEW.ID_Partita;
18
19         IF ControlloCambioRiuscito IS NULL THEN
20             RAISE EXCEPTION 'Il cambio non ha riportato un membro
21             → entrante nella partita';
22         END IF;
23
24         -- Gestione delle ammonizioni
25         IF NEW.tipoAzione = 'Ammonizione' THEN
26             SELECT count(*) INTO CountCartelliniGialli
27             FROM COMPETE
28             WHERE ID = NEW.ID
29             AND tipoAzione = 'Ammonizione';
30
31         IF CountCartelliniGialli > 1 THEN
32             UPDATE CARRIERA
33             SET cartelliniRossiAnnuali = cartelliniRossiAnnuali + 1
34             WHERE ID = NEW.ID;
35     END IF;
```

```

36
37     UPDATE CARRIERA
38     SET cartelliniGialliAnnuali = cartelliniGialliAnnuali + 1
39     WHERE ID = NEW.ID;
40 END IF;

41
42 -- Gestione degli infortuni
43 IF NEW.tipoAzione = 'Infortunio' THEN
44     UPDATE CARRIERA
45     SET infortuniAnnuali = infortuniAnnuali + 1
46     WHERE ID = NEW.ID;
47 END IF;

48
49 -- Gestione dei goal
50 IF NEW.tipoAzione = 'Goal' THEN
51     -- Incrementa i goal segnati per il giocatore che ha effettuato
52     -- il goal
53     UPDATE CARRIERA
54     SET goalEseguitiAnnuali = goalEseguitiAnnuali + 1
55     WHERE ID = NEW.ID;

56
57     -- Determina la squadra avversaria
58     SELECT DISTINCT C.nomeSquadra
59     INTO squadraAvversaria
60     FROM COMPETE AS C
61     WHERE C.ID_Partita = NEW.ID_Partita
62     AND C.nomeSquadra <> NEW.nomeSquadra;

63
64     -- Trova il portiere titolare della squadra avversaria
65     SELECT FP.ID_Calciatore
66     INTO avversarioPortiereID
67     FROM FA_PARTE AS FP
68     JOIN ROSA AS R ON FP.ID_Rosa = R.ID_Rosa
69     WHERE R.nomeSquadra = squadraAvversaria
70     AND FP.Posizione = 'Portiere'
71     AND FP.titolare = TRUE
72     AND NOT EXISTS (
73         SELECT 1
74         FROM COMPETE
75         WHERE ID = FP.ID_Calciatore
76         AND tipoAzione = 'Cambio_Uscita'
77         AND minutaggio <= NEW.minutaggio

```

```

77      );
78
79      -- Se il portiere titolare è uscito, trova il secondo portiere
80      IF avversarioPortiereID IS NULL THEN
81          SELECT FP.ID_Calciatore
82              INTO secondoPortiereID
83              FROM FA_PARTE AS FP
84              JOIN ROSA AS R ON FP.ID_Rosa = R.ID_Rosa
85              WHERE R.nomeSquadra = squadraAvversaria
86                  AND FP.Posizione = 'Portiere'
87                  AND FP.titolare = FALSE
88                  AND EXISTS (
89                      SELECT 1
90                      FROM COMPETE
91                      WHERE ID = FP.ID_Calciatore
92                          AND tipoAzione = 'Cambio_Entrata'
93                          AND minutaggio <= NEW.minutaggio
94                  )
95                  LIMIT 1; -- Trova il primo portiere valido
96      END IF;
97
98      -- Determina quale portiere aggiornare
99      IF avversarioPortiereID IS NOT NULL THEN
100          -- Aggiorna il portiere titolare
101          UPDATE CARRIERA
102              SET goalSubitiAnnuali = goalSubitiAnnuali + 1
103              WHERE ID = avversarioPortiereID;
104      ELSIF secondoPortiereID IS NOT NULL THEN
105          -- Aggiorna il secondo portiere
106          UPDATE CARRIERA
107              SET goalSubitiAnnuali = goalSubitiAnnuali + 1
108              WHERE ID = secondoPortiereID;
109      END IF;
110  END IF;
111
112      RETURN NULL;
113  END;
114  $$ LANGUAGE plpgsql;
115
116  -- Trigger associato alla tabella COMPETE
117  CREATE TRIGGER updateCareers
118  AFTER INSERT ON COMPETE

```

```
119 FOR EACH ROW  
120 EXECUTE FUNCTION UpdatePlayerCareer();
```

CheckOnCarriera - Trigger Function dell'entità CARRIERA

Questa funzione, permette di sancire la fine di una data carriera da calciatore.
Ovviamente questa funzione ci serve per gestire tutti quei calciatori che poi sono diventati effettivamente allenatori, e sono presenti nel nostro database.

```
1 CREATE OR REPLACE FUNCTION CheckDataCarriera()
2 RETURNS TRIGGER AS $$ 
3 BEGIN
4
5     IF (NEW.dataFineCarriera IS NOT NULL AND NEW.dataFineCarriera <
6         → NEW.dataInizioCarriera) THEN
7         RAISE EXCEPTION 'Data fine carriera del Giocatore non valida.';
8     END IF;
9
10    -- Check dataRitiro is present and valid
11    IF (NEW.dataRitiro IS NOT NULL AND NEW.dataRitiro <
12        → NEW.dataFineCarriera) THEN
13        RAISE EXCEPTION 'Data Ritiro non valida.';
14    END IF;
15
16    RETURN NEW;
17 LANGUAGE plpgsql;
18
19 CREATE TRIGGER CheckOnCarriera
20 BEFORE INSERT ON CARRIERA
21 FOR EACH ROW
22 EXECUTE FUNCTION CheckDataCarriera();
23
```

CheckRitiroPlayer - Trigger Function dell'entità CARRIERA

Questa funzione completa la precedente e serve a confermare che un determinato allenatore o dirigente abbia effettivamente avuto una carriera come calciatore, conclusasi in quel ruolo.

