Basi di Dati

Progetto sulla gestione delle competizioni di calisthenics

Docente :

Alfredo Pulvirenti

Studente :

Antonio Finocchiaro

1000006272

2022 / 2023

Contents

Part I
Introduzione

1 Panoramica concettuale

Il progetto proposto consiste in una base di dati per gestire le informazioni relative alle competizioni di calisthenics.

Il calisthenics è una disciplina a corpo libero complementare alla ginnastica artistica; alcuni degli attrezzi utilizzati in questa disciplina sono: *parallele, anelli, sbarre etc....* Essa sta prendendo sempre più piede in Italia e vi è un numero sempre maggiore di palestre adibite alla pratica.

Il seguente progetto si compone di varie parti :

- **Progettazione concettuale** in cui si individuano le funzionalità del sistema e si produce una descrizione dei dati coinvolti, delle relazioni tra di essi.

  In questa fase si danno dei dettagli sulla descrizione delle entità partecipanti e si eliminano delle ambiguità come omonimie, sinonimie o conflitti di descrizione.

  Si stilerà un glossario di termini, e verrà raffigurato un **Diagramma Entità-Relazione** per la base in esame con le dovute riflessioni come: la strategia di costruzione di esso, l'analisi di correttezza e completezza e altri dettagli di questa fase.

- **Progettazione logica** Questa è la fase successiva nella quale la struttura concettuale del database viene tradotta in un formato che può essere utilizzato dal DBMS scelto.

  In questa fase vengono specificati i dettagli di implementazione del database, come la struttura delle tabelle e la definizione delle chiavi primarie ed esterne.

Part II
Progettazione concettuale

2 Descrizione basi di dati

La base di dati in oggetto gestisce l'organizzazione delle competizioni di Calisthenics.

- **Persona**: rappresenta un individuo. Ha un codice fiscale univoco e altre informazioni come nome, cognome, telefono, residenza e CAP.

  Diverse persone ricoprono diversi ruoli, come atleti, allenatori, giudici e organizzatori.

- **Atleta**: rappresenta un partecipante alle gare. Possiede un attributo per identificare l'allenatore che eventualmente può avere valore NULL, se si tratta di un atleta autodidatta. Ha un id atleta, oltre a informazioni come genere, età, peso e altezza.

- **Allenatore**: rappresenta un allenatore. Ha un identificativo per sè stesso. Ha inoltre un riferimento al nome della palestra in cui lavora.

- **OrganizzatoreEvento**: rappresenta un organizzatore di eventi. Ha un riferimento alla persona e all'associazione di cui fa parte. Ha inoltre il numero di eventi organizzati, un campo su cui poter effettuare successivamente analisi di ridondanza.

- **Giudice**: rappresenta un professionista che giudica le gare. Ha un riferimento alla persona e un'informazione riguardo l'organizzazione di cui fa parte, che può differire da quella dell'organizzatore.

- **Spettatore**: rappresenta uno spettatore. Ha un identificativo per la persona e un attributo per la sua data di nascita.

- **Categoria**: rappresenta una categoria di atleti in una competizione di Calisthenics. Ha un id univoco e altre informazioni come il nome della categoria, il peso minimo e il peso massimo. Il nome della categoria può avere i seguenti valori :

  - **Beginner**
  - **Junior**
  - **Senior**

Ognuna di queste categorie può avere tre differenti sottocategorie, basate sul peso dell'atleta :

  - Beginner 50-69Kg / Beginner 70-89Kg / Beginner 90-109Kg
  - Junior 50-69Kg / Junior 70-89Kg / Junior 90-109Kg
  - Senior 50-69Kg / Senior 70-89Kg / Senior 90-109Kg

  L'id della categoria identifica tutte le combinazione sopraelencate.

- **Competizione**: rappresenta un determinato evento. Ha un id univoco e un riferimento alla struttura dove si svolge (tramite un campo "idStruttura") e alla data in cui si svolge. Una competizione può essere organizzata da molteplici persone.

- **TipologiaGara**: rappresenta una tipologia di gara presente in una competizione. Ha un id univoco, un genere (M/F) per suddividere le gare maschili da quelle femminili ed un riferimento alla categoria (tramite un campo "idCategoria"). Ha inoltre il nome della tipologia con i seguenti valori:

  - **Skills**
  - **Strength**
  - **Endurance**

  Ognuna di queste categorie può avere due differenti sottocategorie, basate sul genere dell'atleta :

  - Skills M / Skills F
  - Endurance M / Endurance F
  - Strength M / Strength F

- **Gara**: rappresenta una gara di Calisthenics in una competizione. Ha un id univoco e un riferimento alla competizione in cui è presente e uno alla tipologia di gara a cui appartiene (tramite il campo "idTipologia") e all'atleta che partecipa (tramite il campo "idAtleta"). Ha inoltre il numero di partecipanti alla gara.

- **Posizionamento**: rappresenta il posizionamento di un atleta in una gara. Ha un riferimento alla gara (tramite il campo "idGara") e all'atleta (tramite il campo "idAtleta"). Ha inoltre un campo "posizione" che indica il posizionamento dell'atleta nella gara e un punteggio ad egli associato.

- **Regolamento**: rappresenta il regolamento di una tipologia di gara. È identificato da una categoria, una tipologia e un anno in cui è stato stilato.

- **Struttura**: rappresenta una struttura che ospita competizioni di sollevamento pesi. Ha un id univoco e il nome della struttura, il luogo in cui si trova e il numero di competizioni ospitate.

Alcuni riflessioni in merito sono :

- Un atleta appartiene ad una categoria, crescendo (in termini di prestazioni e di peso) egli può appartenere ad una categoria differente. Ciò è possibile considerando le chiavi delle tabelle in maniera corretta in fase di progettazione.
- Una tipologia di gara è identificata dal suo id. Essa abbinata alla categoria degli atleti definisce una gara.
- Alcuni attributi come il numero di eventi organizzati da un organizzatore o il numero di partecipanti ad una gara, potranno essere analizzati successivamente dopo aver stimato i volumi, per definire se conviene o meno tenerli all'interno della progettazione.
- C'è il rischio che per un errore di inserimento, a più di un atleta venga assegnata la medesima posizione nella stessa gara, si vedrà in seguito la costruzione di un trigger per evitare questo errato inserimento.

3 Glossario dei termini

In questa sezione si continua con l'analisi dei requisiti attraverso la costruzione del **glossario dei termini**.
Esso serve ad evidenziare concetti espressi nei requisiti, all'interno di una tabella per:

- Individuare **omonimi** e **sinonimi**.
- Fornire una piccola **descrizione** sulle entità.
- Rendere esplicito il **collegamento** tra esse.

