Università degli Studi di Ferrara
Dipartimento di Ingegneria

Corso di Laurea Triennale in Ingegneria Elettronica e Informatica

Documentazione Basi di Dati

Progetto:
Gestione Attività Sportive

Studente:
Michele Vaccari
Matricola 121955

Docente:
Prof. Denis Ferraretti

Anno Accademico 2016–2017

INDICE

ELENCO DELLE FIGURE

1 SPECIFICHE PROGETTO

1.1 CARATTERISTICHE DEL SISTEMA

Il sito *SportFerrara* è una piattaforma per la gestione delle attività sportive attraverso una comunità di utenti.

L'applicazione web ha le seguenti caratteristiche:

1. Possibilità di creare gli utenti per i due ruoli principali: arbitri e gestori di squadra;
2. Possibilità di creare i diversi tornei, di disegnarne lo schema. Ogni torneo ha: nome, descrizione, sponsor e tipo (all'italiana o ad eliminazione).
3. Un torneo è costituito da diversi gironi o fasi, in base alla tipologia di torneo. Un girone ha un nome (ad esempio: girone 1) ed è costituito da un insieme di gare. Una fase ha un nome (ad esempio: semifinale 1) ed è costituita da una sola gara;
4. Una gara è descritta da: data e ora, luogo, nomi delle squadre che si sfidano, uno o più arbitri;
5. Per ogni torneo è associata una classifica generale che viene aggiornata sulla base dei referti di gara compilati dagli arbitri;
6. Possibilità per il gestore di una squadra di creare una squadra con una rosa di massimo 36 giocatori. Ogni squadra ha un nome e può avere uno sponsor. Per ogni giocatore il gestore può specificare: nome, cognome, luogo e data di nascita, numero di maglia e foto;
7. Per ogni gara cui la squadra è assegnata, il gestore della squadra deve fornire la formazione della squadra composta da nome e cognome dei giocatori e rispettivi ruoli;
8. A gara terminata l'arbitro dovrà compilare un referto in cui annoterà: l'orario effettivo di inizio e di fine della gara, risultato finale, numero di reti con i rispettivi giocatori che li hanno realizzati, i giocatori espulsi per ogni squadra e i giocatori ammoniti per ogni squadra;
9. Possibilità di selezionare il torneo desiderato e visualizzare:
    - La pagina relativa ad una gara con: nomi delle squadre, formazioni e referti;
    - La pagina relativa ad una squadra con la rosa dei giocatori e il calendario delle partite;

- La pagina relativa ad un giocatore con le statistiche di gioco (punti realizzati, espulsioni e ammonizioni) per quel torneo.

1.2 UTENTI DEL SISTEMA

Il sistema prevede che le categorie di utenti sia così rappresentata:

AMMINISTRATORI Possono effettuare i punti dal 1 al 5 compresi.

GESTORI DI SQUADRA Possono effettuare i punti dal 6 al 7 compresi.

ARBITRI Possono effettuare solamente il punto 8.

UTENTI PUBBLICI Possono effettuare solamente il punto 9.

2 DIAGRAMMA E-R

Una volta analizzate le specifiche del progetto per il quale si vuole realizzare il database, si identificano e si descrivono le singole entità e le loro associazioni.

2.1 DESCRIZIONE DELLE ENTITÀ

2.1.1 Utente

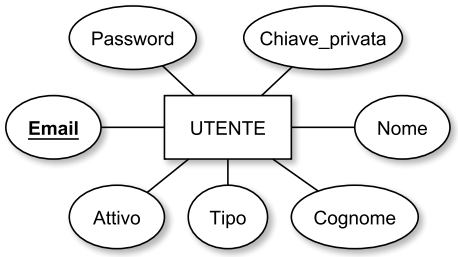

Figura 1: Entità Utente

Tale entità rappresenta un generico utente dell'applicazione. L'utente si distingue in *Registrato* e *Amministratore*.

Descrizione Attributi

EMAIL Chiave primaria dell'entità "Utente" che identifica univocamente un utente. Il valore di tale attributo viene inserito dall'utente con relativo controllo di omonimie.

PASSWORD Identifica la password utilizzata dall'utente in fase di autenticazione. Il valore di tale attributo viene generato cifrando in fase di registrazione, mediante la funzione crittografica MD5, la password inserita dall'utente.

CHIAVE PRIVATA Stringa alfanumerica con lunghezza di 32 caratteri. Il valore di tale attributo viene generato cifrando, mediante la funzione crittografica MD5, la data di registrazione dell'utente, fornita dal server, al momento della registrazione.

NOME Identifica il nome dell'utente con un massimo di 30 caratteri.

COGNOME Identifica il cognome dell'utente con un massimo di 30 caratteri.

TIPO Identifica il tipo di utente ovvero:

- "A" per identificare un amministratore;
- "G" per identificare il gestore di una squadra;
- "R" per identificare un arbitro.

ATTIVO Identifica lo stato dell'utente., ovvero se è attivo oppure no. Può assumere due valori di tipo booleano (true o false); di default assume valore true (attivo).

N.B.

Nessuno di questi attributi può assumere il valore NULL.

Caso Particolare

Esiste sempre un valore particolare, ovvero "admin", che rappresenta sempre il primo utente Amministratore. Tale valore è sempre presente all'interno della tabella.

2.1.2 Amministratore

Figura 2: Entità Amministratore

Tale entità rappresenta la specializzazione dell'entità *Utente*, ereditandone tutti gli attributi. Serve per rappresentare gli utenti *Amministratori*.

2.1.3 Registrato

Figura 3: Entità Registrato

Tale entità rappresenta la specializzazione dell'entità *Utente*, ereditandone tutti gli attributi. Serve per rappresentare gli utenti *Gestori Squadra* e gli utenti *Arbitri* che possono accedere all'applicazione Web. Tale entità può essere creata solo dall'amministratore.

Descrizione Attributi

INDIRIZZO Identifica l'indirizzo dell'utente con un massimo di 50 caratteri.

TELEFONO Identifica il telefono dell'utente con un massimo di 15 caratteri.

N.B.

Nessuno di questi attributi può assumere il valore NULL.

2.1.4 Arbitro

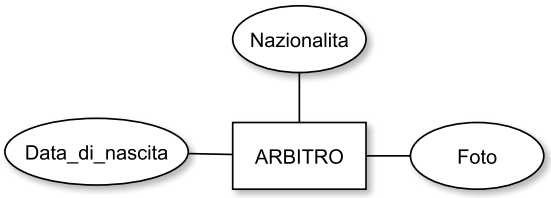

Figura 4: Entità Arbitro

Tale entità rappresenta la specializzazione dell'entità *Registrato*, ereditandone tutti gli attributi. Serve per rappresentare il tipo di utente *Arbitro*. Tale entità può essere creata solo dall'amministratore.

Descrizione Attributi

DATA DI NASCITA Identifica la data di nascita dell'utente.

NAZIONALITA Identifica la nazionalità dell'utente con un massimo di 50 caratteri.

FOTO Identifica il percorso dell'immagine all'interno della cartella *Immagini* del server con un massimo di 50 caratteri.

N.B.

Nessuno di questi attributi può assumere il valore NULL.

2.1.5 Gestore Squadra

Figura 5: Entità Gestore Squadra

Tale entità rappresenta la specializzazione dell'entità *Registrato*, ereditandone tutti gli attributi. Serve per rappresentare il tipo di utente *Gestore Squadra*. Tale entità può essere creata solo dall'amministratore.

2.1.6 Torneo

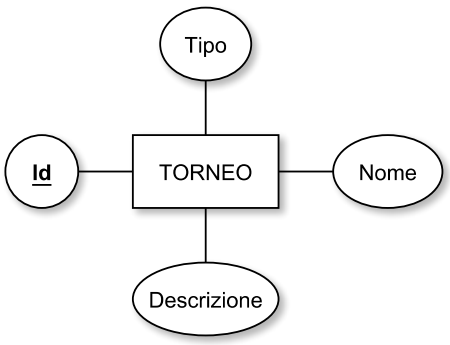

Figura 6: Entità Torneo

Tale entità rappresenta un torneo in svolgimento o già svolto. Può essere di due tipi: ad eliminazione diretta o all'italiana. Tale entità può essere creata solo dall'amministratore.

Descrizione Attributi

ID Chiave primaria dell'entità "Torneo" che identifica univocamente un torneo. Il valore di tale attributo viene assegnato automaticamente dal sistema.

TIPO Identifica la tipologia di torneo ovvero:

- "E" per identificare un torneo ad eliminazione diretta;
- "I" per identificare un torneo all'italiana;

NOME Identifica la nazionalità dell'utente con un massimo di 30 caratteri.

DESCRIZIONE Campo utilizzato dall'amministratore per inserire delle informazioni aggiuntive relative al torneo (storia, luogo, sponsor, ecc.).

N.B.

Ad eccezione del campo "Descrizione", nessuno di questi attributi può assumere il valore NULL.

2.1.7 Partita

Tale entità rappresenta una partita. Tale entità può essere creata solo dall'amministratore.