```
1 CREATE OR REPLACE FUNCTION CheckRitiroPlayerCorrect ()  
2 RETURNS TRIGGER AS $$  
3 DECLARE  
4     calciatore_ritirato_exists BOOLEAN;  
5 BEGIN  
6  
7     IF (NEW.tipologia <> 'Calciatore') THEN  
8  
9         SELECT EXISTS (  
10             SELECT 1  
11                 FROM CARRIERA  
12                 WHERE ID = NEW.ID  
13                     AND tipologia = 'Calciatore'  
14                     AND dataRitiro IS NOT NULL  
15         ) INTO calciatore_ritirato_exists;  
16  
17  
18     IF NOT calciatore_ritirato_exists THEN  
19         RAISE EXCEPTION 'Il giocatore non risulta Ritirato dal ruolo  
→ Calciatore.';  
20     END IF;  
21     END IF;  
22  
23     RETURN NEW;  
24  
25 LANGUAGE plpgsql;  
26  
27 CREATE TRIGGER CheckRitiroPlayer  
28 BEFORE INSERT ON CARRIERA  
29 FOR EACH ROW  
30 EXECUTE FUNCTION CheckRitiroPlayerCorrect();
```

check_nazionalita - Trigger Function dell'entità COMPETIZIONE

Se una data competizione è di tipo "Campionato", il campo nazionalità non può essere nullo.

```
1 CREATE OR REPLACE FUNCTION check_nazionalita()
2 RETURNS TRIGGER AS $$ 
3 BEGIN
4     IF NEW.tipCompetizione = 'Campionato' AND NEW.nazionalita IS NULL
5         THEN
6             RAISE EXCEPTION 'Il campo nazionalita non può essere NULL per le
7                 competizioni di tipo Campionato';
8         END IF;
9     RETURN NEW;
10 END;
11 $$ LANGUAGE plpgsql;
12
13 CREATE TRIGGER check_nazionalita
14 BEFORE INSERT OR UPDATE ON COMPETIZIONE
15 FOR EACH ROW
16 EXECUTE FUNCTION check_nazionalita();
```

3.4 Popolamento del Database

All'interno di questa sezione andremo ad alizzare in maniera generale tutto il popolamento gestito dal DB non entrando nello specifico di ogni singola insert; questo ci permette di mostrare il ragionamento applicato senza andar a rovinare la flessibilità della documentazione. Ovviamente il codice esterno è fornito delle insert complete.

entità PERSONA

```
1 ALTER SEQUENCE public.persona_id_seq RESTART WITH 1;
2 SET datestyle = 'YMD';
3 --Napoli
4 INSERT INTO PERSONA (nome, cognome, data_nascita, nazionalita, altezza,
5   → piede)
6 VALUES
7   ('Alex', 'Meret', '1997-03-22', 'Italia', 1.90, 'Sinistro'),
8   ...
9   ('Antonio', 'Conte', '1969-07-31', 'Italia', 1.76, 'Destro'),
10  ('Aurelio', 'De Laurentiis', '1949-05-24', 'Italia', 1.75, NULL);
11
12 -- Milan
13 INSERT INTO PERSONA (nome, cognome, data_Nascita, nazionalita, altezza,
14   → piede)
15 VALUES
16   ('Mike', 'Maignan', '1995-07-03', 'Francia', 1.91, 'Destro'),
17   ...
18   ('Stefano', 'Pioli', '1965-10-20', 'Italia', 1.82, 'Destro'),
19   ('Paolo', 'Maldini', '1968-06-26', 'Italia', 1.88, 'Sinistro');
20
21 -- Inter
22 INSERT INTO PERSONA (nome, cognome, data_nascita, nazionalita, altezza,
23   → piede)
24 VALUES
25   ('Yann', 'Sommer', '1988-12-18', 'Svizzera', 1.83, 'Destro'),
26   ...
27   ('Simone', 'Inzaghi', '1976-04-05', 'Italia', 1.85, 'Destro'),
28   ('Giuseppe', 'Marotta', '1957-03-27', 'Italia', 1.85, NULL);
```

entità COMPETIZIONE

```
1 -- 2023/2024
2 INSERT INTO COMPETIZIONE (nomeCompetizione, annoSvolgimento,
3   tipCompetizione, nazionalita)
4 VALUES
5   ('Serie A', '2023/2024', 'Campionato', 'Italia'),
6   ('Premier League', '2023/2024', 'Campionato', 'Inghilterra'),
7   ('La Liga', '2023/2024', 'Campionato', 'Spagna'),
8   ('Bundesliga', '2023/2024', 'Campionato', 'Germania'),
9   ('Ligue 1', '2023/2024', 'Campionato', 'Francia'),
10  ('Champions League', '2023/2024', 'Coppa', 'Europa'),
11  ('Europa League', '2023/2024', 'Coppa', 'Europa'),
12  ('Coppa del Mondo', '2023/2024', 'Mondiale', 'Internazionale'),
13  ('Olimpiadi di calcio', '2023/2024', 'Olimpiadi', 'Internazionale'),
14  ('Campionato Europeo', '2023/2024', 'Europei', 'Europa'),
15  ('Coppa Italia', '2023/2024', 'Coppa', 'Italia'),
16  ('FA Cup', '2023/2024', 'Coppa', 'Inghilterra'),
17  ('Copa del Rey', '2023/2024', 'Coppa', 'Spagna'),
18  ('DFB-Pokal', '2023/2024', 'Coppa', 'Germania'),
19  ('Coupe de France', '2023/2024', 'Coppa', 'Francia');
```

entità STADIO

```
1 INSERT INTO STADIO (nomeStadio, capacita)
2 VALUES
3   ('Stadio Diego Armando Maradona', 56726),
4   ('Giuseppe Meazza', 75923),
5   ('Santiago Bernabéu', 81044),
6   ('Wembley Stadium', 90000);
```

entità SQUADRA