Esso è il secondo passo della progettazione concettuale e fornisce l'input per ciò che sarà il diagramma entità relazione.
Ecco in seguito, il glossario dei termini per la base di dati in esame:

Termini	***Descrizione***	***Sinonimi***	***Collegamenti***
Persona	Informazioni personali sulle persone che ricoprono i vari ruoli, come gli allenatori, gli atleti, i giudici.	Individuo, Soggetto	Atleta, Allenatore, OrganizzatoreEvento, Giudice
Atleta	Partecipante alle gare, avente determinati attributi come : altezza, peso, età, appartenenti ad una categoria, allenati da un Allenatore.	Partecipante	Persona, Atleta, Allenatore
Allenatore	Professionista che allena uno o più atleti, lavora in una determinata struttura.	Coach, Trainer	Persona, Atleta
Spettatore	Persona che appartiene al pubblico di una compezione.	Pubblico	Persona, Competizione
OrganizzatoreEvento	Persona che ha creato l'evento e ne ha diretto tutti gli aspetti organizzativi.	Coordinatore, Direttore	Persona, Competizione
Giudice	Professionista che si occupa di arbitrare i vari set degli atleti in gara.	Arbitro	Persona, Gara
Categoria	Insieme di atleti che soddisfano delle determinate caratteristiche in base anche al loro livello atletico.	Classe, Divisione	Atleta
Competizione	Evento organizzato da un organizzatore, contenente diverse gare, atleti, allenatori ecc...	Evento	OrganizzatoreEvento, Struttura
TipologiaGara	Descrizione sulle possibili tipologie di gara, generalmente contenente tre valori.	Genere, Ramo	Gara, Categoria, Atleta
Gara	Sottoinsieme della competizione, all'interno di quest'ultima vi sono diverse gare, suddivise in diverse tipologie.	Sfida, Incontro	TipologiaGara, Competizione, Atleta
Posizionamento	Posizione ottenuta da un determinato atleta, in una gara e il punteggio ad egli associato.	Piazzamento	Gara, Atleta
Regolamento	Descrive le regole di una specifica tipologia di gare.	Norma, Legge	TipologiaGara, Categoria
Struttura	Luogo ospitante una competizione, può essere all'aperto o al coperto.	Palestra, Area attrezzata	Competizione

4 Dettagli sui dati

Adesso, si esplicitano le varie entità partecipanti, dando per ognuna delle specifiche ed esplicitando le connessioni tra di esse:

Una **Persona** è identificata da un codice fiscale, può svolgere il ruolo di atleta, allenatore, organizzatore o giudice. Tra gli attributi vi è un campo numerico contenente il numero di telefono per garantirne la tracciabilità.

Un **Atleta** è identificato da un id, ha un identificatore del suo allentore e contiene delle informazioni utili per l'inserimento all'interno di una categoria/tipologia di gara.

Ogni atleta può avere al più un allenatore.

Esso può far parte di categoria differenti in archi temporali differenti, può competere in diversi eventi, in più gare di tipologie diverse e può ottenere in esse, diversi piazzamenti.

Un **allenatore** possiede almeno un allievo e ha una palestra dove lavora riferimento.
Un **OrganizzatoreEvento** è una persona che ha organizzato almeno un evento, con una sua associazione.

Un **Giudice** è una persona che può giudicare una o più gare, anche all'interno della stessa competizione, ma non può *naturalmente* giudicare più volte la stessa identica gara, appartiene ad un'organizzazione.

Uno **Spettatore** è una persona che partecipa passivamente ad una competizione, possiede un attributo per ricavare la sua età.

Una **Categoria** ha all'interno delle classi di atleti, si differiscono principalmente i: Beginners/Junior/Senior e come detto in precedenza, ciascuna di essa può avere 3 diverse sottocategorie di peso.

Una **Competizione** rappresenta l'evento, organizzato da un organizzatore; in una determinata data.

Ogni **TipologiaGara** rappresenta i diversi rami di Calisthenics per cui un atleta gareggia. Una tipologia comprende un identificativo, il genere (M/F) e il suo nome.

Una **Gara** è contenuta all'interno di una competizione, vi possono essere più gare presenti all'interno dello stesso evento; è identificata da un id suo e come attributi sono presenti gli identificatori della categoria e della tipologia, per avere una descrizione dettagliata su una determinata gara.

Inoltre all'interno vi sono tutti gli atleti partecipanti e il numero totale di essi.

Ogni **Posizionamento** è il piazzamento ottenuto da un atleta, in una specifica gara e il suo punteggio ottenuto.

Un atleta può apparire più volte in gare differenti.

Un **Regolamento** contiene i documenti che contengono i punteggi, l'elenco di skills, i vari circuiti ecc..

Vi è un regolamento per ogni tipologia di gara e categorie di atleti. Il regolamento viene ripubblicato ogni anno, con eventuali modifiche. Infine una **Struttura** appare se ha ospitato almeno una competizione.

5 Diagramma Entità-Relazione

Vi sono due approcci per la costruzione di un modello ER:

1. Strategia : **Top-down**

   Esso parte dalle entità più generali e si sposta verso quelle più specifiche. In questo modo, si inizia definendo le entità di alto livello, quindi si scompongono in entità più specifiche man mano che si scende nei dettagli. Ad esempio, si potrebbe iniziare definendo un'entità "Persona" e poi scomporla in entità più specifiche come "Atleta" e "Allenatore".

2. Strategia : **Bottom-up**

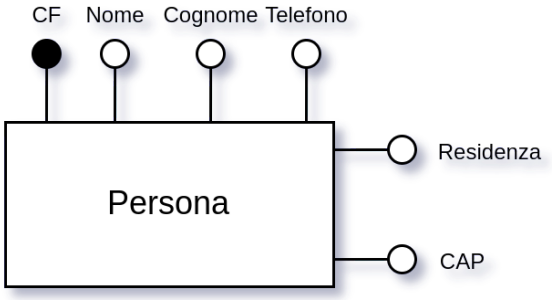
   In questo caso invece, si parte dalle entità più specifiche e si sposta verso quelle più generali. In questo modo, si inizia definendo le entità di basso livello, quindi si raggruppano in entità più generali man mano che si sale nei dettagli. Ad esempio, si potrebbero definire le entità "Atleta" e "Allenatore" e poi raggrupparle in un'entità più generale come "Persona".

In questo progetto è stato stabilito un criterio **top-down**.