Descrizione Attributi

ID Chiave primaria dell'entità "Partita" che identifica univocamente una partita. Il valore di tale attributo viene assegnato automaticamente dal sistema.

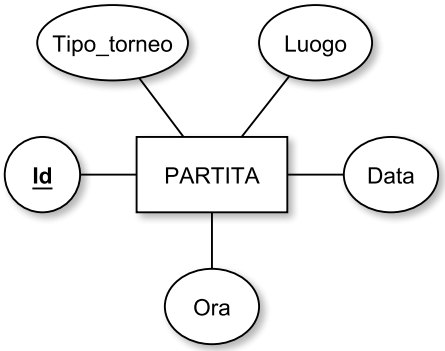

Figura 7: Entità Partita

TIPO TORNEO Identifica la tipologia di torneo ovvero:

- "E" per identificare un torneo ad eliminazione diretta;
- "I" per identificare un torneo all'italiana;

Questo attributo serve per gestire in modi diversi le classifiche.

LUOGO Identifica il luogo in cui è si svolge la partita con un massimo di 30 caratteri.

DATA Identifica la data in cui è si svolge la partita.

ORA Identifica l'ora in cui si svolge la partita.

N.B.

Nessuno di questi attributi può assumere il valore NULL.

2.1.8 Referto

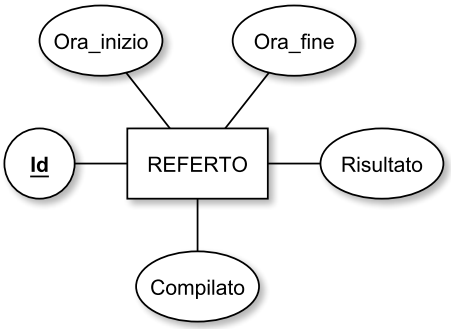

Figura 8: Entità Referto

Tale entità rappresenta il referto di una determinata gara. Tale entità può essere creata solo dall'arbitro.

Descrizione Attributi

ID Chiave primaria dell'entità "Referto" che identifica univocamente una partita. Il valore di tale attributo viene assegnato automaticamente dal sistema.

ORA INIZIO Identifica l'orario effettivo di inizio della partita.

ORA FINE Identifica l'orario effettivo di fine della partita.

RISULTATO Identifica il risultato finale dell'incontro.

COMPILATO Identifica lo stato del referto, ovvero se è stato compilato oppure no. Può assumere due valori di tipo booleano (true o false).

N.B.

Nessuno di questi attributi può assumere il valore NULL.

2.1.9 Squadra

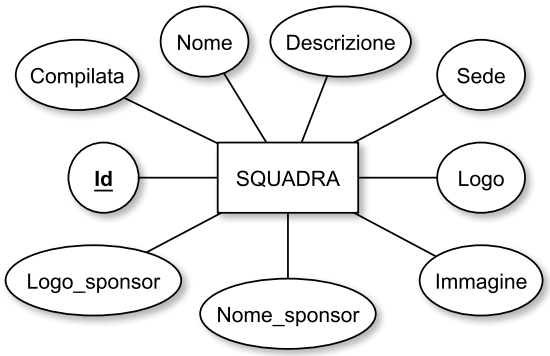

Figura 9: Entità Squadra

Tale entità rappresenta una squadra. Tale entità viene creata dall'amministratore e successivamente viene completata dal gestore della squadra.

Descrizione Attributi

ID Chiave primaria dell'entità "Squadra" che identifica univocamente una partita. Il valore di tale attributo viene assegnato automaticamente dal sistema.

COMPILATA Identifica lo stato della squadra, ovvero se è stata compilata oppure no. Può assumere due valori di tipo booleano (true o false). Se la squadra non è stata compilata, la prima volta che si accede all'applicazione web si dovranno fornire i dati richiesti. Se non si forniscono i dati della squadra non si può svolgere nessuna attività.

NOME Identifica il nome legale della società con un massimo di 30 caratteri.

DESCRIZIONE Campo utilizzato dall gestore della squadra per inserire delle informazioni aggiuntive relative alla squadra.

SEDE Identifica la sede legale della società con un massimo di 30 caratteri.

LOGO Identifica il percorso del logo della squadra all'interno della cartella *Immagini* del server con un massimo di 50 caratteri.

IMMAGINE Identifica il percorso della foto della squadra all'interno della cartella *Immagini* del server con un massimo di 50 caratteri.

NOME SPONSOR Identifica il nome dello sponsor ufficiale della squadra con un massimo di 30 caratteri.

LOGO SPONSOR Identifica il percorso del logo dello sponsor all'interno della cartella *Immagini* del server con un massimo di 50 caratteri.

N.B.

Nessuno di questi attributi può assumere il valore NULL.

2.1.10 Classifica

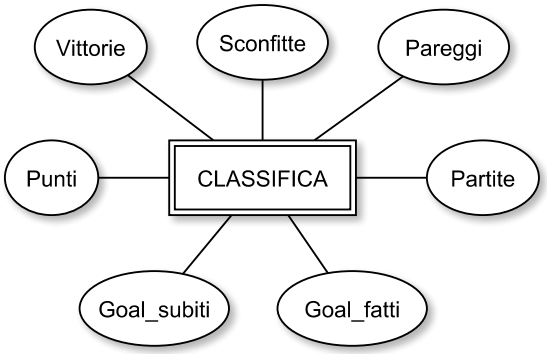

Figura 10: Entità Classifica

Tale entità rappresenta la classifica di un determinato torneo. Tale entità è un'entità debole in quanto non ha propri attributi chiave ma è un torneo ad identificarla univocamente.

Descrizione Attributi

PUNTI Identifica i punti totalizzati fino a quel momento da una determinata squadra in un determinato torneo.

VITTORIE Identifica il numero totale delle partite vinte da una squadra in un determinato torneo.

SCONFITTE Identifica il numero totale delle partite perse da una squadra in un determinato torneo.

PAREGGI Identifica il numero totale delle partite pareggiate da una squadra in un determinato torneo.

PARTITE Identifica il numero totale delle partite disputate da una squadra in un determinato torneo.

GOAL FATTI Identifica il totale dei goal fatti dai giocatori di una determinata squadra in un torneo.

GOAL SUBITI Identifica il totale dei goal subiti dai giocatori una determinata squadra in un torneo.

N.B.

Nessuno di questi attributi può assumere il valore NULL.

2.1.11 Giocatore

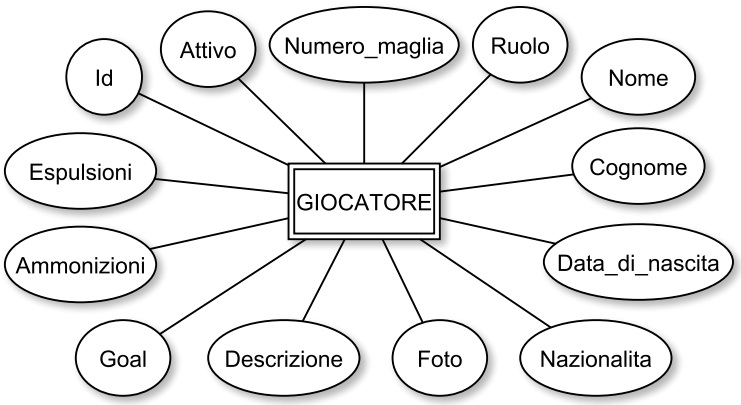

Figura 11: Entità Giocatore

Tale entità rappresenta la un determinato giocatore di una squadra. Tale entità è un'entità debole in quanto non ha propri attributi chiave ma è una squadra ad identificarla univocamente.

Descrizione Attributi

ID Chiave parziale dell'entità "Giocatore" che identifica univocamente un giocatore appartenente ad una determinata squadra. Il valore di tale attributo viene assegnato automaticamente dal sistema.

ATTIVO Identifica lo stato del giocatore, ovvero se è attivo oppure no. Può assumere due valori di tipo booleano (true o false); di default assume valore true (attivo).

NUMERO MAGLIA Identifica il numero maglia con cui il giocatore prende parte alle partite.

RUOLO Identifica il ruolo che il giocatore interpreta in campo con un massimo di 50 caratteri.

NOME Identifica il nome del giocatore con un massimo di 30 caratteri.

COGNOME Identifica il cognome del giocatore con un massimo di 30 caratteri.

DATA DI NASCITA Identifica la data di nascita del giocatore.

NAZIONALITA Identifica il nome del giocatore con un massimo di 20 caratteri.

FOTO Identifica il percorso della foto del giocatore all'interno della cartella *Immagini* del server con un massimo di 50 caratteri.

DESCRIZIONE Campo utilizzato dall gestore della squadra per inserire delle informazioni aggiuntive relative al giocatore.

GOAL Indica il numero di goal effettuati dal giocatore.

AMMONIZIONI Indica il numero di ammonizioni del giocatore.

ESPULSIONI Indica il numero di espulsioni del giocatore.

N.B.

Nessuno di questi attributi può assumere il valore NULL.