```
1  INSERT INTO SQUADRA (nomeSquadra, annoFondazione, campAppartenenza,
2    ↪ nazionalita, nomeStadio)
3
4  VALUES
5    ('SSC Napoli', 1926, 'Serie A', 'Italia', 'Stadio Diego Armando
6      ↪ Maradona'),
7    ('AC Milan', 1899, 'Serie A', 'Italia', 'Giuseppe Meazza'),
8    ('Inter', 1908, 'Serie A', 'Italia', 'Giuseppe Meazza');
```

entità SKILLS

```
1  INSERT INTO SKILLS (nomeSkill, descrizione)
2
3  VALUES
4    ('Controllo di palla', 'Capacità di ricevere e mantenere il
5      ↪ possesso del pallone sotto pressione.'),
6    ...
7    ('Intelligenza calcistica', 'Comprendere il gioco e prendere
8      ↪ decisioni che aumentano le probabilità di successo per la
9      ↪ squadra.'),
10   ('Autocontrollo', 'Gestire le emozioni e mantenere la disciplina
11     ↪ durante il gioco.');
```

entità POSSIEDE

```
1  INSERT INTO POSSIEDE (id, nomeSkill)
2
3  VALUES
4    (1, 'Controllo di palla'), (1, 'Tiro'), (1, 'Passaggio'),
5    ...
6    (25, 'Velocità'), (25, 'Forza'), (25, 'Tiro');
```

entità RUOLO

```
1  INSERT INTO RUOLO (abbrRuolo, nomeRuolo, Descrizione)
2  VALUES
3      ('POR', 'Portiere', 'Il giocatore che ha il compito di difendere la
4      → porta.'),
5      ('DIF', 'Difensore', 'Il giocatore che ha il compito di difendere la
6      → propria area e impedire agli avversari di segnare.'),
7      ('DC', 'Difensore Centrale', 'Difensore che gioca al centro della
8      → difesa'),
9      ('DCD', 'Difensore Centrale Destro', 'Difensore centrale che gioca
10     → principalmente sul lato destro del campo.'),
11      ('DCS', 'Difensore Centrale Sinistro', 'Difensore centrale che gioca
12     → principalmente sul lato sinistro del campo.'),
13      ('TD', 'Terzino Destro', 'Difensore che gioca principalmente sulla
14     → fascia destra.'),
15      ('TS', 'Terzino Sinistro', 'Difensore che gioca principalmente sulla
16     → fascia sinistra.'),
17      ('CC', 'Centrocampista Centrale', 'Giocatore che gioca nella parte
18     → centrale del campo, spesso con compiti sia difensivi che
19     → offensivi.'),
20      ('CDC', 'Centrocampista Difensivo Centrale', 'Centrocampista con
21     → compiti principalmente difensivi, situato davanti alla difesa.'),
22      ('CO', 'Centrocampista Offensivo Centrale', 'Centrocampista con
23     → compiti principalmente offensivi, situato dietro gli
24     → attaccanti.'),
25      ('AD', 'Ala Destra', 'Giocatore offensivo che gioca principalmente
26     → sulla fascia destra.'),
27      ('AS', 'Ala Sinistra', 'Giocatore offensivo che gioca principalmente
28     → sulla fascia sinistra.'),
29      ('ATT', 'Attaccante', 'Giocatore il cui principale compito è segnare
30     → i gol.'),
31      ('ATTDS', 'Attaccante Seconda Punta', 'Attaccante che gioca vicino
32     → alla punta principale e si muove tra centrocampo e attacco.'),
33      ('ATTPC', 'Attaccante Punta Centrale', 'Attaccante che gioca
34     → principalmente nella posizione centrale dell attacco.');
35
```

entità GIOCA

```
1  INSERT INTO GIOCA (ID, abbrRuolo)
2  VALUES
3      --Portieri
4      (1, 'POR'), -- Alex Meret
5      (2, 'POR'), -- Pierluigi Gollini
6      ...
7      (16, 'CC'), (16, 'CDC'), -- Frank Anguissa
8      (17, 'CC'), (17, 'CO'), -- Piotr Zieliński
9      ...
10     (25, 'ATT'), (25, 'ATTPC'), -- Victor Osimhen
11     (26, 'ATT'), (26, 'ATTPC'), -- Giovanni Simeone
12
13     -- Allenatore
14     (27, 'CC'), (27, 'AD'); -- Antonio Conte
```

entità TROFEO_INDIVIDUALE

```
1  INSERT INTO Trofeo_Individuale (ID_Trofeo_IN, nomeAssegnazione,
2  →   dataSvolgimento) VALUES
3  ('IN000001', 'Pallone d''Oro', '2024-10-30'),
4  ('IN000002', 'Scarpa d''Oro', '2024-06-01'),
5  ('IN000003', 'Miglior Portiere', '2024-10-30'),
6  ('IN000004', 'Miglior Difensore', '2024-09-24'),
7  ('IN000005', 'Miglior Centrocampista', '2024-09-24'),
8  ('IN000006', 'Miglior Attaccante', '2024-09-24'),
9  ('IN000007', 'Giocatore dell''Anno', '2024-10-17'),
10  ('IN000008', 'Trofeo Kopa', '2024-10-30'),
11  ('IN000009', 'Golden Boy', '2024-12-18'),
12  ('IN000010', 'FIFA World Player', '2024-09-24');
```

entità VINCE

```
1  INSERT INTO VINCE (ID_TROFEO_IN, ID_TROFEO_DS, ID, dataVincita) VALUES
2  -- Victor Osimhen vince la Scarpa d'Oro
3  ('IN000002', NULL, 26, '2024-06-01'),
4  -- Alex Meret vince il Miglior Portiere
5  ('IN000003', NULL, 1, '2024-10-30'),
6  -- Amir Rahmani vince il Miglior Difensore
7  ('IN000004', NULL, 5, '2024-09-24'),
8  -- Davide Frattesi vince il Miglior Centrocampista
9  ('IN000005', NULL, 76, '2024-09-24'),
10 -- Khvicha Kvaratskhelia vince il Miglior Attaccante
11 ('IN000006', NULL, 22, '2024-09-24'),
12 -- Olivier Giroud vince il Giocatore dell'Anno
13 ('IN000007', NULL, 54, '2024-10-17'),
14 -- Jens Cajuste vince il Trofeo Kopa
15 ('IN000008', NULL, 19, '2024-10-30'),
16 -- Matteo Politano vince il Golden Boy
17 ('IN000009', NULL, 23, '2024-12-18'),
18 -- Ruben Loftus-Cheek vince il FIFA World Player
19 ('IN000010', NULL, 45, '2024-09-24'),
20 -- Rafael Leao vince il Pallone d'Oro
21 ('IN000001', NULL, 48, '2024-10-30');
```

entità PARTECIPA

