Università degli Studi di Bologna

Facoltà di Ingegneria e Scienze Informatiche (A.A. 2014/2015)

Studenti:

Fabio Pazzini - Matricola: 0000692345 - email: fabio.pazzini3@studio.unibo.it

Aldo Dushku - Matricola: 0000691687 - email: aldo.dushku@studio.unibo.it

Relazione per il Corso di Basi di Dati

Base di Dati per un Torneo di Karate

Indice

Analisi dei requisiti

Lo scopo di questo progetto è la realizzazione di una Base di Dati per coordinare e mantenere i dati relativi ad un Campionato di Karate, che si deve svolgere in una certa palestra. Il progetto considera le varie prove effettuate da determinati atleti, mostrando attenzione anche alla parte di giuria, a cui fanno parte arbitri, giudici ed un tavolo di giuria. La gara prevede l'esecuzione di un Kata su un Tatami (parte di pavimento destinata alla prova), allo svolgimento della quale viene dato un voto dall'arbitraggio. Ogni atleta appartiene, inoltre, ad una specifica categoria e proviene da una certa palestra, così come gli arbitri e i giudici.

Il programma prevede due tipi di utilizzazione: una in modalità "root", che consente l'aggiunta, la modifica e la rimozione di un certo elemento, ed una in modalità "user" che permette solamente operazioni di ricerca.

Il progetto non rispecchia totalmente una situazione realistica in quanto manca la gestione dei turni, eliminatorie e finali, e premiazioni.

La realizzazione del Database è stata effettuata utilizzando il DBMS PHPMyAdmin, con il linguaggio MySQL, mentre la parte relativa alla comunicazione, alle query, e a qualsiasi altro aspetto di interazione con esso è stata eseguita scrivendo codice Java e utilizzando le librerie del JDBC. Le query, invece, sono scritte in SQL.

Requisiti in linguaggio naturale

"*L'associazione sportiva di Karate KarateKid richiede un sistema informatico per la gestione del Torneo di Karate. Il sistema deve gestire la manifestazione come segue: una gara si deve svolgere in una determinata palestra (1*), e prevede di sostenere una prova che consiste nell'esecuzione di un Kata alla quale gli arbitri daranno una votazione. La prova verrà coordinata da un tavolo di giuria al quale fanno parte dei giudici che provengono, come atleti e arbitri, da una certa palestra (2*). Tutte le categorie possono partecipare, e vogliamo che venga visualizzato il primo classificato in base ai voti (3*). Gli atleti eseguiranno la prova su un tatami, che è la parte di pavimentazione destinata alla gara. Della manifestazione si vogliono mantenere la data e la palestra in cui si è svolta (4*).*

Si vuole mantenere lo storico riguardante l'appartenenza di un certo atleta ad una determinata categoria (5).*

L'amministratore di sistema, identificato dalla password "root", deve avere la possibilità di accedere al Database e di modificarlo aggiungendo, rimuovendo o modificando elementi a sua scelta; l'utente finale deve invece avere la possibilità di eseguire solo delle ricerche all'interno della Base di Dati."

1 : Una palestra può ospitare più gare, mentre una gara può essere ospitata da una ed una sola palestra.*

2 : Si deve evitare che un giudice o un arbitro partecipi alla gara.*

3 : Per ogni categoria si vuole visualizzare il primo classificato.*

4 : In una certa data, e nella stessa palestra, non si possono svolgere due gare (anche con nomi diversi).*

5 : Un atleta può passare da una categoria all'altra nel corso degli anni.*

Scelta della strategia di Progetto

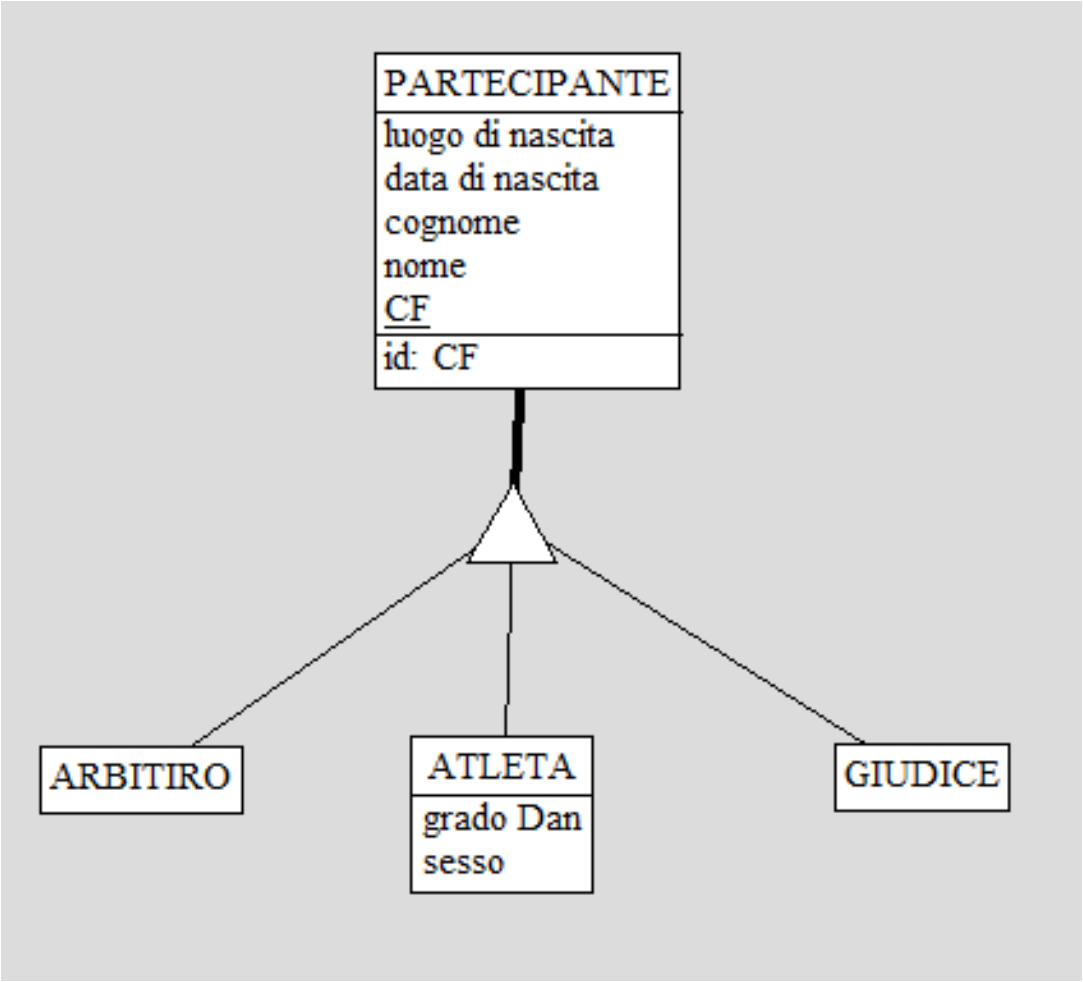
Riscriviamo il testo del Progetto e sottolineiamo i concetti fondamentali, quelli che ci consentiranno di realizzare un primo schema scheletro del nostro elaborato.