2.2 DESCRIZIONE DELLE ASSOCIAZIONI

2.2.1 Ha

Associazione presente tra Torneo e Classifica. L'entità classifica è debole per questo è all'interno di un doppio rettangolo; anche il rombo è doppio per lo stesso motivo. Un entità è debole se non ha attributi chiave; tali entità hanno sempre un vincolo di partecipazione totale perché altrimenti non sarebbero identificabili. Come si può notare è presente un vincolo di partecipazione totale dalla parte di classifica. La molteplicità è 1 : N in quanto ad un torneo sono associate più classifiche (una per ogni squadra), ma una classifica può essere associata soltanto ad un singolo torneo.

2.2.2 Presente

Associazione presente tra Squadra e Classifica; Valgono le stesse considerazioni fatte per la relazione "Ha". La molteplicità è 1 : N in quanto ad una

squadra possono essere associate più classifiche (una per ogni torneo a cui la squadra partecipa), ma una classifica può essere associata soltanto ad una squadra.

2.2.3 Composta

Associazione presente tra Squadra e Giocatore. Indica quali sono i giocatori che compongono una determinata squadra. In una squadra possono prendere parte al massimo 36 giocatori. I giocatori possono far parte di una sola squadra. Come si può vedere la molteplicità è 1 : 36. I 36 giocatori sono la rosa della squadra.

2.2.4 Gestisce

Associazione presente tra Gestore Squadra e Squadra. Utile per sapere da chi è gestita una determinata squadra; è presente un vincolo di partecipazione totale dalla parte di squadra in quanto non può esistere una squadra se non esiste un gestore che la crea. La molteplicità è 1 : 1 in quanto una squadra può essere gestita da un solo gestore e viceversa.

2.2.5 Riferito

Associazione presente tra Partita e Referto. Serve per associare un referto ad una determinata partita che, deve essere o è già stata, giocata. È presente un vincolo di partecipazione totale dalla parte di referto in quanto non esiste un referto se non esiste una partita. La molteplicità è 1 : 1 poiché ad una determinata partita è associato un solo referto e viceversa.

2.2.6 Gestisce Registrato

Associazione presente tra Registrato e Amministratore. I Registrati del sistema (Arbitri e Gestori squadra) devo essere, creati da uno degli amministratori. Per questo motivo è presente un vincolo di partecipazione totale dalla parte di Registrato. Un Registrato non può esistere se non è presente un Amministratore che lo abbia creato. La molteplicità è 1 : N in quanto un amministratore può creare più utenti Registrati, ma un utente Registato può essere creato da un solo amministratore.

2.2.7 Gestisce Torneo

Associazione presente tra Amministratore e Torneo. L'amministratore è l'unico a poter creare un torneo (ad Eliminazione diretta o all'Italiana). Per questo motivo è presente un vincolo di partecipazione totale dalla parte di Torneo. Un torneo non può esistere se non c'è un Amministratore che lo

abbia creato. La molteplicità è 1 : N in quanto un amministratore può creare molti tornei, ma un torneo può essere creato da un solo amministratore.

2.2.8 Compila

Associazione presente tra Arbitro e Referto. L'Arbitro, prima scelto da un amministratore, sarà colui che "scrive" il referto di una specifica partita. È presente un vincolo di partecipazione totale dalla parte di referto, in quanto questo non può esistere se non è esiste una arbitro che può compilare tale referto. La molteplicità è 1 : N in quanto un arbitro può compilare N referti ma un solo referto può essere compilato da un solo Arbitro.

2.2.9 Partecipa

Associazione presente tra Torneo e Squadra. Indica a quali tornei partecipano le squadre e quali squadre partecipano ad un determinato torneo. La partecipazione totale è dalla parte di torneo perché non può esistere se non sono presenti delle squadre che vi possano partecipare. La molteplicità è M : N in quanto una squadra può partecipare a più tornei ed un torneo può avere più squadre che vi partecipano, con un minimo di 2 per un torneo ad eliminazione diretta e con un minimo di 4 per un torneo all'italiana.

2.2.10 Cartellino

Associazione presente tra Arbitro e Giocatore. Indica chi sono i giocatori che hanno avuto un ammonizione o un'espulsione dall'arbitro. Tale associazione presenta l'attributo Numero che specifica il numero di maglia del giocatore ammonito? La partecipazione parziale in entrambe le parti perchè può esistere un arbitro che non ha mai assegnato un ammonizione o un'espulsione e, viceversa, giocatore con nessuna ammonizione o espulsione da parte di un alrbitro. La molteplicità è M : N in quanto un arbitro può ammonire ed epellere più di un giocatore e viceversa.

2.2.11 Marcatore

Associazione presente tra Referto e Giocatore. Indica chi sono i giocatori che hanno segnato un goal in un referto. La partecipazione parziale in entrambe le parti perchè può esistere un referto con nessun giocatore che ha segnato un goal. La molteplicità è M : N in quanto in un referto possono esserci più giocatori che segnano un goal e, viceversa, più giocatore possono segnare più goal.

2.2.12 Scelto

Associazione ternaria presente tra Arbitro, Amministratore e Partita. Indica che un determinato arbitro è stato scelto da un determinato amministratore per una specifica partita. Un arbitro può essere scelto per N diverse partite e per ogni partita è scelto un singolo arbitro. Un amministratore può scegliere N arbitri per N diverse partite il ciò vuol dire che un arbitro arbitra una singola partita.

2.2.13 Formazione

Associazione ternaria presente tra Squadra, Giocatore e Referto. Serve per sapere chi sono i giocatori titolari più riserve che faranno parte alla formazione di una determinata partita. Nell'associazione è presente un attributo (Riserva) che indica proprio le riserve della partita. Se questo è uguale a "true" allora il giocatore scelto sarà una riserva, in caso sia "false" il giocatore sarà un titolare. Per ogni referto sono fornite due formazioni, una per ogni squadra, composte da 18 giocatori (11 titolari e 7 riserve).

2.2.14 Giocano

Associazione ternaria presente tra Partita, Torneo e Squadra. Indica quali squadre di un determinato torneo si devono affrontare. Tale associazione presenta l'attributo Nome_partita che specifica il nome della fase o il nome della giornata dei tornei rispettivamente ad eliminazione diretta o all'italiana. Una partita viene giocata sempre da 2 squadre e fa parte di un singolo torneo, in un torneo sono possibili N partite. C'è un vincolo di partecipazione totale dalla parte di partita in quanto essa non può esistere se non è stato creato un torneo o sono presenti delle squadre.

2.3 IL MODELLO E-R COMPLETO

Per il modello E-R completo si veda la figura 12 nella pagina seguente.