```
1  INSERT INTO PARTECIPA (nomeCompetizione, annoSvolgimento, nomeSquadra,
2  →   posizioneFinale)
3  VALUES
4  -- Serie A 2023/2024
5  ('Serie A', '2023/2024', 'SSC Napoli', 4),
6  ('Serie A', '2023/2024', 'Inter', 6),
7  ('Serie A', '2023/2024', 'AC Milan', 1), --Vincitore
8  ...
9  ('Coppa Italia', '2023/2024', 'AC Milan', 3); -- Quarti di finale
```

entità CARRIERA

```
1  INSERT INTO CARRIERA (ID, nomeSquadra, dataInizioCarriera,
2    →  dataFineCarriera, cartelliniRossiAnnuali, cartelliniGialliAnnuali,
3    →  tipologia, infortuniAnnuali, goalSubitiAnnuali, goalEseguitiAnnuali,
4    →  valoreDiMercato, dataRitiro)
5
6  VALUES
7
8    →      -- Portieri
9
10   (1, 'SSC Napoli', '2018-07-01', NULL, 0, 0, 'Calciatore', 0, 0, 0,
11    →  3500000, NULL), -- Alex Meret (Portiere)
12
13   ...
14
15   (27, 'SSC Napoli', '2022-06-10', NULL, 0, 0, 'Allenatore', 0, 0, 0,
16    →  5000000, '2006-03-13'); -- Antonio Conte (Allenatore)
```

entità ROSA

```
1  INSERT INTO ROSA (nomeSquadra, Stagione)
2
3  VALUES
4
5  ('SSC Napoli','2023/2024'),
6  ('AC Milan','2023/2024'),
7  ('Inter', '2023/2024');
```

entità FA_PARTE

```
1
2  --NAPOLI
3
4  INSERT INTO FA_PARTE (ID_Rosa, ID_Calciatore, titolare, Posizione)
5
6  VALUES
7
8    →  (1, 1,TRUE, 'Portiere'), -- Alex Meret
9
10   ...
11
12   (1, 26, FALSE, 'Attaccante Punta Centrale'), -- Giovanni Simeone
13
14   (1, 27, FALSE, 'Allenatore'); -- Antonio Conte
```

entità PARTITA

```
1 -- 'Serie A' 2023/2024
2
3 -- SSC Napoli vs AC Milan
4 INSERT INTO PARTITA (ID_Partita, nomeSquadra, nomeCompetizione,
5   → annoSvolgimento, ID_Rosa, inCasa, data, ora, risultato, nomeStadio)
6 VALUES
7   → (1, 'SSC Napoli', 'Serie A', '2023/2024', 1, TRUE, '2023-09-10',
8     → '20:45:00', '0', 'Stadio Diego Armando Maradona'),
9   → (1, 'AC Milan', 'Serie A', '2023/2024', 2, FALSE, '2023-09-10',
10    → '20:45:00', '1', 'Stadio Diego Armando Maradona');
11 ...
12
```

entità COMPETE