Si inizia subito mostrando i vari passi verso il diagramma finale. Prima di tutto, definiamo le entità, rappresentando per ciascuna di essa, le chiavi e i vari attributi :

Persona

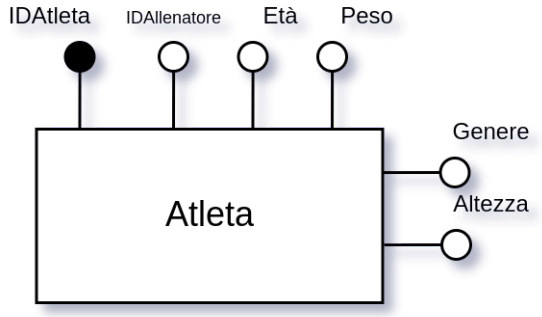
Dall'astrazione persona, ricaviamo i seguenti attributi.



- **Codice Fiscale**
- Nome
- Cognome
- Telefono
- Residenza
- CAP

Atleta

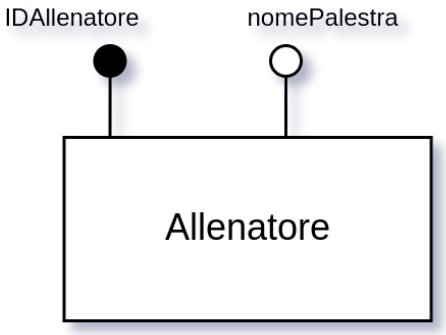
Dall'astrazione atleta, ricaviamo i seguenti attributi.



- **IDAtleta**
- IDAllenatore
- Genere
- Età
- Peso
- Altezza

Allenatore

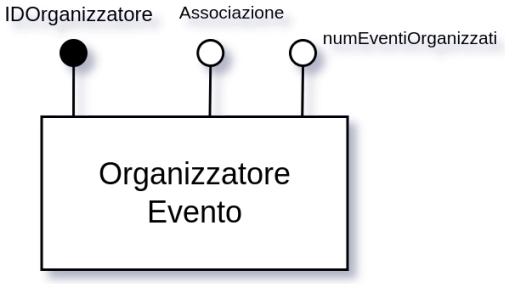
Dall'astrazione allenatore, ricaviamo i seguenti attributi.



- **IDAllenatore**
- nomePalestra

OrganizzatoreEvento

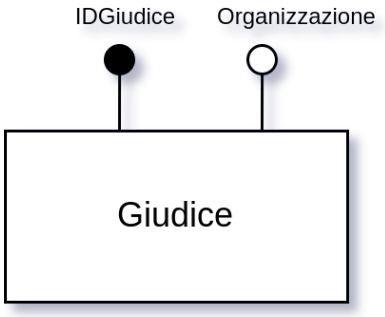
Dall'astrazione OrganizzatoreEvento, ricaviamo i seguenti attributi.



- **IDOrganizzatore**
- Associazione
- numEventiOrganizzati

Giudice

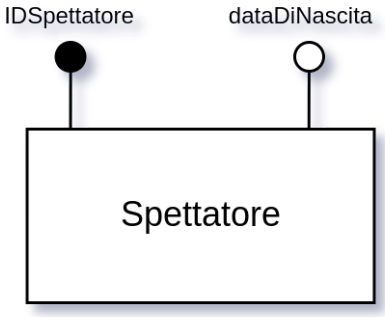
Dall'astrazione Giudice, ricaviamo i seguenti attributi.



- **IDGiudice**
- Organizzazione

Spettatore

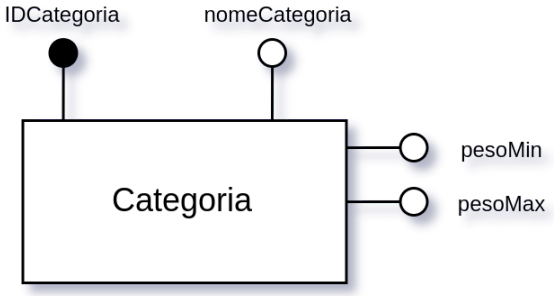
Dall'astrazione Spettatore, ricaviamo i seguenti attributi.



- **IDSpettatore**
- dataDiNascita

Categoria

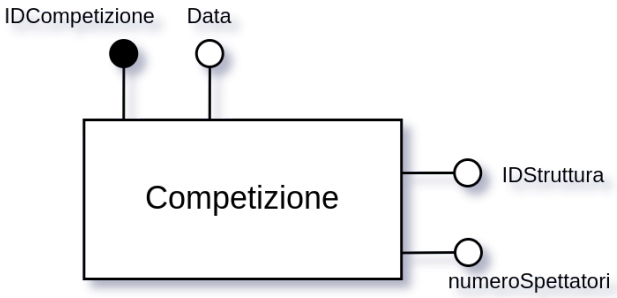
Dall'astrazione Categoria, ricaviamo i seguenti attributi.



- **IDCategoria**
- nomeCategoria
- pesoMin
- pesoMax

Competizione

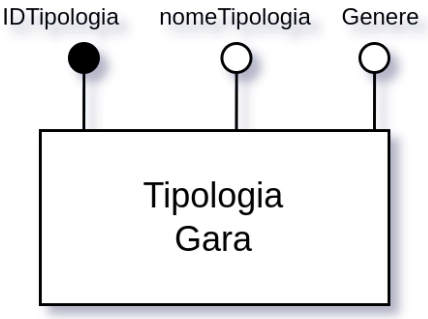
Dall'astrazione Competizione, ricaviamo i seguenti attributi.



- **IDCompetizione**
- Data
- IDStruttura
- numeroSpettatori

TipologiaGara

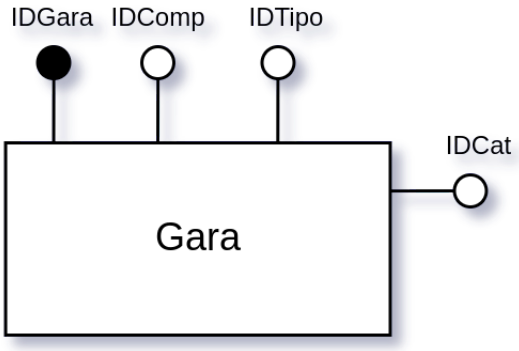
Dall'astrazione TipologiaGara, ricaviamo i seguenti attributi.