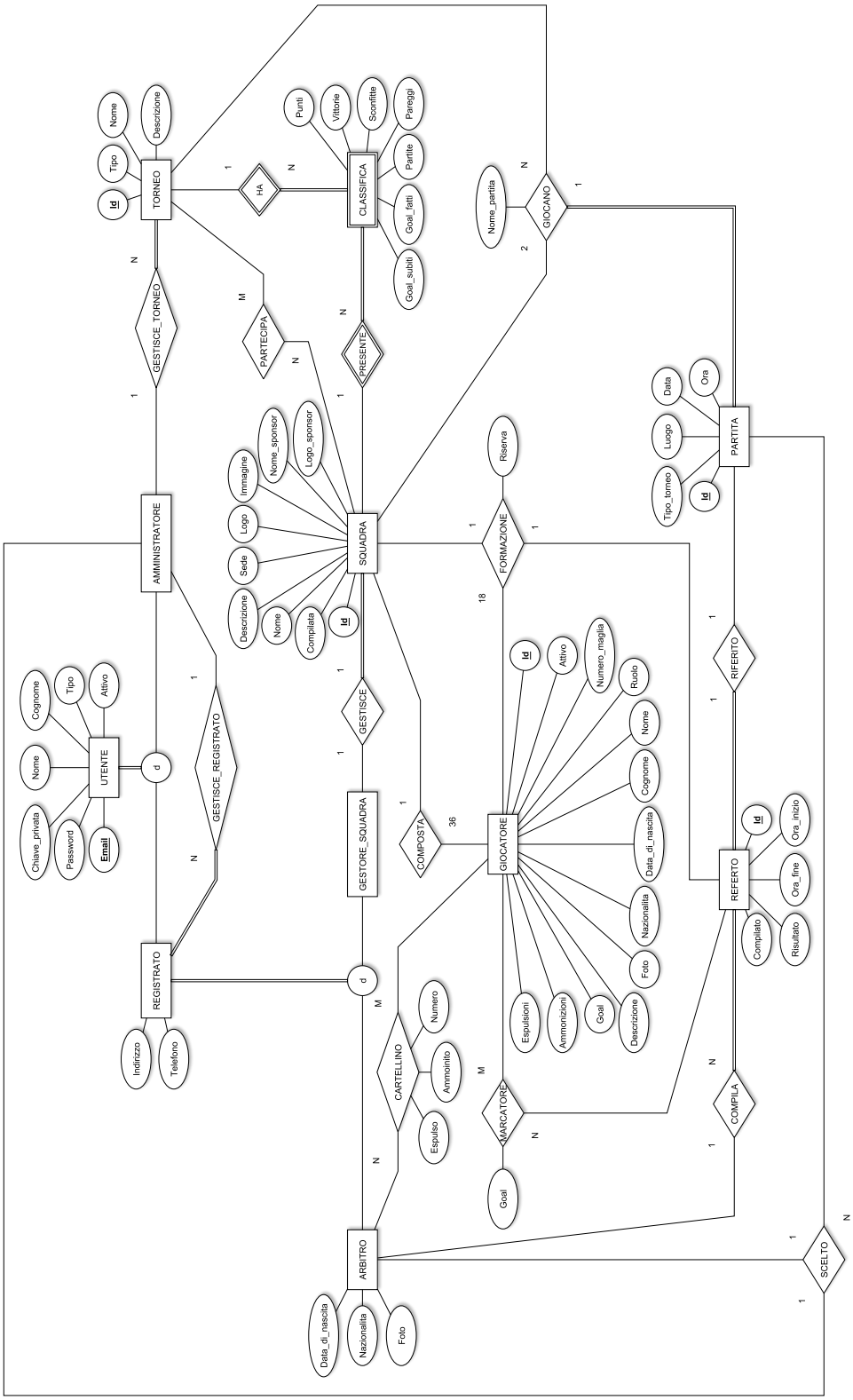

Figura 12: Schema E-R completo

3 MODELLO RELAZIONALE

3.1 L'ALGORITMO DI TRADUZIONE

La fase successiva della progettazione di una base di dati consiste nel passare dalla progettazione concettuale alla progettazione logica. Per passare al modello relazionale, si applica un semplice algoritmo.

L'algoritmo che si utilizza per la traduzione dal modello E-R in modello relazionale è formato dalle seguenti operazioni:

1. Traduzione di tipi di entità;
2. Traduzione della specializzazione;
3. Traduzione di tipi di entità deboli;
4. Traduzione di associazioni binarie di tipo 1 : 1;
5. Traduzione di associazioni binarie di tipo 1 : N;
6. Traduzione di associazioni binarie di tipo N : M;
7. Traduzione di associazioni N-arie;

3.1.1 Traduzione di entità

Per ogni tipo di entità (forte) nello schema E-R si costruisce una relazione che contiene tutti gli attributi semplici dell'entità.

3.1.2 Traduzione della specializzazione

Per ogni tipo di specializzazione dello schema E-R con tipo di entità generale, si costruisce una relazione e si inseriscono tutti gli attributi semplici dell'entità specializzata come attributi della relazione. Si inseriscono come attributi di chiave esterna della relazione gli attributi di chiave primaria delle relazioni corrispondenti ai tipi di entità generali. La chiave primaria della relazione è data dalla combinazione delle chiavi primarie delle entità generali e dalla chiave parziale del tipo di entità specializzata (se esiste).

3.1.3 Traduzione di entità deboli

Per ogni tipo di entità debole dello schema E-R con tipo di entità proprietario, si costruisce una relazione e si inseriscono tutti gli attributi semplici dell'entità debole come attributi della relazione. Si inseriscono come attributi di

chiave esterna della relazione gli attributi di chiave primaria delle relazioni corrispondenti ai tipi di entità proprietari. La chiave primaria della relazione è data dalla combinazione delle chiavi primarie delle entità proprietarie e dalla chiave parziale del tipo di entità debole (se esiste).

3.1.4 Traduzione di associazioni binarie 1:1

Per ogni tipo di associazione binaria 1 : 1 nello schema E-R si individuano le due relazioni coinvolte dall'associazione. Per la traduzione si usa l'approccio basato su chiavi esterne. Si sceglie una delle due relazioni coinvolte, preferibilmente la relazione corrispondente a un tipo di entità con vincolo di partecipazione totale, e si inserisce come chiave esterna la chiave primaria della seconda relazione. Infine si inseriscono sulla relazione scelta tutti gli attributi semplici del tipo di associazione 1 : 1.

3.1.5 Traduzione di associazioni binarie 1:N

Per ogni tipo di associazione binaria 1 : N nello schema E-R si individuano le due relazioni coinvolte dall'associazione. Per la traduzione si individuata la relazione che rappresenta il tipo di entità partecipante lato-N del tipo di associazione e si inserisce come chiave esterna la chiave primaria della relazione che rappresenta l'altro tipo di entità partecipante all'associazione. Infine si inseriscono sulla relazione scelta tutti gli attributi semplici del tipo di associazione 1 : N.

3.1.6 Traduzione di associazioni binarie N:M

Per ogni tipo di associazione binaria M : N nello schema E-R si costruisce una nuova relazione che rappresenta l'associazione. Si inseriscono come attributi di chiave esterna della relazione le chiavi primarie delle relazioni che rappresentano i tipi di entità partecipanti, le loro combinazioni formano la chiave primaria della relazione. Infine si inseriscono nella nuova relazione tutti gli attributi semplici del tipo di associazione M : N.

3.1.7 Traduzione di associazioni N-arie

L'ultimo passo dell'algoritmo si occupa dell'analisi e conversione dei tipi di associazioni N-arie. In questo caso si costruisce una nuova relazione in cui verranno inserite le chiavi primarie delle entità coinvolte; tali campi verranno considerati come chiavi esterne.

3.2 APPLICAZIONE DELL'ALGORITMO DI TRADUZIONE

Di seguito si riporta la traduzione in modello relazionale del modello E-R analizzato:

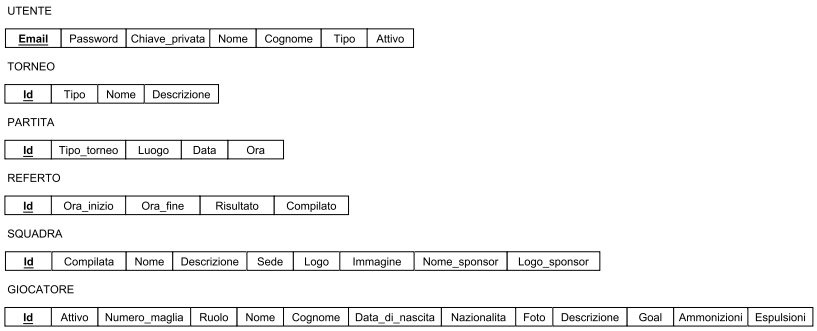

Figura 13: Traduzione di entità

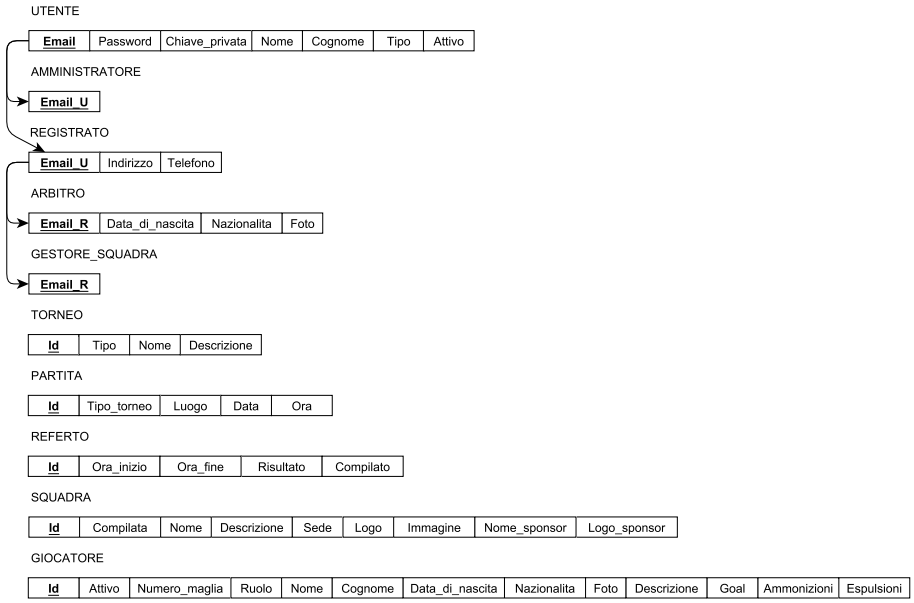

Figura 14: Traduzione di entità specializzate

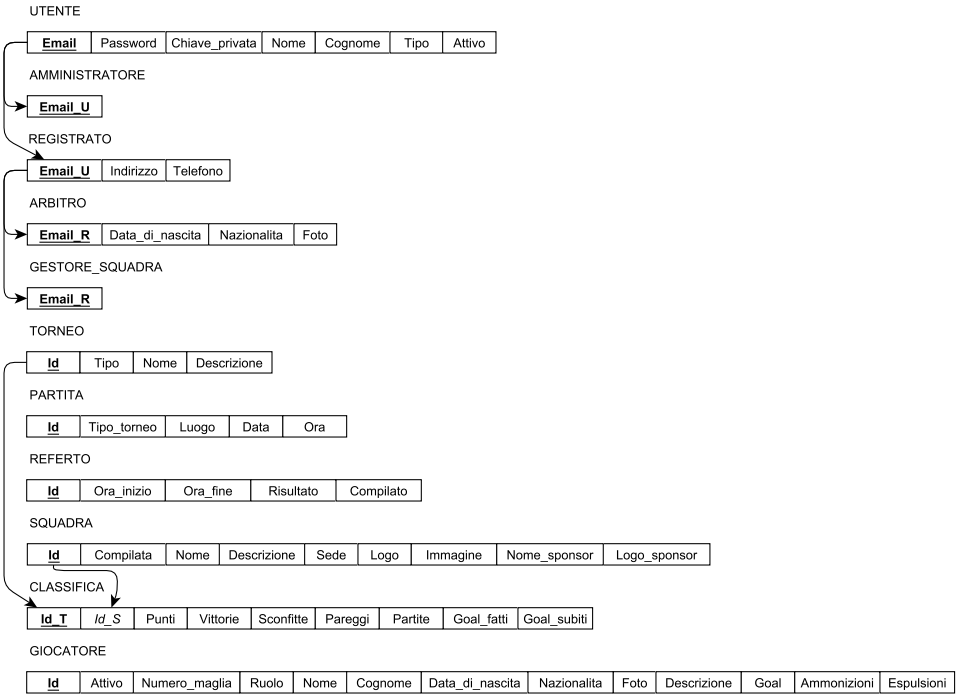

Figura 15: Traduzione di entità deboli

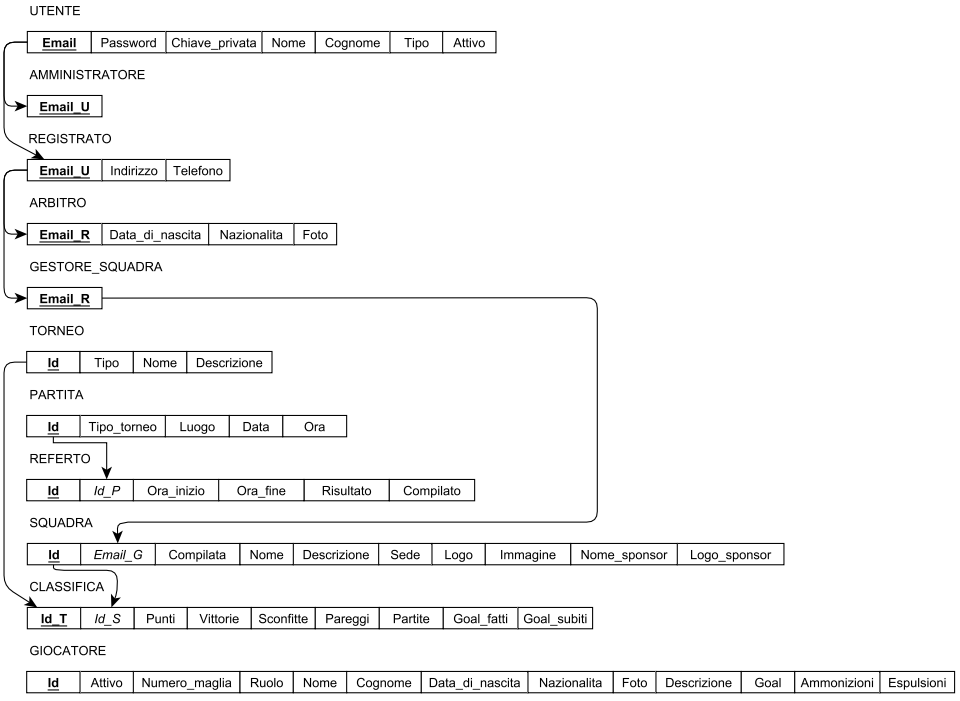

Figura 16: Traduzione di associazioni binarie 1:1

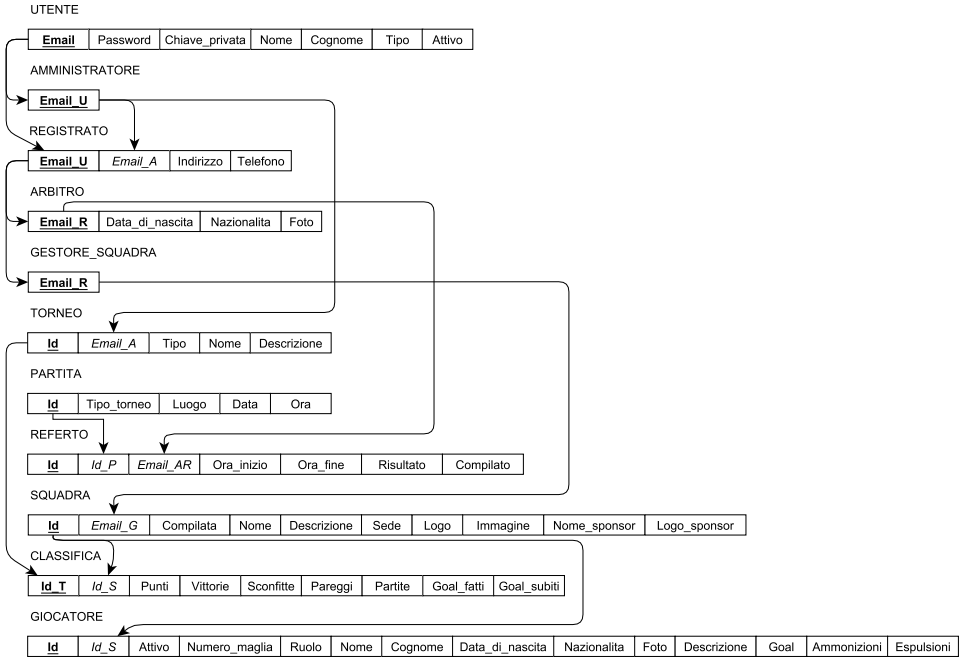

Figura 17: Traduzione di associazioni binarie 1:N

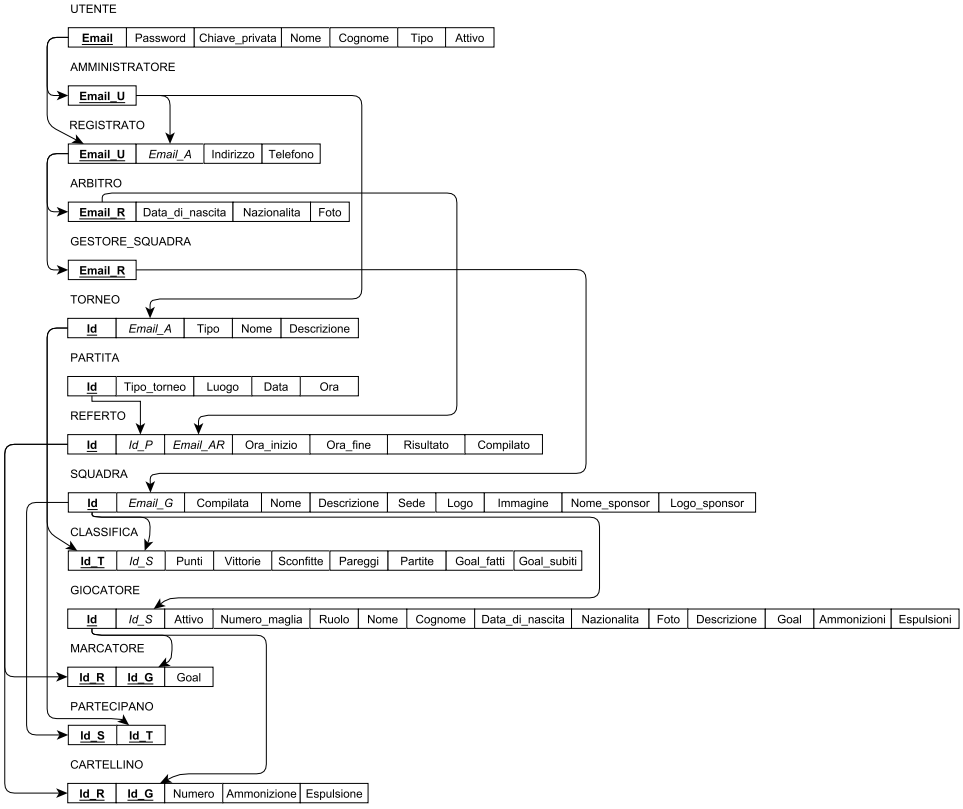

Figura 18: Traduzione di associazioni binarie M:N

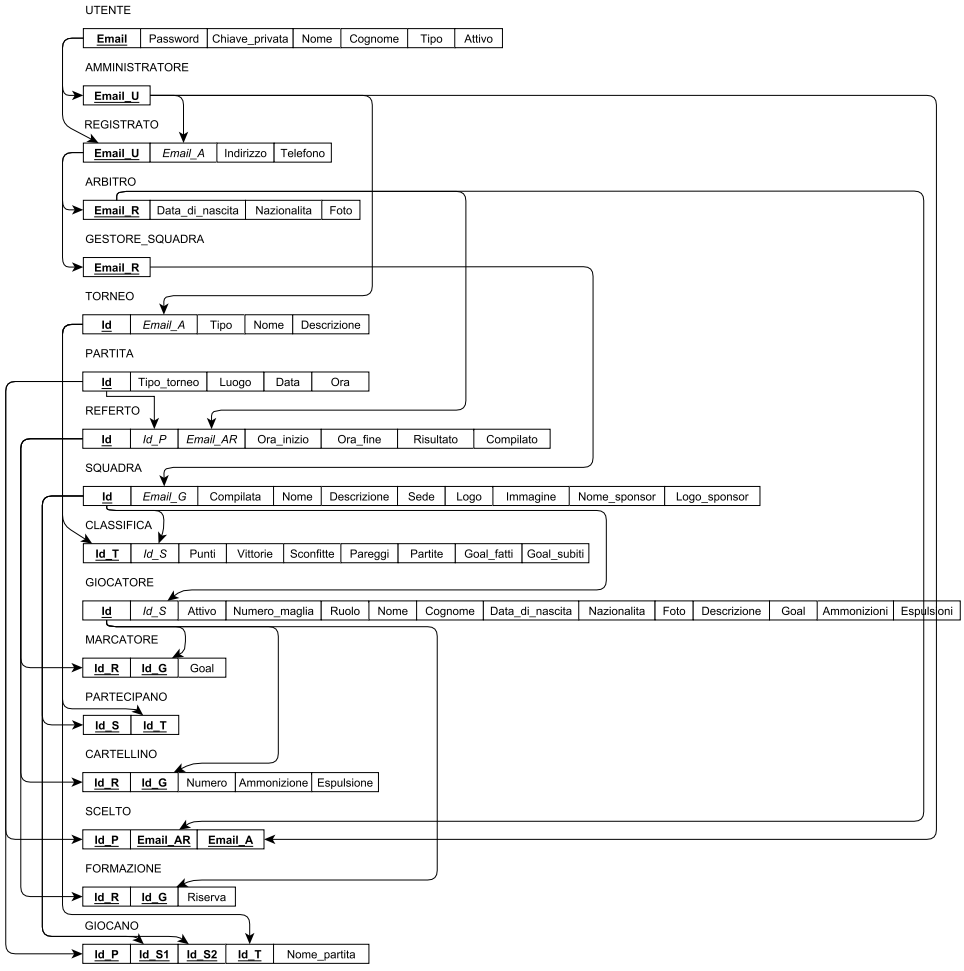

Figura 19: Traduzione di associazioni ternarie

4 NORMALIZZAZIONE

4.1 CENNI TEORICI

Per normalizzazione si intende il procedimento che ha lo scopo di eliminare la ridondanza e le eventuali incoerenze del database. Il processo di normalizzazione sottopone uno schema relazionale a una serie di test per certificare se soddisfa una data forma normale.

4.1.1 Prima forma normale (1NF)

Si dice che uno schema relazionale è in prima forma normale se il dominio di un attributo comprende valori atomici e se il valore di un qualsiasi attributo in una tupla sia un singolo valore del dominio, ovvero:

- Tutte le righe della tabella contengono lo stesso numero di colonne;
- Gli attributi rappresentano informazioni elementari;
- I valori che compaiono in una colonna sono dello stesso tipo, cioè appartengono allo stesso dominio;
- Ogni riga è diversa da tutte le altre, cioè non ce ne possono essere due con gli stessi valori nelle colonne;
- L'ordine con il quale le righe compaiono nella tabella è irrilevante.

La prima forma normale (1NF) è già parte integrante della definizione formale di relazione nel modello relazionale.

4.1.2 Seconda forma normale (2NF)

Si dice che uno schema relazionale è in seconda forma normale (2NF) quando è in prima forma normale (1NF) e tutti i suoi attributi non-chiave dipendono dall'intera chiave, cioè non possiede attributi che dipendono soltanto da una parte della chiave.

La seconda forma normale elimina la dipendenza parziale degli attributi dalla chiave e riguarda il caso di relazioni con chiavi composte, cioè formate da più attributi.

4.1.3 Terza forma normale (3NF)

Si dice che uno schema relazionale è in terza forma normale (3NF) quando è in seconda forma normale (2NF) e tutti i suoi attributi non-chiave dipen-

dono direttamente dalla chiave, cioè non possiede attributi non-chiave che dipendono da altri attributi non-chiave.

La terza forma normale elimina la dipendenza transitiva degli attributi dalla chiave.

4.2 PROCESSO DI NORMALIZZAZIONE

Partendo dalla figura 19 a pagina 21, si analizzano le varie chiavi e le varie dipendenze funzionali per ridurre lo schema relazionale precedentemente descritto in 2NF e in 3NF.

Si analizza la relazione FATTURA e si nota che essa risulta essere già in seconda forma normale (2NF); infatti essendoci un solo attributo chiave (*Id*) tutti gli altri attributi dipendono funzionalmente da quest'ultimo. Infine non essendoci dipendenze funzionali transitive, tale relazione risulta essere anche in terza forma normale (3NF).