```
1 -- Inter vs AC Milan (Champions League, ID_Partita = 4)
2 INSERT INTO COMPETE (ID, ID_Partita, nomeSquadra, tipoAzione, minutaggio)
3 VALUES
4   → (87, 4, 'Inter', 'Capitano', 0), -- Lautaro Martinez (capitano Inter)
5   → (40, 4, 'AC Milan', 'Capitano', 0), -- Davide Calabria (capitano
6     → Milan)
7   → (49, 4, 'AC Milan', 'Goal', 10), -- Rafael Leão segna
8   → (51, 4, 'AC Milan', 'Goal', 80), -- Christian Pulisic segna
9   → (75, 4, 'Inter', 'Ammonizione', 55), -- Nicòlò Barella ammonito
10  → (90, 4, 'Inter', 'Cambio_Entrata', 65), -- Alexis Sanchez entra
11  → (88, 4, 'Inter', 'Cambio_Uscita', 65); -- Marcus Thuram esce
```

Ricordiamo che questo popolamento e discussione del codice è un riassunto dell’effettivo database instaurato dal gruppo. L’intero codice completo è presente nella repository di GitHub.

La versione di **PostgreSQL** utilizzata per consultare e verificare la validità del codice è la **versione 16**.

4 Object Orientation

4.1 Introduzione

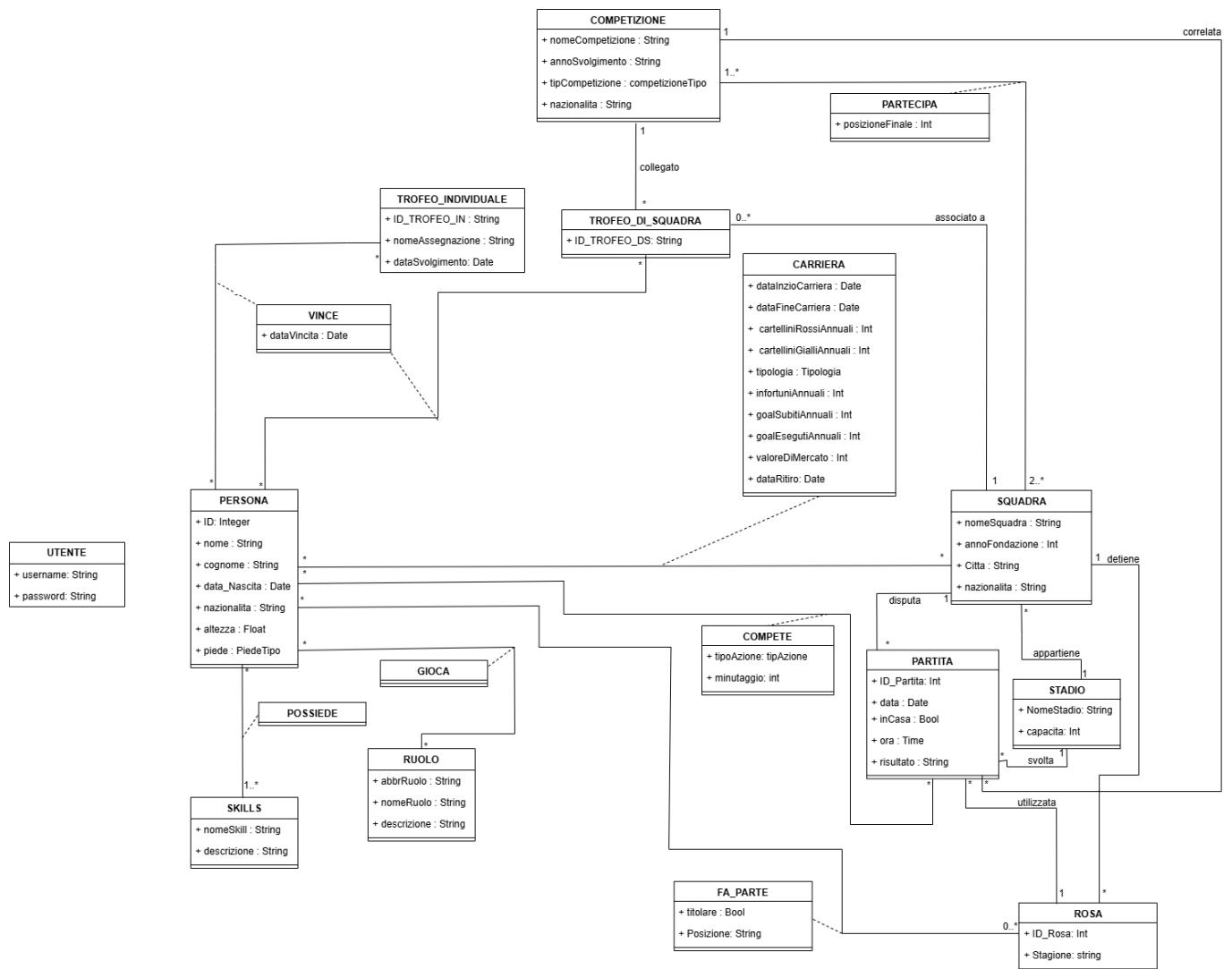
L'applicativo sviluppato ha l'obiettivo di consentire la visualizzazione e la modifica del database contenente le informazioni relative ai calciatori, precedentemente realizzato. In fase progettuale, il focus principale è stato rivolto all'identificazione delle tipologie di utenti che potranno accedere e utilizzare il sistema. A tale scopo, sono state individuate e definite tre distinte categorie di utenti, caratterizzate da specifici livelli di autorizzazione, definiti con il termine “Ruolo”. Questo attributo permette di riconoscere, all'interno del database, le autorizzazioni assegnate ad ogni utente registrato. Il primo livello di accesso è rappresentato dall'utente non ancora presente nel database, che, registrandosi, potrà effettuare il login e accedere esclusivamente alla ricerca e alla visualizzazione delle informazioni richieste. Il secondo livello comprende il ruolo “Calciatore”, il