- **IDTipologia**
- Genere
- nomeTipologia

Gara

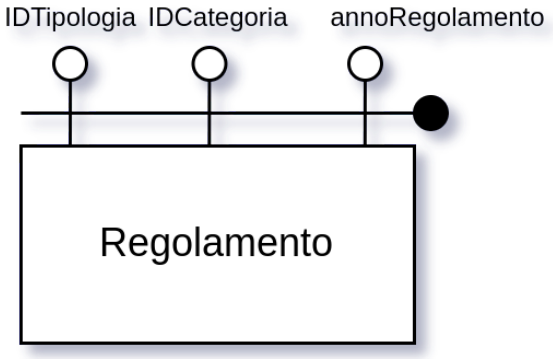
Dall'astrazione Gara, ricaviamo i seguenti attributi.



- **IDGara**
- IDCompetizione
- IDTipologia
- IDCategoria

Regolamento

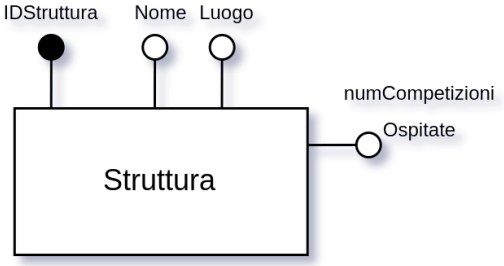
Dall'astrazione Regolamento, ricaviamo i seguenti attributi.



- **IDTipologia**
- **IDCategoria**
- **annoRegolamento**

Struttura

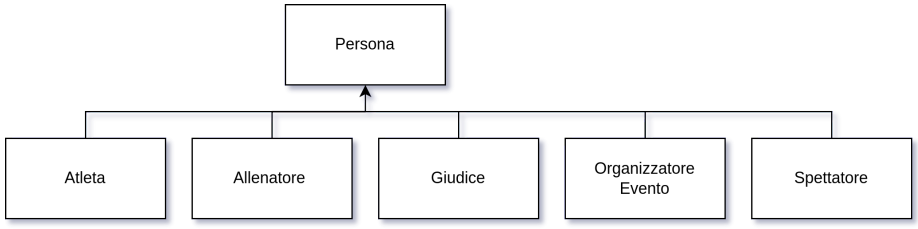
Dall'astrazione Struttura, ricaviamo i seguenti attributi.



- **IDStruttura**
- Nome
- Luogo
- numCompetizioniOspitate

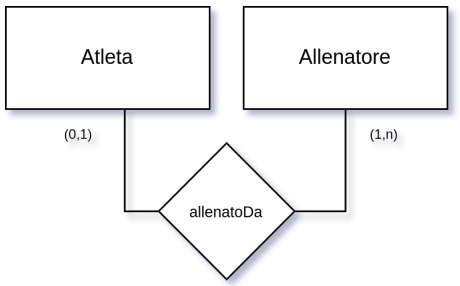
Una volta ottenute le entità andiamo a definire le relazioni che vi sono tra di esse.

Una prima relazione è una gerarchia e si ha tra Persona e Atleta, Giudice, Organizzatore, Allenatore e Spettatore.



La gerarchia è totale (ciò significa che ogni istanza dell'entità padre deve necessariamente far parte di una delle entità figlie) e non esclusiva (ogni istanza del padre può far parte di più di un'entità figlie).

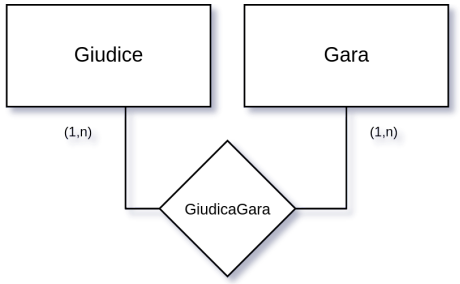
Vi è inoltre una relazione di associazione tra Atleta e Allenatore in quanto il primo, **è allenato** dal secondo.



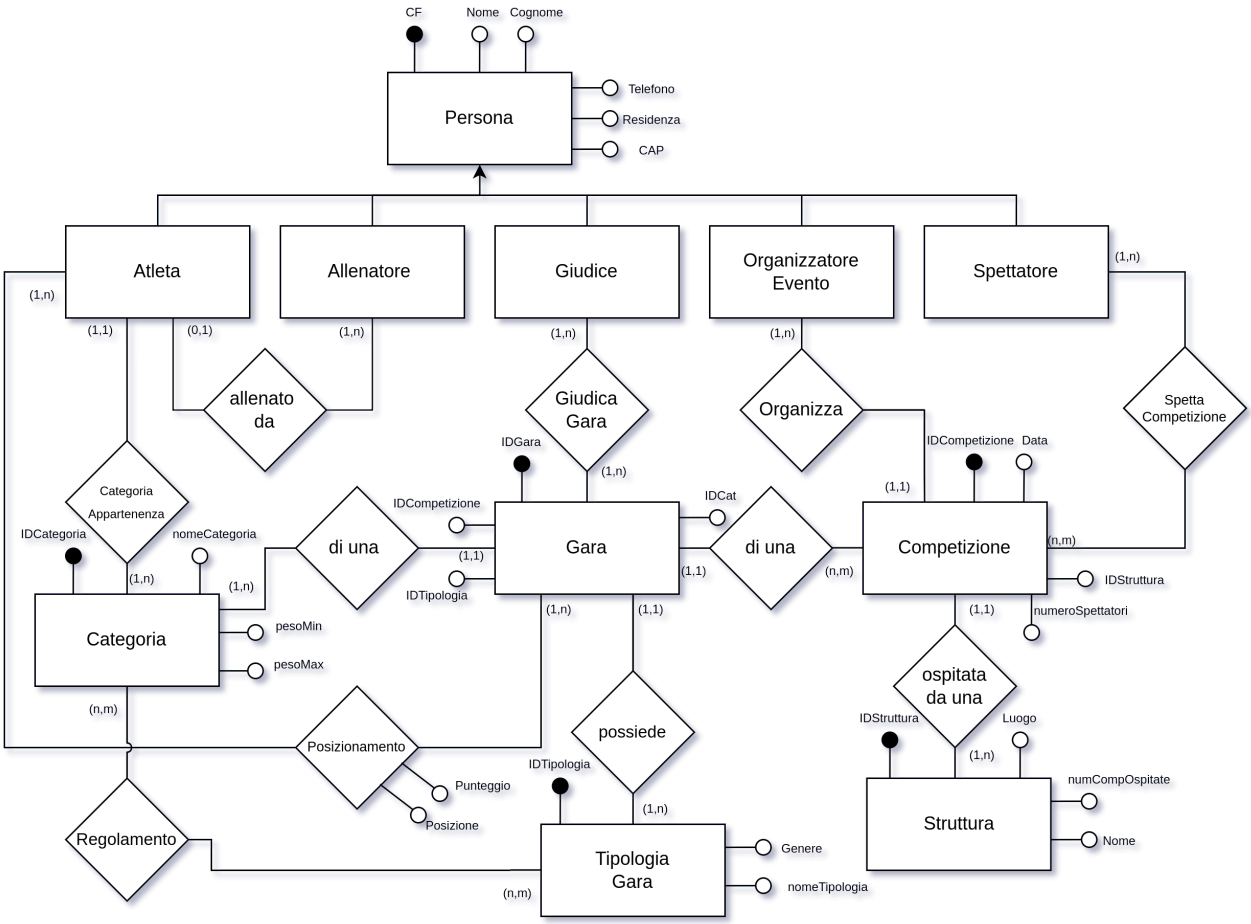
Vi sono tante altre relazioni tre le varie entità, ad esempio il posizionamento che ottiene un atleta dopo aver gareggiato in una gara.