Lo stesso ragionamento si può applicare alle relazioni SCADENZA, INDIRIZZO_DI_SPEDIZIONE e TRACKING.

Si considerano ora le seguenti relazioni:

- MARCHIO
- CARATTERISTICA
- REPARTO
- UTENTE_REGISTRATO
- PAGAMENTO

Tutte le relazioni sono formate da un unico attributo chiave e da una chiave esterna, pertanto risultano essere in 3NF.

La relazione UTENTE_AMMINISTRATORE è formata da un'unico attributo chiave e presenta l'attributo *Email_USA* non-primo che dipende funzionalmente, direttamente e completamente dalla chiave; quindi ne segue che la relazione è in 3NF.

Si considerano ora le seguenti relazioni:

- PRODOTTO
- COUPON

Entrambe le relazioni sono formate da un unico attributo chiave e da due chiavi esterne, pertanto risultano essere in 3NF.

La relazione ORDINE presenta un attributo chiave e tre chiavi esterne, quindi risulta essere in 3NF.

La relazione RICETTA presenta due attributi chiave e una esterna, quindi risulta essere in 3NF.

La relazioni LISTA_DELLA_SPESA e INDIRIZZO presentano due attributi chiave, quindi risulta essere in 3NF.

Per quanto riguarda le relazioni:

- COMPONE
- CONTENUTO_IN_CARRELLO
- HA_PARTICOLARE
- GESTISCE_PRODOTTI

Tali relazioni sono formate da due attributi chiave che sono chiavi esterne, pertanto risultano essere in 3NF.

Le relazioni CONTENUTO_IN_LISTA e IMPIEGATO_IN presentano tre attributi chiave che sono chiavi esterne, quindi risultano essere in 3NF.

Dall'analisi svolta si deduce che la figura **??** a pagina ?? è in terza forma normale (3NF).

5 CODICE SQL

5.1 DDL

Si riporta il codice SQL utilizzato per creare le tabelle che costituiscono la base di dati e i vincoli di integrità referenziale.

5.1.1 Utente

Struttura della tabella *UTENTE*:

```
CREATE TABLE IF NOT EXISTS ‘UTENTE‘ (
        ‘Email‘ VARCHAR(30) NOT NULL,
        ‘Password‘ VARCHAR(32) NOT NULL,
        ‘Chiave_privata‘ VARCHAR(32) NOT NULL,
        ‘Nome‘ VARCHAR(30) NOT NULL,
        ‘Cognome‘ VARCHAR(30) NOT NULL,
        ‘Tipo‘ VARCHAR(1) NOT NULL,
        ‘Attivo‘ BOOLEAN NOT NULL DEFAULT TRUE,
        PRIMARY KEY (‘Email‘)
) ENGINE = InnoDB;
```

5.1.2 Amministratore

Struttura della tabella *AMMINISTRATORE*:

```
CREATE TABLE IF NOT EXISTS ‘AMMINISTRATORE‘ (
        ‘Email_U‘ VARCHAR(30) NOT NULL,
        PRIMARY KEY (‘Email_U‘),
        FOREIGN KEY (‘Email_U‘) REFERENCES UTENTE(‘Email‘)
) ENGINE = InnoDB;
```

5.1.3 Registrato

Struttura della tabella *REGISTRATO*:

```
CREATE TABLE IF NOT EXISTS ‘REGISTRATO‘ (
        ‘Email_U‘ VARCHAR(30) NOT NULL,
        ‘Email_A‘ VARCHAR(30) NOT NULL,
        ‘Telefono‘ VARCHAR(15) NOT NULL,
        ‘Indirizzo‘ VARCHAR(50) NOT NULL,
        PRIMARY KEY (‘Email_U‘),
        FOREIGN KEY (‘Email_U‘) REFERENCES UTENTE(‘Email‘),
        FOREIGN KEY (‘Email_A‘) REFERENCES AMMINISTRATORE(‘Email_U‘)
) ENGINE = InnoDB;
```