"*L'associazione sportiva di Karate KarateKid richiede un sistema informatico per la gestione del Torneo di Karate. Il sistema deve gestire la manifestazione come segue: una gara si deve svolgere in una determinata palestra (1*), e prevede di sostenere una prova che consiste nell'esecuzione di un Kata alla quale gli arbitri daranno una votazione. La prova verrà coordinata da un tavolo di giuria al quale fanno parte dei giudici che provengono, come atleti e arbitri, da una certa palestra (2*). Tutte le categorie possono partecipare, e vogliamo che venga visualizzato il primo classificato in base ai voti (3*). Gli atleti eseguiranno la prova su un tatami, che è la parte di pavimentazione destinata alla gara. Della manifestazione si vogliono mantenere la data e la palestra in cui si è svolta (4*).*

Sviluppo delle entità

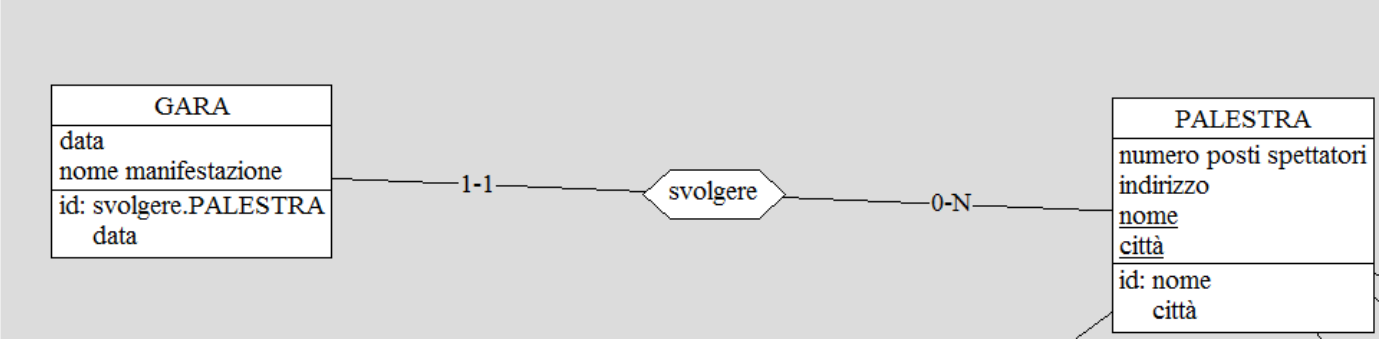
Entità PARTECIPANTE:

Dato che arbitri, atleti e giudici di partecipano alla gara, può essere conveniente introdurre una generalizzazione a cui daremo il nome di PARTECIPANTE. Un generico partecipante sarà identificato dal proprio Codice Fiscale, e di esso verranno mantenute informazioni anagrafiche quali nome, cognome, luogo di nascita, data di nascita. Questi dati verranno ereditati dalle singole entità ATLETA, ARBITRO e GIUDICE. Dell'atleta si vogliono anche mantenere informazioni legate al sesso e al Dan (grado della cintura nera).



Entità PALESTRA:

Una palestra è identificata dal proprio nome e dalla palestra in cui si trova, e da una palestra provengono vari partecipanti, indipendentemente che essi siano atleti, arbitri o giudici. Di una palestra si mantiene anche l'informazione riguardante l'indirizzo presso cui si trova.

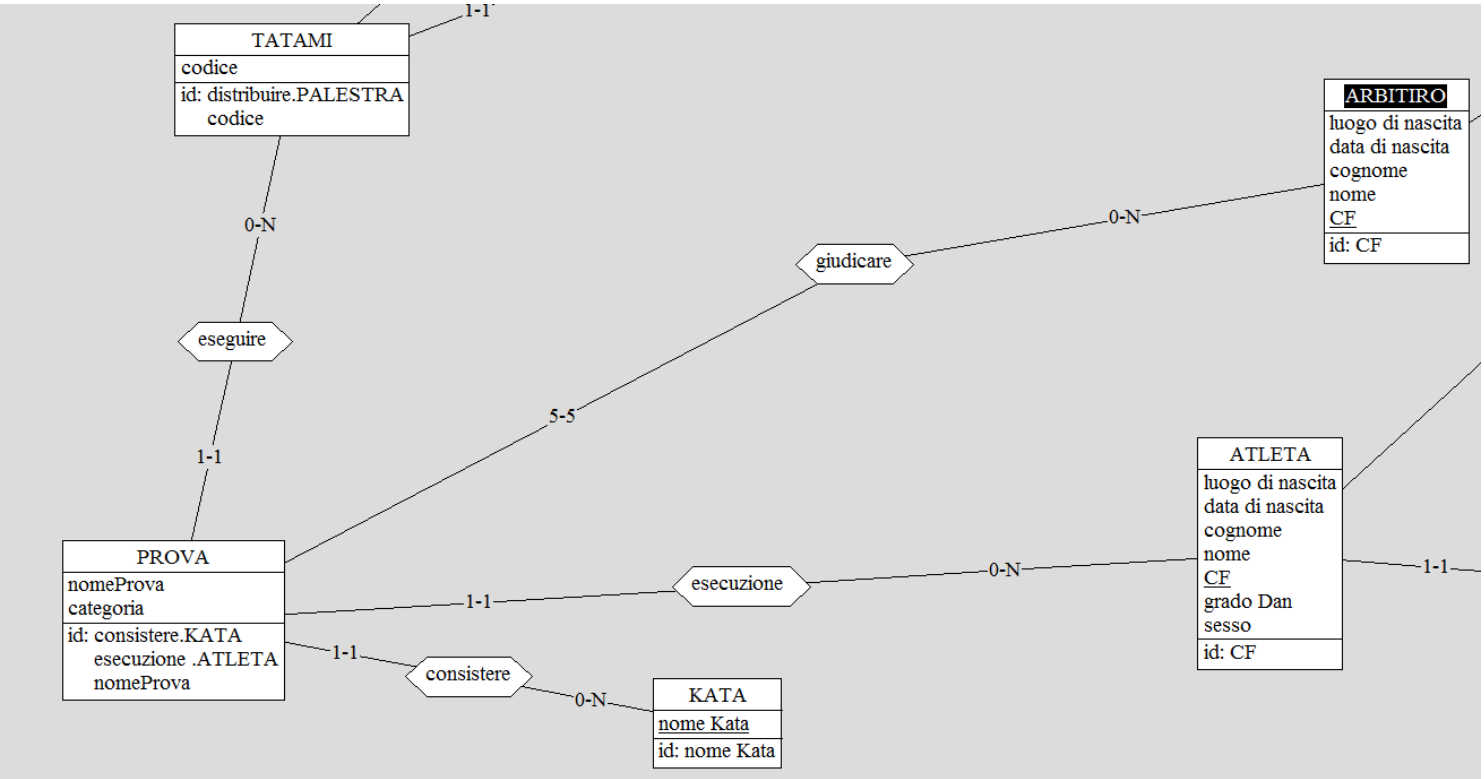


Entità PROVA:

L'entità PROVA è una delle più importanti entità dello schema, in quanto rappresenta la singola prova eseguita da un determinato atleta. La prova, come specificato nel testo del Progetto, prevede l'esecuzione di un Kata e l'assegnazione di un voto da parte degli arbitri. Di conseguenza, imposteremo le cardinalità in modo tale da avere come identificatore dell'entità la composizione del nome della prova, del Codice Fiscale dell'atleta e dal nome del Kata eseguito. Di una prova si vuole anche mantenere il Codice Fiscale dell'arbitro che ha giudicato e del voto che ha dato. Inoltre, per

mantenere lo storico delle prove e della categoria in cui la prova è stata effettuata, abbiamo inserito un campo “categoria” che dovremo riempire al momento di un nuovo inserimento.

Dal momento che un prova si esegue su un Tatami, mentre su un Tatami si possono svolgere N prove, importiamo l’identificatore di Tatami che è il suo codice univoco.



Entità CATEGORIA, TAVOLO DI GIURIA, GARA, KATA:

Queste, essendo enti minori, le tratteremo tutte insieme. KATA e CATEGORIE sono entità che contengono un solo campo e le abbiamo inserite per facilitare il controllo dell’inserimento di una prova e di un atleta (un atleta appartiene ad una certa categoria e deve eseguire un Kata, esistenti nel Database).

La GARA è l'entità che rappresenta il contesto dell'applicativo ed è identificata dalla composizione tra la data e la palestra in cui si tiene, mentre TAVOLO DI GIURIA è l'oggetto che rappresenta il tavolo che presiede e coordina ogni tatami, composto da 5 giudici. Ogni tavolo è identificato da un proprio codice; all'interno del nostro Progetto non ha particolare importanza, ma ci è sembrato più opportuno conservare la giuria poiché in un Torneo reale essa occupa il ruolo di verificare la presenza di irregolarità causate da errori umani.

Schema concettuale finale:

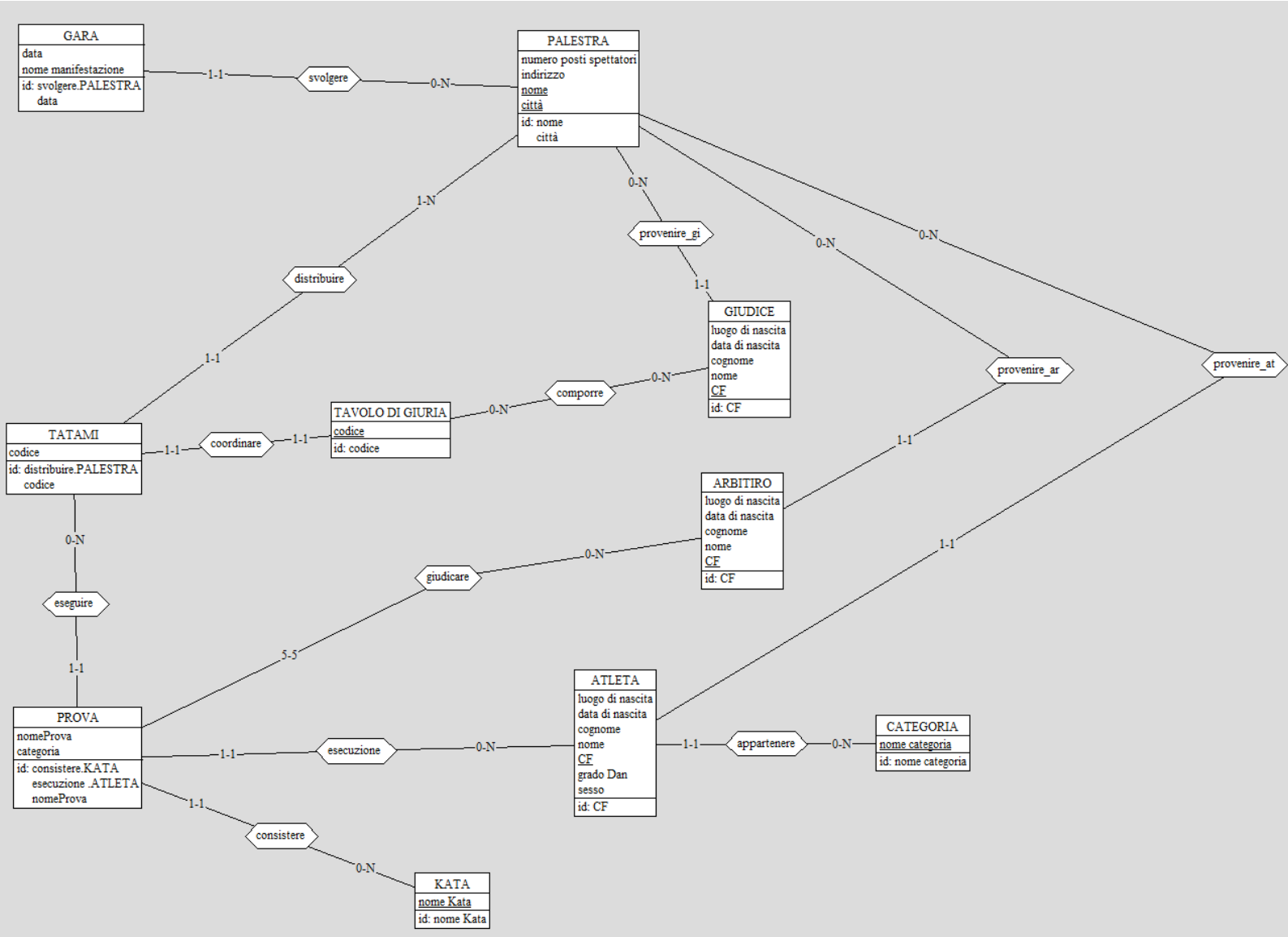


TABELLA DEI VOLUMI

CONCETTO	TIPO	VOLUME
Palestra	E	400
Città	E	200
Gare	E	40
Atleti	E	8000
Prove	E	8000
Katà	E	10
Tatami	E	2000
Giudici	E	800
Tavolo giuria	E	200
Categorie	E	4
Arbitri	E	800
Svolgere	R	40
Distribuire	R	2000
Provenire Gi	R	800
Provenire At	R	8000
Provenire Ar	R	800
Coordinare	R	2000
Comporre	R	600
Eseguire	R	8000
Esecuzione	R	8000
Appartiene	R	8000

Schema di navigazione

Le principali operazioni richieste al sistema sono:

- Inserimento di un nuovo partecipante
- Inserimento di una nuova prova
- Ricerca dei partecipanti
- Ricerche sulle prove
- Ricerca del migliore atleta di ogni categoria
- Operazioni di update e rimozione

Vediamo ora nello schema E/R, per ognuna di queste operazioni, i passi da compiere per ottenere il risultato desiderato.