In questa, l'associazione eredita le chiavi delle due entità e al suo interno si definisce l'attributo Posizione.

Un altro esempio è l'associazione che si ha tra l'entità Giudice e Gara.



Per motivi di praticità, è stato stilato un Diagramma intermedio, con tutte le relazioni incluse. Esso tuttavia non è pronto alla conversione in Progettazione logica ma possiede comunque un primo livello di significato.


CF
Nome
Cognome
Persona
Telefono
Residenza
CAP
Atleta
Allenatore
Giudice
Organizzatore Evento
Spettatore
(1,n)
(1,1)
(0,1)
(1,n)
(1,n)
(1,n)
(1,n)
allenato da
Giudica Gara
Organizza
Spetta Competizione
IDGara
(1,n)
IDCompetizione
Data
(1,1)
Categoria Appartenenza
IDCategoria
nomeCategoria
(1,n)
(1,n)
di una
IDCompetizione
(1,1)
IDTipologia
Gara
IDCat
(1,1)
di una
(n,m)
Competizione
(n,m)
IDStruttura
numeroSpettatori
(1,1)
Categoria
pesoMin
pesoMax
(n,m)
(1,n)
(1,1)
ospitata da una
IDStruttura
Luogo
(1,n)
Posizionamento
Punteggio
Posizione
possiede
IDTipologia
(1,n)
Struttura
numCompOspitate
Nome
Regolamento
Tipologia Gara
Genere
nomeTipologia
(n,m)

Part III
Progettazione Logica

6 Ristrutturazione del diagramma intermedio

Lo schema attuale non può essere convertito in schema logico poichè vi è presente una gerarchia. Essa può essere risolta attraverso due criteri:

- **Collasso verso l'alto**
- **Collasso verso il basso**

Poichè il collasso verso l'alto ha come conseguenza un'unica relazione con la quale stabilire a quale sottoclasse fare riferimento (solitamente viene introdotto un selettore); ciò nel caso in questione introdurrebbe tanti valori nulli poichè i valori obbligatori delle classi figlie, diventerebbero opzionali.

Pertanto, si opta per un collasso verso il basso in cui si elimina l'entità padre trasferendone gli attributi su tutte le entità figlie.

Questa soluzione introdurrebbe un'enorme quantità di attributi all'interno di ogni entità, pertanto non conviene considerare ultimato il diagramma a seguito di questa trasformazione.

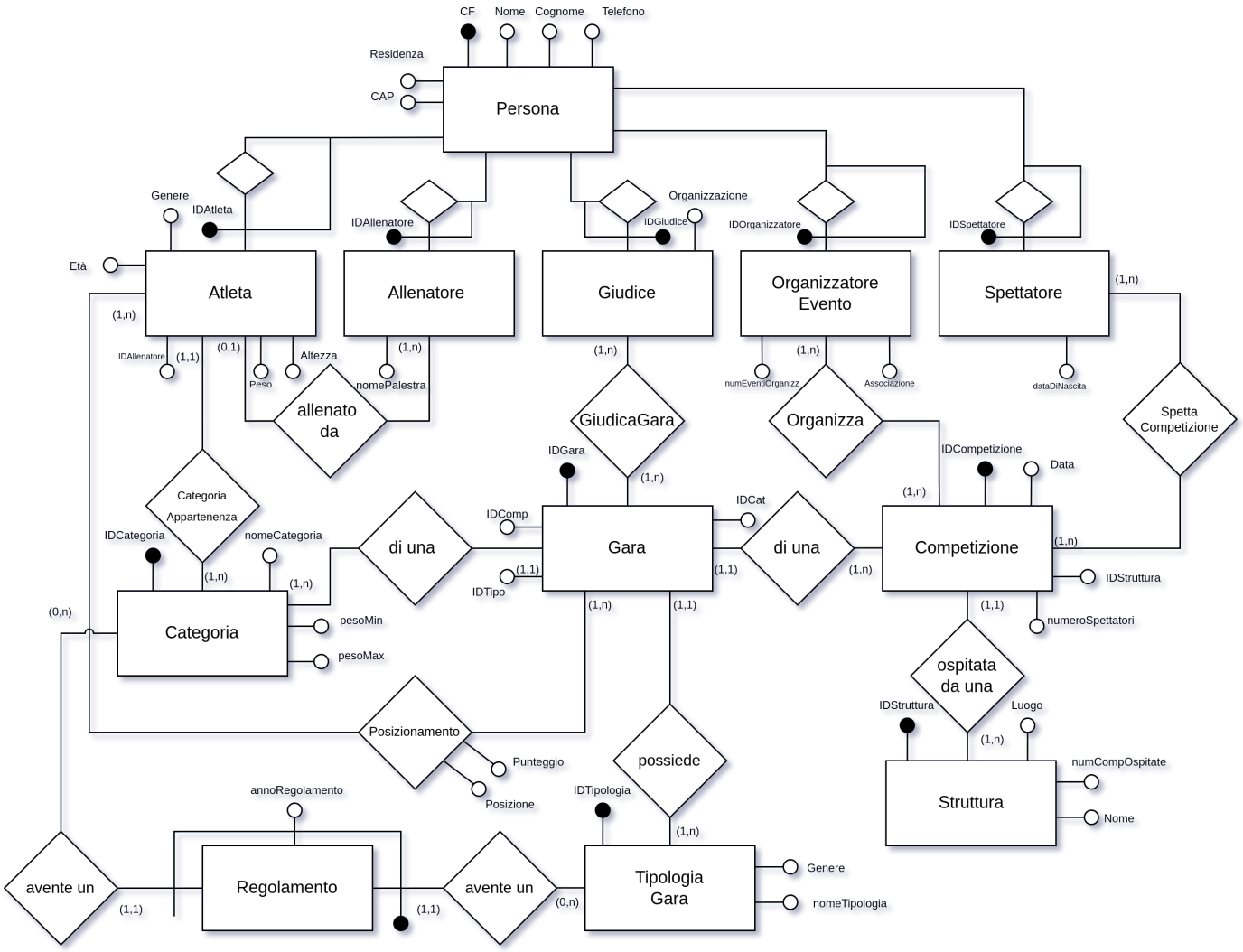
Si utilizza un **partizionamento verticale**. Esso è una tecnica che consiste nel suddividere i dati di una tabella in più partizioni, dove ogni partizione contiene solo una parte delle colonne della tabella originale.