quale include, per semplificazione, anche le funzioni tipiche di allenatori e dirigenti. Gli utenti con questo ruolo, oltre ad avere i privilegi di base, potranno modificare i dati relativi al proprio profilo, limitatamente alle informazioni inerenti alla carriera sportiva, escludendo i dati di accesso quali email e password. Infine, il ruolo “Admin” conferisce all'utente la possibilità di apportare modifiche e aggiungere informazioni in tutto il database, operazione resa possibile mediante una specifica interfaccia grafica, in cui alcuni pulsanti saranno visibili esclusivamente agli utenti con il permesso amministrativo.

Versioni Utilizzate:

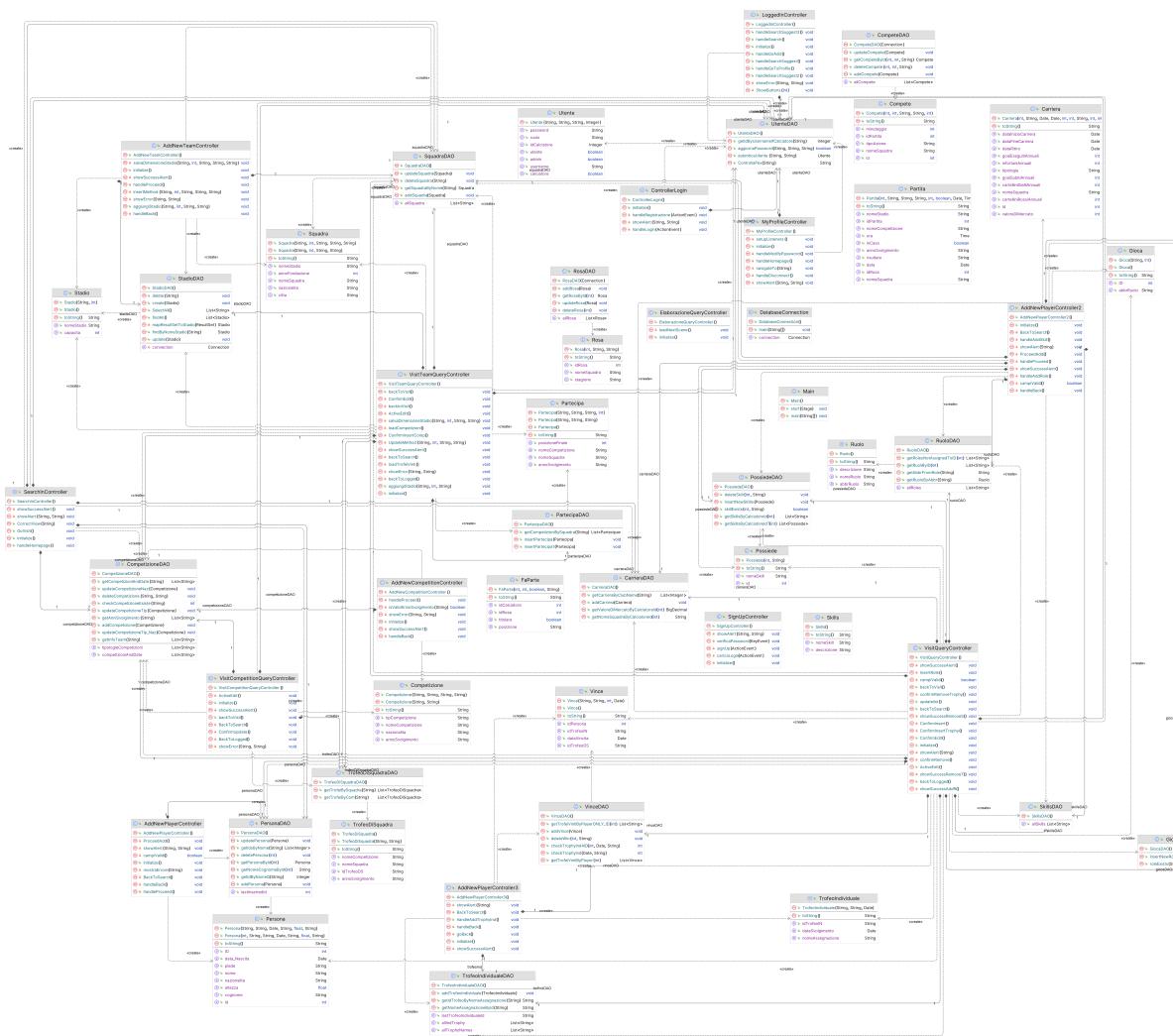
PostgreSQL v.16

JDK v.23

4.2 Diagramma del Problema



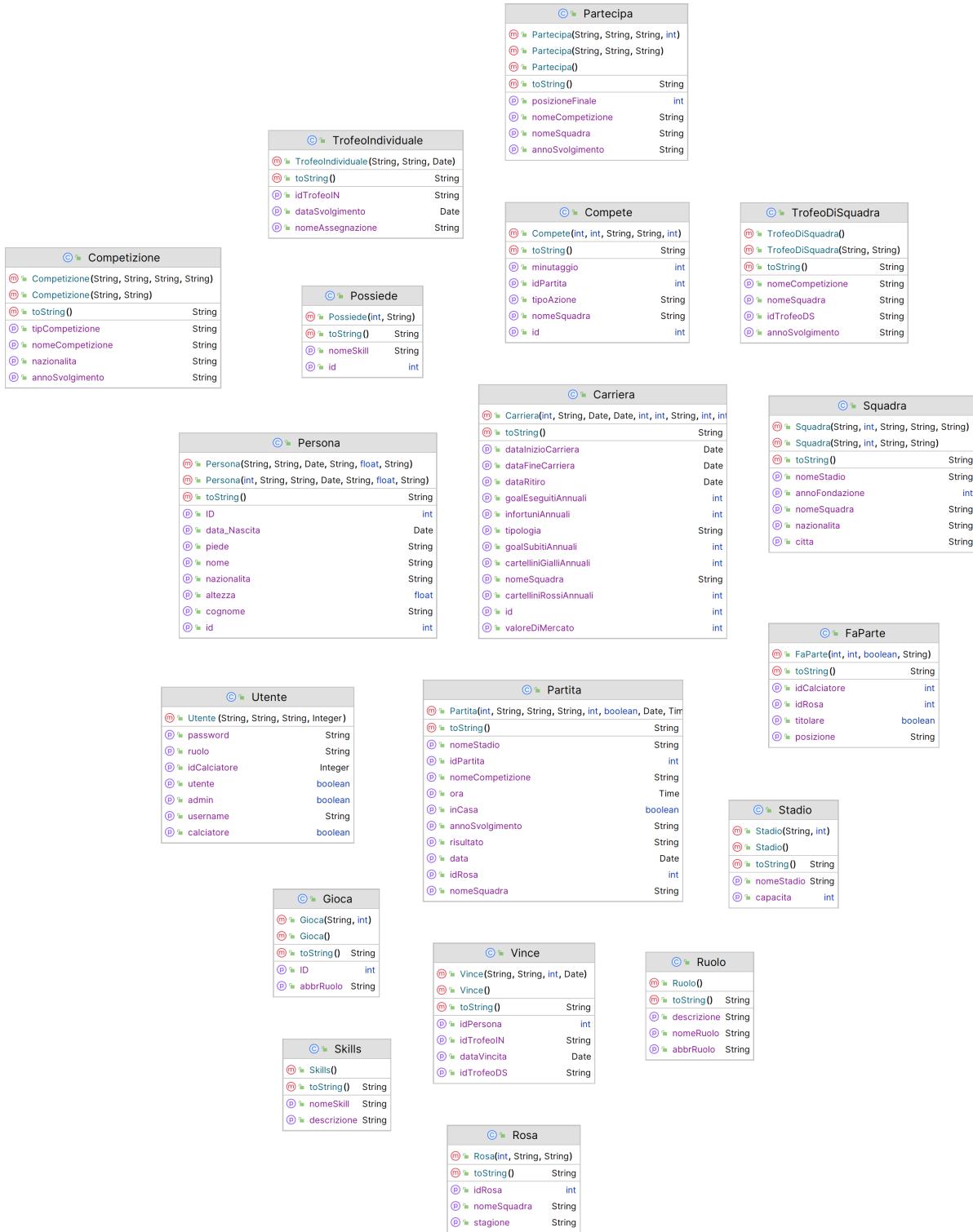
4.3 Diagramma della Soluzione



4.4 Controller Package

<code>VisitQueryController</code>	<code>VisitTeamQueryController</code>	<code>AddNewPlayerController2</code>	<code>LoggedInController</code>
<pre>(@ VisitQueryController() @ backToVisit() void @ updateList() void @ backToSearch() void @ ConfirmEdit() void @ initialize() void @ ActiveEdit() void @ backToLogged() void @ showSuccessAlert() void @ InsertRole() void @ campiValid() boolean @ confirmRemoveTrophy() void @ showSuccessRemoveS() void @ ConfirmInsert() void @ ConfirmInsertTrophy() void @ showAlert(String) void @ confirmRemove() void @ showSuccessRemoveT() void @ showSuccessAddR() void</pre>	<pre>(@ VisitTeamQueryController() @ backToVisit() void @ aggiungiStadio(String, int, String) void @ initialize() void @ ConfirmEdit() void @ backtoVisit() void @ ActiveEdit() void @ salvaDimensioneStadio(String, int, String, String) void @ loadCompetizioni() void @ ConfirmInsertComp() void @ UpdateMethod (String, int, String, String) void @ showSuccessAlert() void @ backToSearch() void @ loadTrofeiVinti() void @ showError(String, String) void @ backToLogged() void</pre>	