Inserimento di un nuovo partecipante

Questa operazione inserisce un nuovo partecipante all'interno del Database, indipendentemente che esso sia un atleta, un giudice o un arbitro. Tuttavia vi è un vincolo legato all'esistenza della palestra dalla quale tale partecipante proviene: avendo la chiave importata dell'entità PALESTRA, ovvero la composizione del nome e della città in cui si trova, prima di poter inserire un nuovo record dobbiamo controllare l'esistenza di questa palestra nella Basi di Dati. Qualora non esistesse, viene richiesto all'Amministratore di sistema di inserire la suddetta palestra e reinserire il nuovo partecipante; viceversa, l'inserimento andrà a buon fine.

Inserimento di una nuova prova

La prova, come già sottolineato, è uno dei principali elementi del Database in quanto rappresenta l'esecuzione, da parte di un atleta, di un certo Kata.
Questa entità ha come identificatore la composizione del nome della prova, solitamente "kata" o "bunkai", del nome del Kata eseguito e del Codice Fiscale dell'atleta.
Prima di inserire un nuovo record, anche qui, dobbiamo eseguire i controlli alla pressione del pulsante Aggiungi presente nella GUI, che manda l'informazione al Database: se il CF dell'atleta inserito esiste e il nome del Kata è presente nella Base di Dati, allora si è in grado di inserire il nuovo record. Altrimenti, nel caso in cui venisse a mancare la prima condizione l'Amministratore dovrà creare l'atleta che ha sostenuto la prova, se venisse a mancare la seconda egli dovrà inserire il Kata nella rispettiva tabella. Dopodiché potrà inserire la prova.

Ricerca di un partecipante

La ricerca di un partecipante si svolge sostanzialmente ricercando, tramite l'operazione di SELECT, il Codice Fiscale della persona e restituendo le sue anagrafiche e mostrandole nella GUI. Se il CF non esiste non dovrà essere visualizzato nulla.

Ricerche sulle prove

Sull'entità PROVA vengono eseguite varie operazioni di ricerca leggermente più complesse. Ad esempio vi è la possibilità di ricercare il migliore atleta per ogni categoria, sfruttando l'attributo "votazione" che è univocamente determinato dal nome della prova, dal CF dell'atleta e dal Kata che ha eseguito. Si ha anche la possibilità di ottenere l'anagrafica di tutti gli atleti che hanno eseguito una certa prova su uno specifico Tatami.

Trasformazione dallo schema concettuale allo schema logico

In questa sezione vediamo come procedere per giungere alla rappresentazione sotto forma di schema logico del nostro E/R.

Innanzitutto, vediamo le chiavi importate e gli identificatori delle entità.

- **PARTECIPANTE**: essendo collegato a PALESTRA tramite un'associazione 1-N (in quanto un partecipante proviene da una ed una sola palestra mentre da una palestra possono provenire N atleti) questa entità dovrà possedere la chiave importata di PALESTRA. Al momento della trasformazione in schema logico eseguiremo un collasso verso il basso così da spostare gli attributi nelle sotto-entità della gerarchia ATLETA, ARBITRO e GIUDICE. ATLETA sarà collegato tramite un'associazione 1-N con CATEGORIA, per cui avremo come chiave importata anche il nome della categoria a cui appartiene.
- **PALESTRA**: una generica palestra è univocamente definita dalla composizione di "nome" e "città", in quanto non possono esistere due palestre che hanno lo stesso nome nella medesima città.
- **GARA**: questa entità rappresenta il contesto fondamentalmente. Una gara si svolge in una palestra mentre in una palestra si possono svolgere N gare, di conseguenza importiamo la chiave di palestra in questa entità e come identificatore combineremo la chiave importata di palestra e la data della gara. Da questo deduciamo che in una certa palestra e in una stessa data, non si può svolgere più di una gara.
- **KATA**: un Kata è univocamente determinato dal suo nome. Non esistono più Kata con lo stesso nome.

- **CATEGORIA**: entità utilizzata per tenere la lista delle categorie. Ogni categoria è univocamente determinata dal suo nome.
- **PROVA**: entità che rappresenta l'esecuzione della prova da parte di un atleta. Essa è collegata tramite un'associazione 1-N con ATLETA e tramite un'associazione 1-N con Kata, conseguentemente importiamo le chiavi sia dell'una che dell'altra entità e definiamo l'identificatore di PROVA come la composizione del CF dell'atleta, del nome del Kata eseguito e del nome della prova.
- **TATAMI**: un tatami è univocamente determinato dal suo codice, ma essendo collegato a PALESTRA tramite un'associazione 1-N avrà la chiave importata di PALESTRA. E' anche collegata tramite un'associazione 1-N a TAVOLO DI GIURIA, quindi avremo anche il codice del tavolo importato.
- **TAVOLO DI GIURIA**: un tavolo di giuria è univocamente determinato dal suo codice.

Eliminazione eventuali gerarchie

Dal momento che le entità ATLETA, ARBITRO e GIUDICE hanno i medesimi attributi (ad eccezione di ATLETA che possiede altri due campi), nello schema E/R le abbiamo messe come sotto-entità della gerarchia PARTECIPANTE. Ora, per passare allo schema logico, dobbiamo eseguire un collasso: dato che non possiamo utilizzare PARTECIPANTE in quanto un arbitro, ad esempio, non può essere un atleta, dobbiamo collassare verso il basso, spostando quindi ogni attributo dell'entità PARTECIPANTE nelle entità sottostanti.

Traduzione delle entità

Traduciamo qui le entità del nostro Database. Il campo sottolineato corrisponde alla chiave primaria, mentre le chiavi importate sono precedute dal nome (in maiuscolo) dell'entità da cui derivano.

- ATLETA(CF, nome, cognome, data_di_nascita, luogo_di_nascita, sesso, grado_Dan, PALESTRA: nomePalestra, PALESTRA: cittàPalestra)
- ARBITRO(CF, nome, cognome, data_di_nascita, luogo_di_nascita, PALESTRA: nomePalestra, PALESTRA: cittàPalestra)
- GIUDICE(CF, nome, cognome, data_di_nascita, luogo_di_nascita, PALESTRA:nomePalestra, PALESTRA:cittàPalestra)
- KATA(nome_Kata)
- CATEGORIA(nome_categoria)
- TATAMI(codice, TAVOLO_DI_GIURIA:CodiceTavoloGiuria, PALESTRA:nomePalestra, PALESTRA:cittàPalestra)
- TAVOLO_DI_GIURIA(codice)
- GARA(data, PALESTRA:nomePalestra, PALESTRA:cittàPalestra, nome_manifestazione)
- PALESTRA(nome, città, indirizzo, numero_posti_spettatori)
- PROVA(nomeProva, nomeKata, ATLETA:CodiceFiscaleAtleta, categoria, ARBITRO:CodiceFiscaleArbitro, TATAMI:codiceTatami, votazione)

Traduzione delle associazioni

Traduciamo le associazioni tra le entità.