Lo si applicherà alle entità Atleta, Allenatore, Giudice, OrganizzatioreEvento e Spettatore; partizionando le loro informazioni personali in una tabella Persona.

Il partizionamento verticale viene solitamente utilizzato per migliorare le prestazioni delle query in un database, poiché permette di ridurre il numero di colonne da caricare in memoria durante l'esecuzione di una query. Inoltre, il partizionamento verticale può essere utile per gestire meglio la scalabilità di un database, poiché permette di distribuire i dati su più server o su più dischi.

L'associazione regolamento, avente cardinalità (n,m) con Categoria e TipologiaGara è stata trasformata in un'entità debole; la chiave composta è formata dai due identificativi delle entità adiacenti e dall'attributo annoRegolamento.

Ecco lo schema a seguito della trasformazione :



7 Dizionario dei dati

Prima di passare alla traduzione in modello logico, vengono visualizzate le due tabelle valide come **Dizionario delle entità** e **Dizionario delle relazioni** inerenti all'ultimo schema proposto:

Ecco il dizionario delle entità :

Entità	***Descrizione***	***Attributi***	***Chiave***
Persona	Individuo ricoprente un singolo ruolo	Nome, Cognome, Telefono, Residenza, CAP	CF
Atleta	Professionista della disciplina	IDAllenatore, Genere, Età, Peso, Altezza	IDAtleta
Allenatore	Insegnante della disciplina	nomePalestra	IDAllenatore
Giudice	Arbitro di gara	Organizzazione	IDGiudice
OrganizzatoreEvento	Direttore di una competizione	numEventiOrganizzati, Associazione	IDOrganizzatore
Spettatore	Individuo che ricopre un ruolo passivo in una competizione	dataDiNascita	IDSpettatore
Categoria	Classe di suddivisione degli atleti	nomeCategoria, pesoMin, pesoMax	IDCategoria
Competizione	Evento	Data, IDStruttura, numeroSpettatori	IDCompetizione
TipologiaGara	Classe di suddivisione delle gare	Genere, nomeTipologia	IDTipologia
Gara	Performance di atleti di una tipologia e categoria	IDCompetizione, IDTipologia, IDCategoria	IDGara
Struttura	Luogo adibito ad ospitare gli eventi	Nome, Luogo, competizioniOspitate	IDStruttura
Regolamento	Documento contenente informazioni sulle regole di una determinata tipologia di gara	-	IDTipologia, IDCategoria, annoRegolamento

Ed ecco il dizionario delle relazioni in cui l'unica relazione ad avere un attributo è Piazzamento. In essa l'attributo Posizione è fondamentale per risalire al numero in classifica dell'atleta.

Tutte le relazioni nella tabella hanno due entità coinvolte.

Relazione	***Descrizione***	***Entità coinvolte***	***Attributi***
allenatoDa	Atleta allenato da un allenatore	Atleta, Allenatore	-
CategoriaAppartenenza	Categoria a cui appartiene un atleta	Atleta, Categoria	-
diUna	Categoria di una gara	Categoria, Gara	-
diUna	Gara presente in una competizione	Gara, Competizione	-
ospitataDaUna	Competizione tenuta in una struttura	Competizione, Struttura	-
Possiede	Tipologia posseduta da una gara	Gara, TipologiaGara	-
Organizza	Un organizzatore crea un evento	OrganizzatoreEvento, Competizione	-
SpettaCompetizione	Presenza di una persona tra gli spalti	Spettatore, Competizione	-
Posizionamento	Posizione ottenuta da un atleta in una gara e il suo punteggio	Atleta, Gara	Posizione, Punteggio
aventeUn	Tipologia Gara avente un Regolamento per essa	Regolamento, TipologiaGara	-
aventeUn	Categoria avente un Regolamento per essa	Regolamento, Categoria	-
GiudicaGara	Lavoro svolto da un giudice in una gara	Giudice, Gara	-

8 Analisi delle ridondanze

Nel diagramma soprastante, sono presenti alcune entità con degli attributi da analizzare.

L'analisi che si effettuerà ha il ruolo di stabilire se in base a dei volumi stimati, è corretto mantenere la ridondanza o meno, in termini di Operazioni/ArcoTemporale.

Prima di procedere, vengono allegate due tabella contenente i volumi e le operazioni effettuabili.

I seguenti volumi sono stati stimati e in fase di progettazione fisica possono chiaramente variare in base alle esigenze.

Concetto	***Tipo***	***Volume***
Persona	E	115.600
Atleta	E	10.000
Allenatore	E	500
Spettatore	E	100.000
Organizzatore	E	100
Giudice	E	5.000
Categoria	E	9
Competizione	E	1.000
Tipologia Gara	E	6
Gara	E	20.000
Regolamento	E	60
Struttura	E	500
AllenatoDa	R	20
GiudicaGara	R	4
Organizza	R	10
SpettaCompetizione	R	100
Piazzamento	R	20.000
CategoriaAppartenenza	R	1110
GareCompetizione	R	20
CategorieGare	R	3330

Ed in seguito un insieme di operazioni che potrebbero essere svolte :

Operazione	*Frequenza*	*Tipo*
Visualizzazione del nome degli atleti in nelle gare di Skills tenute nelle competizioni nella struttura di nome "FitnessAndFight"	20/w	I
Visualizzazione del numero di eventi ospitati da una struttura	50/w	I
Inserimento di una competizione in una struttura	40/w	I
Visualizzazione dell'allenatore con più allievi	2/m	B
Inserimento di un nuovo spettatore ad un evento	50/w	I
Visualizzazione di tutti i posizionamenti ottenuti dall'atleta "Emanuele Majeli" in tutte le gare effettuate	500/w	I
Visualizzazione il numero di eventi organizzati da un organizzatore	5/w	I
Inserimento di un nuovo evento creato da un organizzatore	4/w	I
Visualizzazione del numero di spettatori in un evento	100/w	I
Calcolo, per ogni categoria di atleti, l'atleta con peso maggiore a quello medio	2/m	B
Inserimento di un nuovo regolamento	1/a	B
Visualizzazione del nome e cognome dell'atleta che ha gareggiato in almeno una gara di Endurance e il quale nome inizi per M e del suo allenatore	5/m	B

Grazie ad un'attenta analisi iniziale che ha portato a mantenere nelle entità, attributi essenziali di carattere semantico; si è limitato l'insieme di attributi da analizzare. Le operazioni osservate saranno prelevate dalla Tabella delle operazioni.