<pre>(@ AddNewPlayerController2() @ initialize() void @ BackToSearch() void @ handleAddSkill() void @ showAlert (String) void @ ProceedAdd() void @ handleProceed() void @ showSuccessAlert() void @ handleAddRole() void @ campiValid() boolean @ handleBack() void</pre>	<pre>(@ LoggedInController() @ handleSearchSuggest3() void @ handleSearch() void @ initialize() void @ handleGoAdd() void @ handleSearchSuggest() void @ handleGoToProfile() void @ handleSearchSuggest2() void @ showError(String, String) void @ ShowButtons (int) void</pre>
<code>AddNewTeamController</code>	<code>VisitCompetitionQueryController</code>	<code>AddNewPlayerController</code>	<code>AddNewPlayerController3</code>
<pre>(@ AddNewTeamController() @ initialize() void @ aggiungiStadio(String, int, String, String) void @ showSuccessAlert() void @ handleProceed() void @ insertMethod (String, int, String, String, String) void @ showError(String, String) void @ salvaDimensioneStadio(String, int, String, String) void @ handleBack() void</pre>	<pre>(@ VisitCompetitionQueryController() @ initialize() void @ backToVisit() void @ showSuccessAlert() void @ ActiveEdit() void @ BackToSearch() void @ ConfirmUpdate() void @ BackToLogged() void @ showError(String, String) void</pre>	<pre>(@ AddNewPlayerController() @ initialize() void @ showAlert (String, String) void @ campiValid() boolean @ ProceedAdd() void @ mostraErrore(String) void @ BackToSearch() void @ handleBack() void @ handleProceed() void</pre>	<pre>(@ AddNewPlayerController3() @ handleAddTrophyInd() void @ initialize() void @ showAlert(String) void @ handleBack() void @ goBack() void @ BackToSearch() void @ handleProceed() void</pre>
<code>MyProfileController</code>	<code>AddNewCompetitionController</code>	<code>SearchInController</code>	<code>SignUpController</code>