5.1.4 Arbitro

Struttura della tabella *ARBITRO*:

```
CREATE TABLE IF NOT EXISTS ‘ARBITRO‘ (
        ‘Email_R‘ VARCHAR(30) NOT NULL,
        ‘Data_di_nascita‘ DATE NOT NULL,
        ‘Nazionalita‘ VARCHAR(50) NOT NULL,
        ‘Foto‘ VARCHAR(50) NOT NULL,
        PRIMARY KEY (‘Email_R‘),
        FOREIGN KEY (‘Email_R‘) REFERENCES REGISTRATO(‘Email_U‘)
) ENGINE = InnoDB;
```

5.1.5 Gestore Squadra

Struttura della tabella *GESTORE_SQUADRA*:

```
CREATE TABLE IF NOT EXISTS ‘GESTORE_SQUADRA‘ (
        ‘Email_R‘ VARCHAR(30) NOT NULL,
        PRIMARY KEY (‘Email_R‘),
        FOREIGN KEY (‘Email_R‘) REFERENCES REGISTRATO(‘Email_U‘)
) ENGINE = InnoDB;
```

5.1.6 Torneo

Struttura della tabella *TORNEO*:

```
CREATE TABLE IF NOT EXISTS ‘TORNEO‘ (
        ‘Id‘ INT UNSIGNED NOT NULL,
        ‘Email_A‘ VARCHAR(30) NOT NULL,
        ‘Tipo‘ CHAR(1) NOT NULL,
        ‘Nome‘ VARCHAR(30) NOT NULL,
        ‘Descrizione‘ TEXT(5000),
        PRIMARY KEY (‘Id‘),
        FOREIGN KEY (‘Email_A‘) REFERENCES AMMINISTRATORE(‘Email_U‘)
) ENGINE = InnoDB;
```

5.1.7 Partita

Struttura della tabella *PARTITA*:

```
CREATE TABLE IF NOT EXISTS ‘PARTITA‘ (
        ‘Id‘ INT UNSIGNED NOT NULL,
        ‘Tipo_torneo‘ CHAR(1) NOT NULL,
        ‘Luogo‘ VARCHAR(30) NOT NULL,
        ‘Data‘ DATE NOT NULL,
        ‘Ora‘ TIME NOT NULL,
        PRIMARY KEY (‘Id‘)
) ENGINE = InnoDB;
```

5.1.8 Referto

Struttura della tabella *REFERTO*:

```
CREATE TABLE IF NOT EXISTS ‘REFERTO‘ (
        ‘Id‘ INT UNSIGNED NOT NULL,
        ‘Id_P‘ INT UNSIGNED NOT NULL,
        ‘Email_AR‘ VARCHAR(30) NOT NULL,
        ‘Ora_inizio‘ TIME NOT NULL,
        ‘Ora_fine‘ TIME NOT NULL,
        ‘Risultato‘ VARCHAR(10) NOT NULL,
        ‘Compilato‘ BOOLEAN NOT NULL,
        PRIMARY KEY (‘Id‘),
        FOREIGN KEY (‘Id_P‘) REFERENCES PARTITA(‘Id‘),
        FOREIGN KEY (‘Email_AR‘) REFERENCES ARBITRO(‘Email_R‘)
) ENGINE = InnoDB;
```

5.1.9 Squadra

Struttura della tabella *SQUADRA*:

```
CREATE TABLE IF NOT EXISTS ‘SQUADRA‘ (
        ‘Id‘ INT UNSIGNED NOT NULL,
        ‘Email_G‘ VARCHAR(30) NOT NULL,
        ‘Compilata‘ BOOLEAN NOT NULL,
        ‘Nome‘ VARCHAR(30) NOT NULL,
        ‘Descrizione‘ TEXT(5000),
        ‘Sede‘ VARCHAR(30),
        ‘Logo‘ VARCHAR(50),
        ‘Immagine‘ VARCHAR(50),
        ‘Nome_sponsor‘ VARCHAR(30),
        ‘Logo_sponsor‘ VARCHAR(50),
        PRIMARY KEY (‘Id‘),
        FOREIGN KEY (‘Email_G‘) REFERENCES GESTORE_SQUADRA(‘Email_R‘)
) ENGINE = InnoDB;
```

5.1.10 Classifica

Struttura della tabella *CLASSIFICA*:

```
CREATE TABLE IF NOT EXISTS `CLASSIFICA` (
        `Id_T` INT UNSIGNED NOT NULL,
        `Id_S` INT UNSIGNED NOT NULL,
        `Punti` INT UNSIGNED DEFAULT 0,
        `Vittorie` INT UNSIGNED DEFAULT 0,
        `Sconfitte` INT UNSIGNED DEFAULT 0,
        `Pareggi` INT UNSIGNED DEFAULT 0,
        `Partite` INT UNSIGNED DEFAULT 0,
        `Goal_fatti` INT UNSIGNED DEFAULT 0,
        `Goal_subiti` INT UNSIGNED DEFAULT 0,
        PRIMARY KEY (`Id_T`, `Id_S`),
        FOREIGN KEY (`Id_T`) REFERENCES TORNEO(`Id`),
        FOREIGN KEY (`Id_S`) REFERENCES SQUADRA(`Id`)
) ENGINE = InnoDB;
```

5.1.11 Giocatore

Struttura della tabella *GIOCATORE*:

```
CREATE TABLE IF NOT EXISTS `GIOCATORE` (
        `Id` INT UNSIGNED NOT NULL,
        `Id_S` INT UNSIGNED NOT NULL,
        `Attivo` BOOLEAN DEFAULT TRUE,
        `Numero_maglia` INT UNSIGNED,
        `Ruolo` VARCHAR(50) NOT NULL,
        `Nome` VARCHAR(30) NOT NULL,
        `Cognome` VARCHAR(30) NOT NULL,
        `Data_di_nascita` DATE NOT NULL,
        `Nazionalita` VARCHAR(20) NOT NULL,
        `Foto` VARCHAR(50) NOT NULL,
        `Descrizione` TEXT(5000),
        `Goal` INT UNSIGNED DEFAULT 0,
        `Ammonizioni` INT UNSIGNED DEFAULT 0,
        `Espulsioni` INT UNSIGNED DEFAULT 0,
        PRIMARY KEY (`Id`),
        FOREIGN KEY (`Id_S`) REFERENCES SQUADRA(`Id`)
) ENGINE = InnoDB;
```

5.1.12 Marcatore

Struttura della tabella *MARCATORE*:

```
CREATE TABLE IF NOT EXISTS ‘MARCATORE‘ (
        ‘Id_R‘ INT UNSIGNED NOT NULL,
        ‘Id_G‘ INT UNSIGNED NOT NULL,
        ‘Goal‘ INT UNSIGNED DEFAULT 0,
        PRIMARY KEY (‘Id_R‘, ‘Id_G‘),
        FOREIGN KEY (‘Id_R‘) REFERENCES REFERTO(‘Id‘),
        FOREIGN KEY (‘Id_G‘) REFERENCES GIOCATORE(‘Id‘)
) ENGINE = InnoDB;
```

5.1.13 Partecipano

Struttura della tabella *PARTECIPANO*:

```
CREATE TABLE IF NOT EXISTS ‘PARTECIPANO‘ (
        ‘Id_S‘ INT UNSIGNED NOT NULL,
        ‘Id_T‘ INT UNSIGNED NOT NULL,
        PRIMARY KEY (‘Id_S‘, ‘Id_T‘),
        FOREIGN KEY (‘Id_S‘) REFERENCES SQUADRA(‘Id‘),
        FOREIGN KEY (‘Id_T‘) REFERENCES TORNEO(‘Id‘)
) ENGINE = InnoDB;
```

5.1.14 Cartellino

Struttura della tabella *CARTELLINO*:

```
CREATE TABLE IF NOT EXISTS ‘CARTELLINO‘ (
        ‘Id_R‘ INT UNSIGNED NOT NULL,
        ‘Id_G‘ INT UNSIGNED NOT NULL,
        ‘Numero‘ INT UNSIGNED DEFAULT 0,
        ‘Ammonizione‘ BOOLEAN,
        ‘Espulsione‘ BOOLEAN,
        PRIMARY KEY (‘Id_R‘, ‘Id_G‘),
        FOREIGN KEY (‘Id_R‘) REFERENCES REFERTO(‘Id‘),
        FOREIGN KEY (‘Id_G‘) REFERENCES GIOCATORE(‘Id‘)
) ENGINE = InnoDB;
```

5.1.15 Scelto

Struttura della tabella *SCELTO*:

```
CREATE TABLE IF NOT EXISTS ‘SCELTO‘ (
        ‘Id_P‘ INT UNSIGNED NOT NULL,
        ‘Email_AR‘ VARCHAR(30) NOT NULL,
        ‘Email_A‘ VARCHAR(30) NOT NULL,
        PRIMARY KEY (‘Id_P‘, ‘Email_AR‘, ‘Email_A‘),
        FOREIGN KEY (‘Id_P‘) REFERENCES PARTITA(‘Id‘),
        FOREIGN KEY (‘Email_AR‘) REFERENCES ARBITRO(‘Email_R‘),
        FOREIGN KEY (‘Email_A‘) REFERENCES AMMINISTRATORE(‘Email_U‘)
) ENGINE = InnoDB;
```