Si inizia lo studio dall'attributo **numeroEventiOrganizzati** da un organizzatore.

I seguenti volumi sono stimati :

- 100 organizzatori
- 1000 competizioni

In media dunque, 1 organizzatore, crea 10 eventi.

Operazione 7 : contare il numero di eventi organizzati da un organizzatore, 5 volte a settimana.

- **Analisi con la ridondanza** : 1 Lettura su OrganizzatoreEvento$\times$ 5 = 5 OP/w.
- **Analisi senza la ridondanza** : 1 Lettura su Organizza $\times$ 10 $\times$ 5 = 50L = 50 OP/w.

Operazione 8 : inserire un nuovo evento all'interno dell'entità OrganizzatoreEvento, 4 volte a settimana.

- **Analisi con la ridondanza** : 1S OrganizzatoreEvento + 1S Organizza = 4L $\times$ 4 = 16 OP/w
- **Analisi senza la ridondanza** : 1S Organizza = 2L $\times$ 4 = 8 OP/w

Con i dati presupposti si avrebbe, 17 OP/w in presenza della ridondanza, 18 OP/w in assenza di essa.
I due valori sono circa pari, tutta via sarebbe ottimo mantenerla poichè in presenza di una maggiore mole di dati, questa differenza potrebbe aumentare.

Una seconda analisi verrà effettuata sull'attributo **numeroSpettatori** all'interno dell'entità Competizione.
I seguenti volumi sono stimati :

- 1000 eventi
- 100000 spettatori

In media dunque, ad 1 evento, partecipano 1000 spettatori, i volumi sono maggiori rispetto alla precedente analisi.
Operazione 9 : contare il numero di spettatori presenti ad una competizione, 100 volte a settimana.

- **Analisi con la ridondanza** : 1 Lettura su Competizione. = 10 L $\times$ 100 = 1000 OP/w.
- **Analisi senza la ridondanza** : 1 Lettura su Partecipa $\times$ 1000 = 1000L $\times$ 100 = 100000 OP/w.

Operazione 5 : inserire un nuovo spettatore all'interno dell'entità Competizione, 50 volte a settimana.

- **Analisi con la ridondanza** : 1S Competizione + 1S Partecipa = 4L × 50 = 200 OP/w
- **Analisi senza la ridondanza** : 1S Organizza = 2L × 50 = 100 OP

Con i dati presupposti si avrebbe, 1200 OP/w in presenza della ridondanza, 100100 OP/w in assenza di essa.
Questo è un chiaro caso in cui conviene mantenere la ridondanza, la differenza è sostanziale.

Un'ultima analisi verrà condotta sull'attributo **numeroCompetizioniOspitate** dell'entità Struttura :
Si stimano stavolta, volumi leggermente differenti rispetto ai precedenti ricavati dagli schemi :

- 50 eventi
- 10 strutture

In media dunque, 1 struttura ospita 5 eventi.
Operazione 2 : contare il numero di competizioni ospitate da una struttura , 50 volte a settimana.

- **Analisi con la ridondanza** : 1 Lettura su Struttura. = 10 L × 50 = 10 OP/w.
- **Analisi senza la ridondanza** : 1 Lettura su Ospita × 5 = 5L × 50 = 250 OP/w.

Operazione 3 : inserire un nuova Competizione all'interno di una Struttura, 40 volte a settimana.

- **Analisi con la ridondanza** : 1S Struttura + 1S Ospita = 4L × 5 = 20L × 40 = 800 OP/w
- **Analisi senza la ridondanza** : 1S Organizza = 2L × 5 = 10 L × 40 = 400 OP/w

Con i volumi stimati si avrebbero 810 OP/w in presenza della ridondanza, 650 OP/w in assenza di essa.

In questo caso, a differenza dei precedenti è conveniente rimuovere la ridondanza dallo schema.

Sono state esaminate operazioni **interattive**, ovvero operazioni che vengono eseguite in modo interattivo, in risposta ad un input dell'utente.

Ad esempio, quando si utilizza un'interfaccia grafica per eseguire una query su un database, si sta eseguendo un'operazione interattiva.

Le interazioni **batch**, al contrario, sono quelle che vengono eseguite in modo automatico, senza intervento dell'utente.

Ad esempio, uno script che esegue una serie di operazioni su un database in modo automatico è un'interazione batch.

9 Mapping in modello logico

In questa sezione verrà effettuata la traduzione in schema logico.

Quest'operazione consiste nel tradurre il modello ER in un formato che può essere utilizzato per creare un database fisico. Nelle precedenti sezioni, sono già state fatte delle operazioni preliminari come :

- Lo schema concettuale è stato ristrutturato, eliminando la gerarchia presente
- E' stato stilato un dizionario dei dati, contenente le entità e le relazioni partecipanti
- Sono stati analizzati gli attributi che avrebbero potuto introdurre ridondanza.

Osservazione : nonostante secondo le analisi l'attributo numEventiOrganizzati non introduce un aumento delle operazioni, si è preferito rimuoverlo nella progettazione fisica per evitare di doverlo inserire nelle tabelle manualmente.

Il Modello entità relazione è stato tradotto nel seguente DBMS :

Persona(**CF**, Nome, Cognome, Telefono, Residenza, CAP)

Atleta(**IDAtleta**, IDAllenatore, Genere, Età, Peso, Altezza)

Allenatore(**IDAllenatore**, nomePalestra)

Spettatore(**IDSpettatore**, dataDiNascita)

SpettaCompetizione(**IDSpettatore, IDCompetizione**)

OrganizzatoreEvento(**IDOrganizzatore, Organizzazione**)

Organizza(**IDOrganizzatore, IDCompetizione**)

Giudice(**IDGiudice**, Organizzazione)

GiudicaGara(**IDGiudice, IDGara**)

Categoria(**IDCategoria**, nomeCategoria, pesoMin, pesoMax)

CategoriaAppartenza(**IDCategoria, IDAtleta**)

Competizione(**IDCompetizione**, Data, IDStruttura, numeroSpettatori)

TipologiaGara(**IDTipologia**, genere, nomeTipologia)

Gara(**IDGara**, IDCompetizione, IDTipologia, IDCategoria)

Posizionamento(**IDGara, IDAtleta**, Posizione, Punteggio)

Regolamento(**IDTipologia, IDCategoria, annoRegolamento**)

Struttura(**IDStruttura**, Nome, Luogo)

Part IV
Progettazione Fisica

La progettazione fisica di un database è il processo di creazione di una struttura fisica per archiviare e gestire i dati all'interno del database. Si tratta di un passaggio importante nella progettazione, poiché la struttura fisica del database influisce sulla velocità e sulla efficienza delle operazioni di accesso ai dati.