<pre>(@ MyProfileController() @ initialize() void @ setupListeners () void @ handleModifyPassword() void @ handleHomepage() void @ navigateTo(String) void @ handleDisconnect() void @ showAlert (String, String) void</pre>	<pre>(@ AddNewCompetitionController() @ initialize () void @ isValidAnnoSvolgimento(String) boolean @ showError (String, String) void @ handleProceed() void @ showSuccessAlert() void @ handleBack() void</pre>	<pre>(@ SearchInController() @ CorrectView(String) void @ GoVisit() void @ initialize() void @ handleHomepage () void @ showSuccessAlert() void @ showAlert (String, String) void</pre>	<pre>(@ SignUpController() @ showAlert (String, String) void @ verificaPassword(KeyEvent) void @ signUp (ActionEvent) void @ caricaLogin(ActionEvent) void @ initialize() void</pre>
<code>ControllerLogin</code>	<code>ElaborazioneQueryController</code>		
<pre>(@ ControllerLogin() @ initialize() void @ handleRegistrazione (ActionEvent) void @ showAlert (String, String) void @ handleLogin(ActionEvent) void</pre>	<pre>(@ ElaborazioneQueryController() @ initialize() void @ loadNextScene() void</pre>		

4.5 Model Package

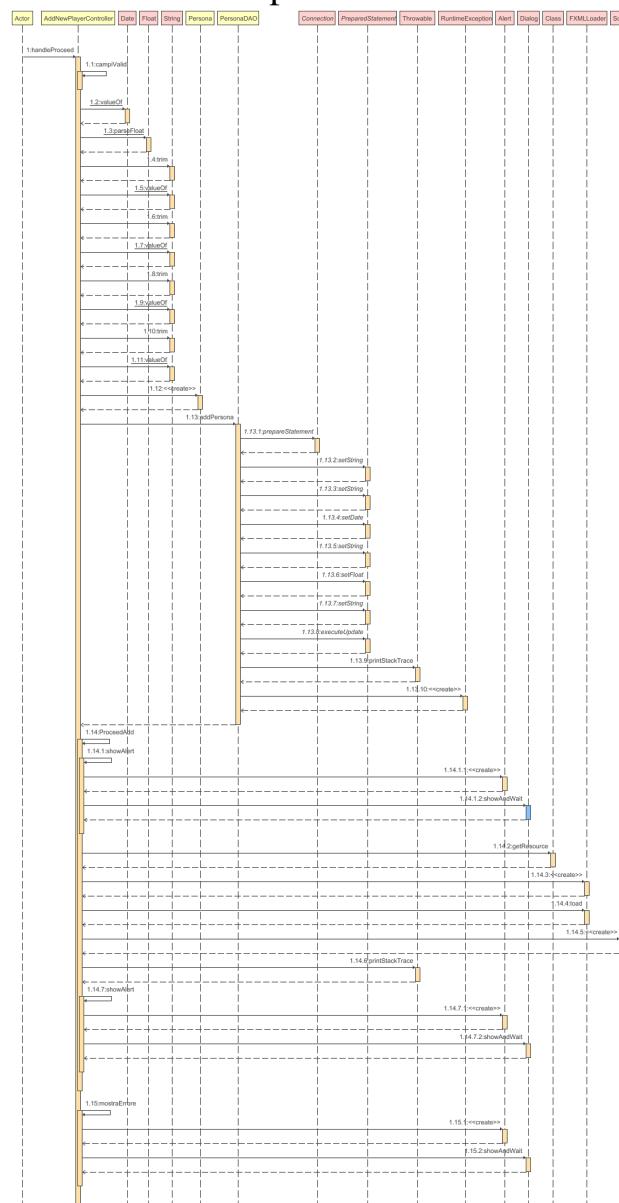


4.6 DAO Package



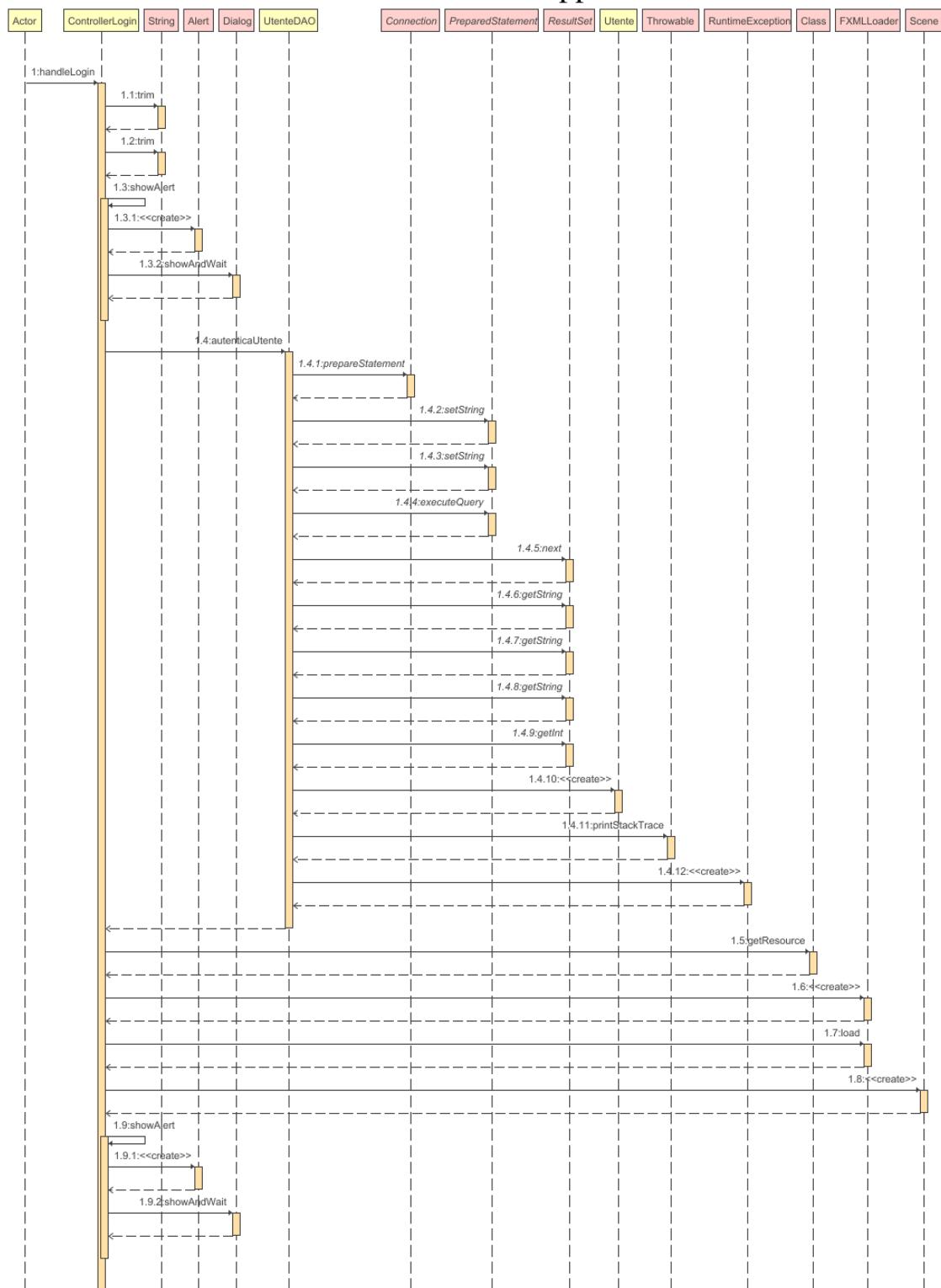
4.7 Sequence Diagram 1 - AddNewPlayer

Come prima funzionalità del sequence abbiamo deciso di interpretare la funzionalità di inserimento del nuovo calciatore, in particolare la prima parte, dove l'amministratore procede ad inserire le generalità del calciatore. A questo proposito, in questa interfaccia vengono chiamati e fatti intervenire diversi metodi e funzioni al fine di garantire un inserimento come previsto da schema database.



4.8 Sequence Diagram 2 - Login

Come seconda funzionalità parleremo del metodo che si preoccupa di gestire il login e controllare i dati inseriti una volta che si preme il pulsante per provare ad accedere all'interno dell'applicativo.



Ringraziamo per la cortese attenzione.

I membri del gruppo:

Roberto Rocco Milanese - N86004554

Valerio Monteforte - N86005290