- atleta_proviene(CodiceFiscaleAtleta, nomePalestra, cittàPalestra

Associazione che lega ATLETA e PALESTRA. Un atleta può provenire da una ed una sola palestra mentre da una palestra possono provenire N atleti.

- arbitro_proviene(CodiceFiscaleArbitro, nomePalestra, cittàPalestra)

  Associazione che lega ARBITRO e PALESTRA. Un arbitro può provenire da una ed una sola palestra mentre da una palestra possono provenire N arbitri.

- giudice_proviene(CodiceFiscaleGiudice, nomePalestra, cittàPalestra)

  Associazione che lega GIUDICE e PALESTRA. Un giudice può provenire da una ed una sola palestra mentre da una palestra possono provenire N giudici.

- ospitare(data, nomePalestra, cittàPalestra)

  Associazione che lega GARA e PALESTRA. Una gara può essere ospitata da una sola palestra mentre una palestra può ospitare N gare.

- eseguire(CodiceFiscaleAtleta, nomeProva, nomeKata)

  Associazione che lega ATLETA e PROVA.

- prevedere(nomeKata, CodiceFiscaleAtleta, nomeProva)

  Associazione che lega PROVA e KATA. Una prova prevede l'esecuzione di un Kata mentre un Kata può essere eseguito in N prove.

- appartenere(CodiceFiscaleAtleta, nome_categoria)

  Associazione che lega ATLETA e CATEGORIA. Un atleta appartiene ad una ed una sola categoria, mentre ad una categoria possono appartenere N atleti.

- arbitrare(nomeKata, CodiceFiscaleAtleta, nomeProva, CodiceFiscaleArbitro)

  Associazione che lega PROVA e ARBITRO. Una prova viene giudicata da 5 arbitri mentre un arbitro può arbitrare N prove.

- comporre(CodiceFiscaleGiudice, CodiceTavoloGiuria)

  Associazione che lega GIUDICE e TAVOLO_DI_GIURIA. Dato che l'associazione

è N-N, al momento della realizzazione dello schema logico questa diventerà un'entità con chiave composta (CodiceFiscaleGiudice, CodiceTavoloGiuria).

- disporre(CodiceTatami, nomePalestra, cittàPalestra)

  Associazione che lega TATAMI e PALESTRA. Un tatami appartiene ad una ed una sola palestra mentre ad una palestra appartengono N tatami.

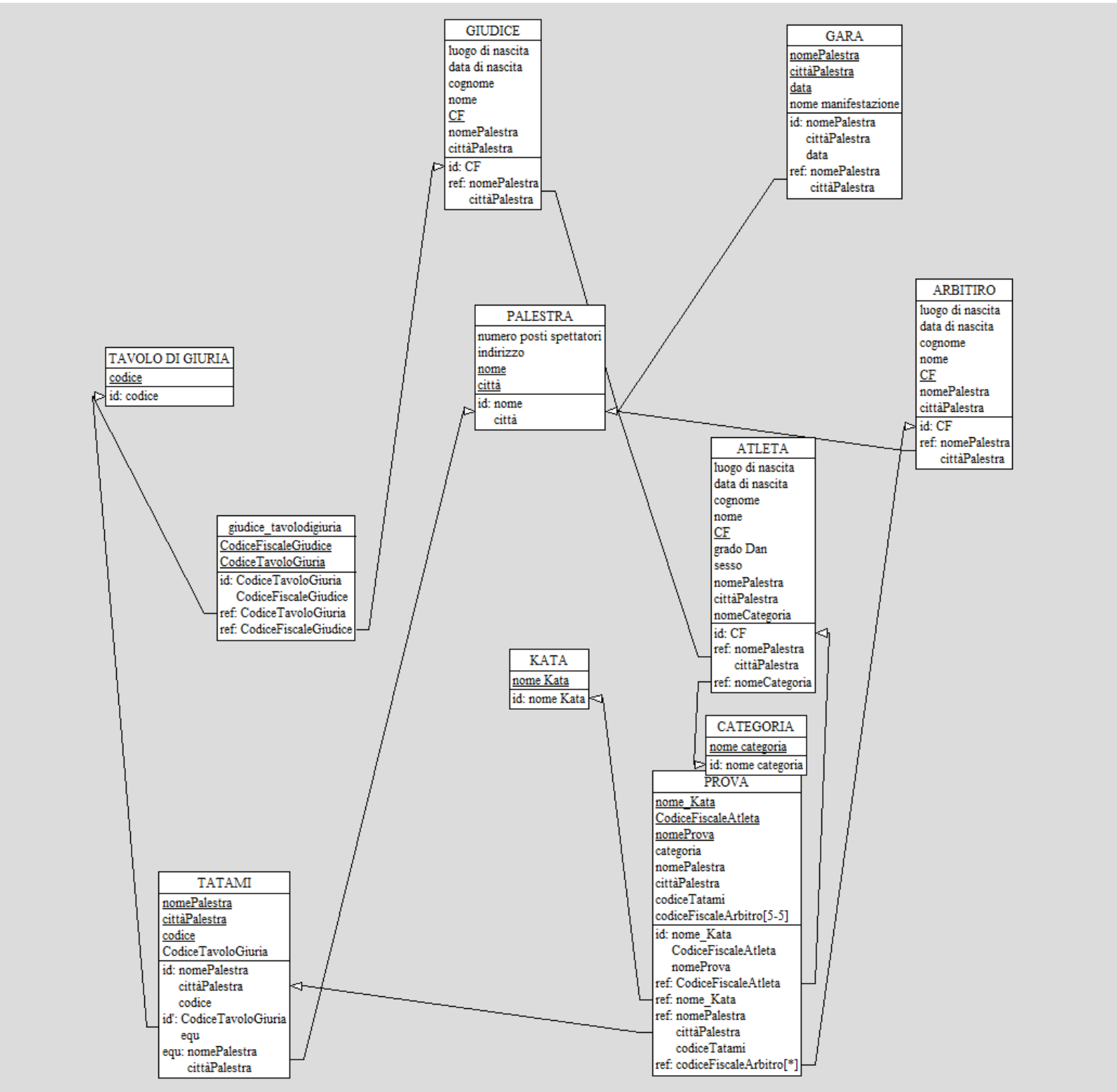
- esecuzione(nomeProva, nomeKata, CodiceFiscaleAtleta, codiceTatami)

  Associazione che lega PROVA e TATAMI. Su un tatami si possono eseguire N prove, mentre una prova si esegue su un solo tatami.