5.1.16 Formazione

Struttura della tabella *FORMAZIONE*:

```
CREATE TABLE IF NOT EXISTS ‘FORMAZIONE‘ (
        ‘Id_R‘ INT UNSIGNED NOT NULL,
        ‘Id_S‘ INT UNSIGNED NOT NULL,
        ‘Id_G‘ INT UNSIGNED NOT NULL,
        ‘Riserva‘ BOOLEAN NOT NULL,
        PRIMARY KEY (‘Id_R‘, ‘Id_S‘, ‘Id_G‘),
        FOREIGN KEY (‘Id_R‘) REFERENCES REFERTO(‘Id‘),
        FOREIGN KEY (‘Id_S‘) REFERENCES SQUADRA(‘Id‘),
        FOREIGN KEY (‘Id_G‘) REFERENCES GIOCATORE(‘Id‘)
) ENGINE = InnoDB;
```

5.1.17 Giocano

Struttura della tabella *GIOCANO*:

```
CREATE TABLE IF NOT EXISTS ‘GIOCANO‘ (
        ‘Id_P‘ INT UNSIGNED NOT NULL,
        ‘Id_S1‘ INT UNSIGNED NOT NULL,
        ‘Id_S2‘ INT UNSIGNED NOT NULL,
        ‘Id_T‘ INT UNSIGNED NOT NULL,
        ‘Nome_partita‘ VARCHAR(30) NOT NULL,
        PRIMARY KEY (‘Id_P‘, ‘Id_S1‘, ‘Id_S2‘, ‘Id_T‘),
        FOREIGN KEY (‘Id_P‘) REFERENCES PARTITA(‘Id‘),
        FOREIGN KEY (‘Id_S1‘) REFERENCES SQUADRA(‘Id‘),
        FOREIGN KEY (‘Id_S2‘) REFERENCES SQUADRA(‘Id‘),
        FOREIGN KEY (‘Id_T‘) REFERENCES TORNEO(‘Id‘)
) ENGINE = InnoDB;
```

5.2 INTERROGAZIONI

Tra le molte interrogazioni utilizzate dall'applicazione web riguardo il database precedente si riportano quelle che meritano un commento.

5.2.1 INSERT

Inserimento arbitro

```
UPDATE arbitro
SET
Foto='Conversion.getDatabaseString(foto)', data_n='Conversion.
    getDatabaseString(data_n)',
Nazionalita='Conversion.getDatabaseString(nazionalita)', Carriera='
    Conversion.getDatabaseString(carriera)'
WHERE SSN=(SELECT SSN FROM utente WHERE email='"+Conversion.
    getDatabaseString(email)')
```

Eliminazione di un utente dall'applicazione

L'amministratore ha la possibilità di togliere i privilegi di accesso all'applicazione ad un utente registrato. In realtà non viene eliminato dal database ma viene solo cambiata la sua visibilità attraverso l'attributo *Attivo*.

```
UPDATE utente
SET flag='N'
WHERE  email='Conversion.getDatabaseString(email)'
```

Inserimento dei marcatori di una partita

Nella creazione del referto viene utilizzata questa query per inserire in una tabella tutti i giocatori che hanno segnato dei goal nella partita a cui fa riferimento il referto. Una cosa simile è stata fatta per la cartella Cartellini.

```
INSERT INTO marcatori (ID_Referto, SSN_Giocatore, Goal)
VALUES ('Conversion.getDatabaseString(""+IDReferto)', 'Conversion.
    getDatabaseString(""+SSNGiocatore)', 'Conversion.getDatabaseString
    (""+goal)')
```

5.2.2 UPDATE

Primo accesso all'applicazione del gestore di una squadra

Primo accesso all'applicazione del gestore di una squadra

```
UPDATE squadra
SET    Nome_squadra = 'Conversion.getDatabaseString(nomeSquadra)',
```

```
Logo_squadra = Conversion.getDatabaseString(logoSquadra),
Immagine_squadra = 'Conversion.getDatabaseString(immagine Squadra)',
Nome_sponsor = 'Conversion.getDatabaseString(nomeSponsor)',
Logo_sponsor = 'Conversion.getDatabaseString(logoSponsor)',
Sede = 'Conversion.getDatabaseString(sede)',
Descrizione = 'Conversion.getDatabaseString(descrizione)',
flag = 'Y'
WHERE  SSN_Gestore = 'Conversion.getDatabaseString(""+gestore)'
```

Inserimento di un nuovo arbitro

L'amministratore ha la possibilità di inserire un nuovo arbitro, a esso gli viene attribuita la lettera "R" (dall'inglese Refree) nell'attributo *Tipo*. Le query sono tre perchè l'arbitro è un registrato che a sua volta è un cliente.

Inserimento utente

```
UPDATE utente
SET
flag='Y', type='R', Nome='Conversion.getDatabaseString(nome)', Cognome='
    Conversion.getDatabaseString(cognome)',
Password='Conversion.getDatabaseString(password)'
WHERE email='Conversion.getDatabaseString(email)'
```

Inserimento registrato

```
UPDATE registrato
SET
Telefono='Conversion.getDatabaseString(telefono)', Indirizzo='Conversion
    .getDatabaseString(indirizzo)',
SSN_Admin='Conversion.getDatabaseString(""+Admin)'
WHERE SSN=(SELECT SSN              FROM utente            WHERE email='
    Conversion.getDatabaseString(email)')
```

5.2.3 SELECT

Ricerca partite senza nessun arbitro

```
SELECT p.Data_partita, R.ID_Referto,g., r.flagRef, t.Nome AS nomeTorneo
FROM referto AS r, giocano AS g, torneo AS t, partita AS p
WHERE   r.flagRef='N' AND
G.ID_Partita=r.ID_Partita AND
g.ID_Torneo=t.ID_Torneo AND
r.ID_Partita=p.ID_Partita AND
g.ID_Partita=p.ID_Partita AND
(G.ID_SquadraA IN
(SELECT ID_SQUADRA
```

```
FROM SQUADRA
WHERE ssn_gESTORE='"+Conversion.getDatabaseString(""+gestore)  AND
ID_Squadra<>'0')
OR G.ID_SquadraB IN
(SELECT ID_SQUADRA
FROM SQUADRA
WHERE ssn_gESTORE='"+Conversion.getDatabaseString(""+gestore) AND
ID_Squadra<>'o'))
AND R.ID_Referto NOT IN
(SELECT ID_Referto
FROM formazione
WHERE ID_Squadra=(SELECT ID_Squadra
FROM squadra
WHERE SSN_Gestore='"+Conversion.getDatabaseString(""+gestore)))
ORDER BY ID_Referto
```

Ricerca dei referti assegnati ad un arbitro da un amministratore

Quando l'amministratore del sistema deve sorteggiare un arbitro per una determinata partita riguardante una certa fase di un determinato torneo, deve poter visualizzare soltanto gli incontri per cui il sorteggio non sia già stato effettuato.

```
SELECT  Ref.ID_Referto, Ref.Risultato, Ref.Ora_inizio,
Ref.Ora_fine, Ref.ID_Partita, Ref.Risultato,
Ref.flagRef,U.Nome AS nomeArbitro,
U.cognome AS cognomeArbitro,T.ID_Torneo,
T.tipologia as tipoTorneo, T.Nome as nomeTorneo,
P.Data_Partita, P.Luogo
FROM    referto AS Ref, utente AS U, registrato AS Reg,
giocano as G, torneo as T, partita as P
WHERE   U.Flag='Y' AND U.type='R' AND
U.SSN=Reg.SSN AND U.SSN=Ref.SSN_Arbitro
AND g.id_partita=Ref.ID_Partita
AND t.ID_Torneo=G.ID_Torneo
AND p.ID_Partita=G.ID_Partita
AND Reg.SSN_Admin='"+Conversion.getDatabaseString(""+Admin)'
ORDER BY Ref.ID_Referto
```

Estrazione dei cartellini dei giocatori in una determinata partita

Questa query viene utilizzata dall'utente pubblico quando va a visionare il referto relativo ad una determinata partita. Vengono estratte dalla tabella i cartellini di tutti i giocatori e le espulsioni e le ammonizioni relative ad una determinata partita. GLi *Id* dei giocatori verranno poi confrontati con gli *Id* dei giocatori delle due formazioni e se risultano uguali allora estraggo il flag ammonito/espulso e li visualizzo a schermo. Una query simile è stata fatta per la cartella Marcatori.

```
SELECT g.Nome, g.Cognome, g.Foto, g.Foto, g.SSN_Gct, G.ID_Squadra, c.
    Flag_Ammonito as FlagAmmonito, c.Flag_Espulso as FlagEspulso
FROM giocatore AS g, cartellini_gialli AS C
WHERE C.SSN_Giocatore=G.SSN_Gct  AND C.ID_Referto='Conversion.
    getDatabaseString(""+IDReferto)'
AND (G.ID_Squadra='Conversion.getDatabaseString(""+IDSquadraA)' OR G.
    ID_Squadra='Conversion.getDatabaseString(""+IDSquadraB)')
```