In conclusione, come tabelle, avremo:

- ATLETA(CF, nome, cognome, data_di_nascita, luogo_di_nascita, sesso, grado_Dan, PALESTRA: nomePalestra, PALESTRA: cittàPalestra)
- ARBITRO(CF, nome, cognome, data_di_nascita, luogo_di_nascita, PALESTRA: nomePalestra, PALESTRA: cittàPalestra)
- GIUDICE(CF, nome, cognome, data_di_nascita, luogo_di_nascita, PALESTRA:nomePalestra, PALESTRA:cittàPalestra)
- KATA(nome_Kata)
- CATEGORIA(nome_categoria)
- TATAMI(codice, TAVOLO_DI_GIURIA:CodiceTavoloGiuria, PALESTRA:nomePalestra, PALESTRA:cittàPalestra)
- TAVOLO_DI_GIURIA(codice)
- GARA(data, PALESTRA:nomePalestra, PALESTRA:cittàPalestra, nome_manifestazione)
- PALESTRA(nome, città, indirizzo, numero_posti_spettatori)
- PROVA(nomeProva, nomeKata, ATLETA:CodiceFiscaleAtleta, categoria, ARBITRO:CodiceFiscaleArbitro, TATAMI:codiceTatami, votazione)
- GIUDICE_TAVOLODIGIURIA(GIUDICE:CodiceFiscaleGiudice, TAVOLO_DI_GIURIA:CodiceTavoloGiuria)

Schema logico finale:



Query SQL implementate

RICERCA:

Atleti:

1) Numero di atleti appartenenti ad una certa categoria:

```
SELECT nomeCategoria, COUNT(*)
FROM  atleta
WHERE nomeCategoria = inputCategory
```

2) Dati atleti con un certo nome:

```
SELECT *
FROM atleta
WHERE nome = inputName
```

3) Dettagli degli atleti provenienti da una certa palestra:

```
SELECT *
FROM  atleta
AND nomePalestra = inputGymName
AND cittàPalestra = inputGymCity
```

Palestra:

1) Nomi palestre in una certa città:

```
SELECT DISTINCT nome
FROM palestra
WHERE città = inputCity
```

Gara:

1) Dettagli di tutte le gare organizzate in una certa palestra:

```
SELECT *
FROM gara
WHERE nomePalestra = inputGymName
AND cittàPalestra = inputGymCity
```

Prova:

1) Anagrafica di tutti atleti che hanno eseguito una prova su un determinato Tatami:

```
SELECT *
FROM atleta
WHERE CF IN (
        SELECT CodiceFiscaleAtleta
        FROM prova p, atleta a
        WHERE p.CodiceFiscaleAtleta = a.CF
        AND p.CodiceTatami = inputTatamiCode
    )
```

2) Somma dei punteggi ottenuti da un certo atleta:

```
SELECT SUM(votazione) AS sumVote
FROM prova
WHERE CodiceFiscaleAtleta = inputAthletCF
```

3) Atleta che ha realizzato il punteggio massimo in una certa categoria:

```
SELECT a.CF, p.votazione
FROM atleta a, prova p
WHERE a.CF = p.CodiceFiscaleAtleta
AND a.categoria = inputCategoryName
AND p.votazione = (
                    SELECT MAX(votazione) as maxVote
                    FROM atleta a2, prova p2

                    WHERE a2.CF = p2.CodiceFiscaleAtleta
                    AND a2.nomeCategoria = inputCategoryName )
```

4) Dettagli prova di un determinato atleta:

```
SELECT *
FROM prova
WHERE CodiceFiscaleAtleta = inputAthletCF
```

Giudice_tavolodigiuria:

1) Tutti i giudici in uno specifico tavolo di giuria:

```
SELECT *
FROM giudice
WHERE CF IN (
                SELECT CodiceFiscaleGiudice
                FROM giudice_tavolodigiuria
```

WHERE codiceTavoloGiuria = *inputJuryTableCode*

)

Giudice:

1) Anagrafica di un certo giudice:

SELECT *

FROM giudice

WHERE CF = *inputJudgeCF*

Query banali:

1) Nomi di tutti i Kata.

2) Lista delle categorie

3) Codici dei tavoli di giuria

INSERIMENTO:

Atleta:

INSERT INTO atleta(luogo_di_nascita, data_di_nascita, cognome, nome, CF, grado_Dan, sesso, nomePalestra, cittàPalestra) VALUES(*inputCity, inputBirthDay, inputSurname, inputName, inputCF, inputDan, inputSex, inputGymName, inputGymCity)*

Palestra:

INSERT INTO palestra(numero_posti_spettatori, indirizzo, nome, città)

VALUES(*inputSeats, inputAddress, inputName, inputCity)*

Arbitro:

INSERT INTO arbitro(luogo_di_nascita, data_di_nascita, cognome, nome, CF, grado, nomePalestra, cittàPalestra) VALUES(*inputCity, inputBirthDay, inputSurname, inputName, inputCF, inputGymName, inputGymCity)*

Giudice:

INSERT INTO giudice(luogo_di_nascita, data_di_nascita, cognome, nome, CF, grado, nomePalestra, cittàPalestra) VALUES(*inputCity, inputBirthDay, inputSurname, inputName, inputCF, inputGymName, inputGymCity)*

Prova:

INSERT INTO prova(nomeKata, CodiceFiscaleAtleta, nomeProva, categoria, CodiceFiscaleArbitro, codiceTatami, votazione) VALUES(*inputKata, inputAthletCF, inputPerformanceName, inputCategory, inputRefreeCF, inputTatamiCode, inputVote)*

Kata:

INSERT INTO kata(nome_Kata) VALUES(*inputKataName)*

Tatami:

INSERT INTO tatami(nomePalestra, cittàPalestra, codice, codiceTavoloGiuria) VALUES(*inputGymName, inputGymCity, inputTatamiCode, inputJuryCode)*

Tavolo di giuria:

INSERT INTO tavolo_di_giuria(codice) VALUES(*inputJuryCode)*

Gara:

INSERT INTO gara(data, nome_manifestazione, nomePalestra, cittàPalestra)

VALUES(*inputDate, inputCompetitionName, inputGymName, inputGymCity)*

Giudice tavolo di giuria:

INSERT INTO giudice_tavolodigiuria(CodiceFiscaleGiudice, codiceTavoloGiuria)

VALUES(*inputJudgeCF, inputJuryCF)*

Categoria:

INSERT INTO categoria(nome_categoria) VALUES(*inputCategoryName)*

Query di cancellazione:

Atleta - cancellazione atleta con un certo Codice Fiscale:

DELETE FROM atleta WHERE CF = *inputAthletCF*

Palestra - cancellazione palestra di una certa città con un certo nome:

DELETE FROM palestra WHERE nome = *inputGymName* AND città = *inputGymCity*

Prova - cancellazione di una prova dato CF dell'atleta, nome prova e nome Kata:

DELETE FROM prova WHERE nomeProva = *inputPerformanceName* AND nomeKata =

inputKataName AND CodiceFiscaleAtleta = *inputAtheltCF*

Gara - cancellazione di una gara avendo la data e la palestra in cui si è tenuta:

DELETE FROM gara WHERE data = *inputDate* AND nomePalestra = *inputGymName* AND cittàPalestra = *inputGymCity*

Kata - cancellazione di uno specifico Kata:

DELETE FROM kata WHERE nome_kata = *inputKataName*

Tatami - cancellazione di uno specifico Tatami:

DELETE FROM tatami WHERE codice = *inputTatamiCode*

Per gli arbitri ed i giudici la query di cancellazione è identica a quella relativa agli atleti.

Query di modifica

Le Query di modifica non sono state implementate per tutte le tabelle. Riportiamo quelle che abbiamo realizzato.

Palestra - update di una palestra dato il nome e la città in cui si trova:

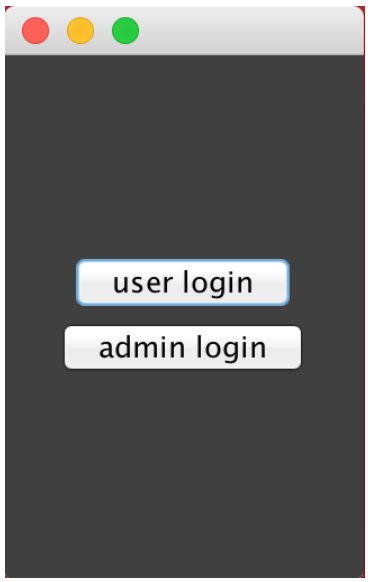
UPDATE palestra SET numero_posti_spettatori = *inputSeats ,* indirizzo = *inputAddress* WHERE nome = *inputGymName AND città =* inputGymCity

Gara - update della data di una gara

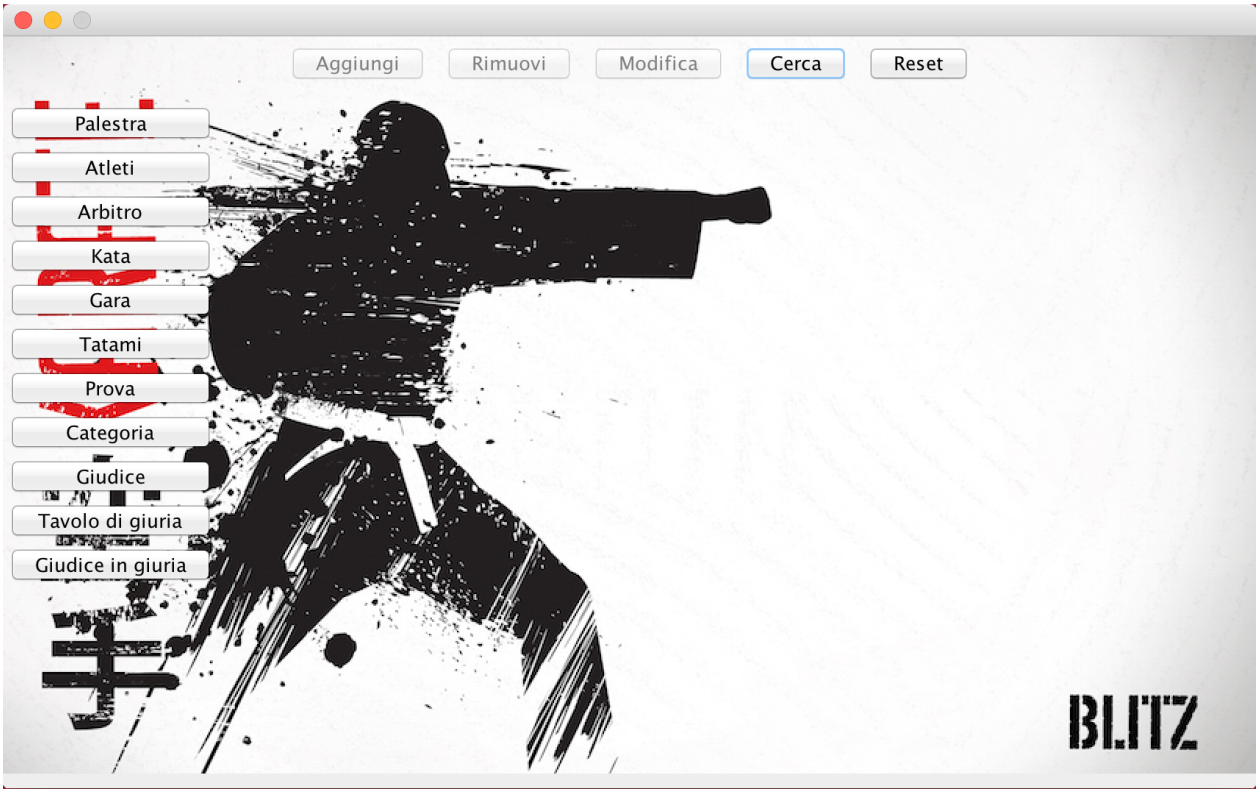
UPDATE gara SET data = *inputDate* WHERE nome_manifestazione = *inputCompetitionName* AND nomePalestra = *inputGymName* AND cittàPalestra = *inputGymCity*

INTERFACCIA GRAFICA

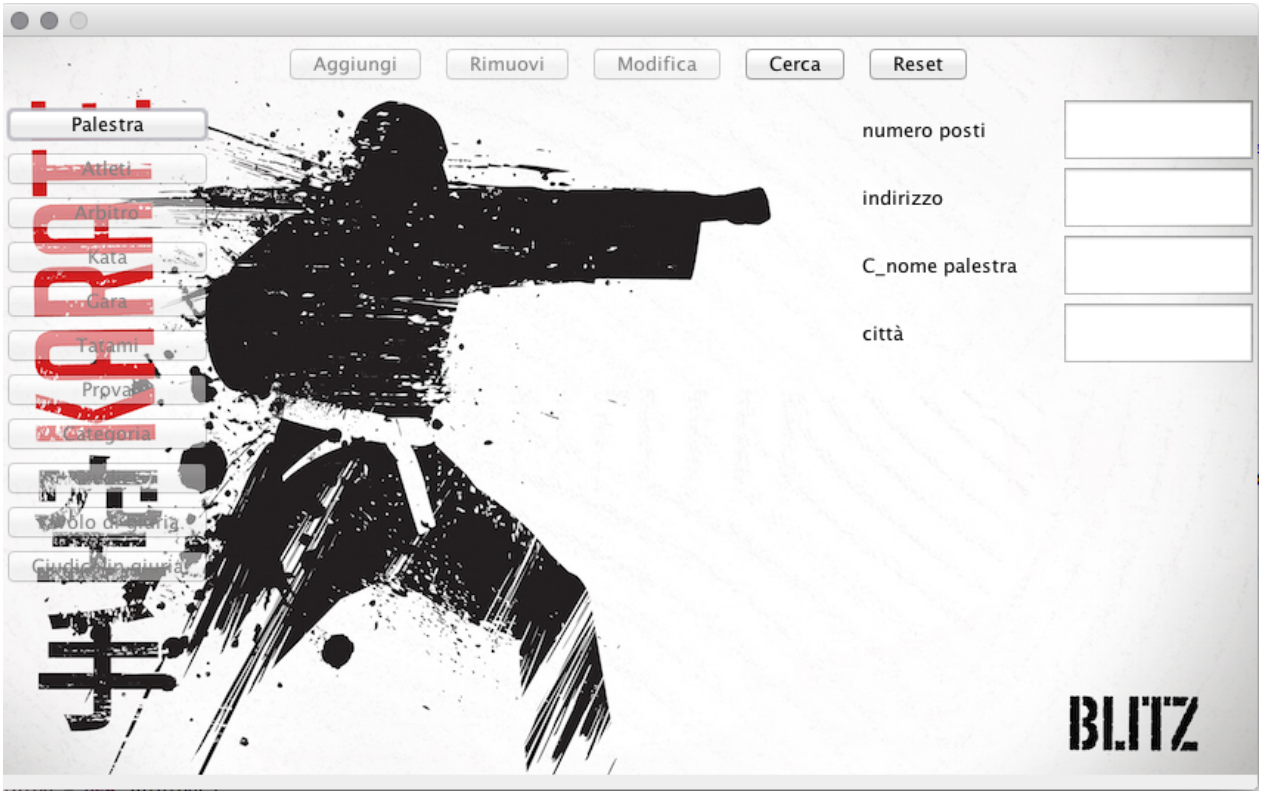
L'interfaccia utente consiste in una GUI realizzata in linguaggio Java. Ad un primo avvio dell'applicativo si può visualizzare questa finestra:



Premendo il pulsante "user" possiamo entrare ed eseguire solamente operazioni di ricerca, in quanto non avremo i permessi per modificare in alcun modo il Database.
Se clicchiamo il pulsante "admin" ci verrà richiesto di inserire la Password che abbiamo scelto. Molto banalmente, la password è "root". Qualora sbagliassimo a digitare il sistema non ci consentirà di entrare.
Una volta eseguito l'accesso avremo di fronte una situazione simile:



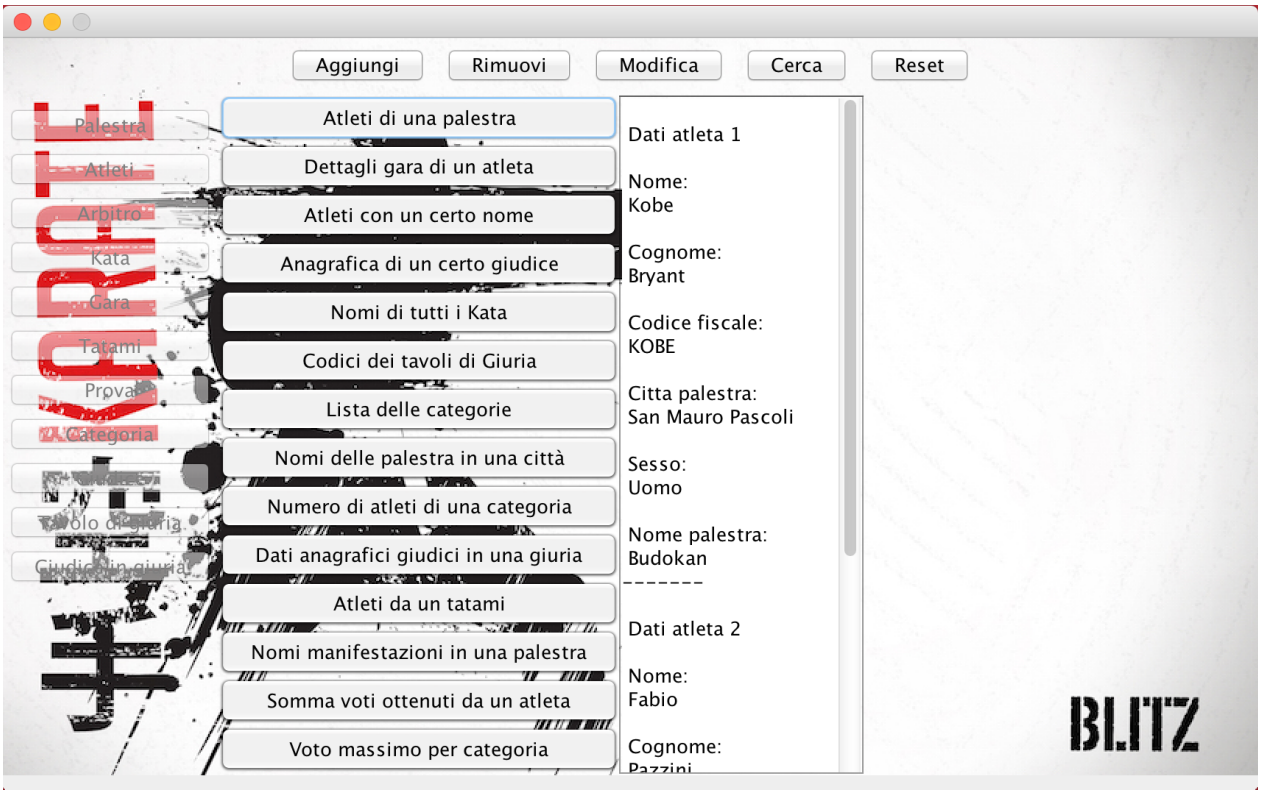
In alto troviamo quattro pulsanti per la modifica del Database ed uno per il reset della GUI: “Aggiungi, “Modifica”, “Elimina”, “Cerca”, “Reset”. Gli ultimi due sono sempre attivi, gli altri invece no. Per poter attivare i restanti dobbiamo selezionare uno dei pulsanti incolonnati a sinistra, che indicano le tabelle in cui andremo ad inserire i record. Selezionando una tabella, a destra appariranno una serie di campi affiancati dalla rispettiva etichetta. Supponiamo di cliccare il pulsante Palestra, che ci permette di aggiungere una nuova palestra al Database, e si avrà questa situazione:



Una volta riempiti i campi potremo cliccare Aggiungi ed aggiungere il nuovo record alla Base di Dati. Qualora volessimo eliminare un record dal Database basterà riempire il solo campo la cui etichetta è affiancata da una “C_” e premere il pulsante Elimina in alto. La funzione del pulsante Modifica invece, che ci consente di modificare uno specifico record, è abilitata ed implementata solo per alcune tabelle.

Premendo il pulsante Cerca abbiamo modo di eseguire ricerche nel Database. Ad esempio, supponiamo di voler ricercare tutti gli atleti provenienti da una specifica palestra: clicchiamo il pulsante Cerca, ci apparirà la lista di possibili interrogazioni alla Base di Dati, e selezioniamo la prima: si aprirà una piccola finestra che ci chiede di inserire il nome e la città della palestra, una volta riempiti i campi premiamo Enter e nella

GUI si visualizzerà un pannello scorrevole in cui troveremo tutti i dati ricercati; l'immagine mostra il risultato.



Il pulsante Reset, infine, resetta tutti i pulsanti e la situazione iniziale (ovvero la situazione della GUI al primo avvio).