Verrà eseguita sul modello logico estratto precedentemente.

Il primo passo da effettuare è quello della creazione delle tabelle, siano esse entità o relazioni.

In seguito viene riportato il codice per la creazione di ognuna delle tabelle.

10 Creazione tabelle in SQL

Persona

```
1
2  CREATE TABLE `Persona` (
3    `CF` varchar(16) NOT NULL,
4    `Nome` varchar(20) NOT NULL,
5    `Cognome` varchar(20) NOT NULL,
6    `Telefono` int(10) NOT NULL,
7    `Residenza` varchar(40) NOT NULL,
8    `CAP` int(5) NOT NULL
9    ADD PRIMARY KEY (`CF`)
10 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
11
```

Atleta

```
1
2  CREATE TABLE `Atleta` (
3    `IDAtleta` varchar(16) NOT NULL,
4    `IDAllenatore` varchar(16) DEFAULT NULL,
5    `Genere` varchar(5) NOT NULL,
6    `Età` int(2) NOT NULL,
7    `Peso` int(3) NOT NULL,
8    `Altezza` int(3) NOT NULL
9  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
10
```

Allenatore

```
1
2  CREATE TABLE `Allenatore` (
3    `IDAllenatore` varchar(16) NOT NULL,
4    `nomePalestra` varchar(30) NOT NULL
5    ADD PRIMARY KEY (`IDAllenatore`),
6    ADD CONSTRAINT `Allenatore_ibfk_1` FOREIGN KEY (`IDAllenatore`) REFERENCES
   ↪  `Persona` (`CF`),
7  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
8
```

Spettatore

```
1
2  CREATE TABLE `Spettatore` (
3    `IDSpettatore` varchar(16) NOT NULL,
4    `dataDiNascita` date NOT NULL,
5    ADD PRIMARY KEY (`IDSpettatore`),
6    ADD CONSTRAINT `testt5` FOREIGN KEY (`IDSpettatore`) REFERENCES `Persona`
   ↪  (`CF`)
7  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
8
```

SpettaCompetizione

```
1
2  CREATE TABLE `SpettaCompetizione` (
3    `IDSpettatore` varchar(16) NOT NULL,
4    `IDCompetizione` int(20) NOT NULL,
5    ADD PRIMARY KEY (`IDSpettatore`,`IDCompetizione`),
6      ADD CONSTRAINT `fkcompetition` FOREIGN KEY (`IDCompetizione`) REFERENCES
   ↪  `Competizione` (`IDCompetizione`),
7    ADD CONSTRAINT `fkspet` FOREIGN KEY (`IDSpettatore`) REFERENCES `Spettatore`
   ↪  (`IDSpettatore`)
8  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
9
```

OrganizzatoreEvento

```
1
2  CREATE TABLE `OrganizzatoreEvento` (
3    `IDOrganizzatore` varchar(16) NOT NULL,
4    `Associazione` varchar(50) DEFAULT NULL,
5    ADD PRIMARY KEY (`IDOrganizzatore`),
6    ADD CONSTRAINT `test02` FOREIGN KEY (`IDOrganizzatore`) REFERENCES `Persona`
   ↪  (`CF`)
7  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
8
```

Organizza

```
1
2  CREATE TABLE `Organizza` (
3    `IDOrganizzatore` varchar(16) NOT NULL,
4    `IDCompetizione` int(20) NOT NULL,
5     ADD PRIMARY KEY (`IDOrganizzatore`,`IDCompetizione`),
6     ADD CONSTRAINT `fkcomp` FOREIGN KEY (`IDCompetizione`) REFERENCES
   ↪  `Competizione` (`IDCompetizione`),
7    ADD CONSTRAINT `fkorg` FOREIGN KEY (`IDOrganizzatore`) REFERENCES
   ↪  `OrganizzatoreEvento` (`IDOrganizzatore`)
8  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
9
```

Giudice

```
CREATE TABLE `Giudice` (
  `IDGiudice` varchar(16) NOT NULL,
  `Organizzazione` varchar(20) NOT NULL,
  ADD PRIMARY KEY (`IDGiudice`),
  ADD CONSTRAINT `tt1` FOREIGN KEY (`IDGiudice`) REFERENCES `Persona` (`CF`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

GiudicaGara

```
CREATE TABLE `GiudicaGara` (
  `IDGiudice` varchar(16) NOT NULL,
  `IDGara` int(20) NOT NULL,
   ADD PRIMARY KEY (`IDGiudice`,`IDGara`),
   ADD CONSTRAINT `fk1` FOREIGN KEY (`IDGiudice`) REFERENCES `Giudice`
↪  (`IDGiudice`),
   ADD CONSTRAINT `fk2` FOREIGN KEY (`IDGara`) REFERENCES `Gara` (`IDGara`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Categoria

```
CREATE TABLE `Categoria` (
  `IDCategoria` int(20) NOT NULL,
  `nomeCategoria` varchar(20) NOT NULL,
  `pesoMin` int(5) NOT NULL,
  `pesoMax` int(5) NOT NULL,
  ADD PRIMARY KEY (`IDCategoria`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

CategoriaAppartenenza

```
1
2  CREATE TABLE `CategoriaAppartenenza` (
3    `IDCategoria` int(20) NOT NULL,
4    `IDAtleta` varchar(16) NOT NULL,
5    ADD PRIMARY KEY (`IDCategoria`,`IDAtleta`),
6    ADD CONSTRAINT `testtt1` FOREIGN KEY (`IDAtleta`) REFERENCES `Atleta`
   ↪  (`IDAtleta`),
7    ADD CONSTRAINT `testtt2` FOREIGN KEY (`IDCategoria`) REFERENCES `Categoria`
8  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
9
```

Competizione

```
1
2  CREATE TABLE `Competizione` (
3    `IDCompetizione` int(20) NOT NULL,
4    `Data` date NOT NULL,
5    `IDStruttura` int(20) NOT NULL,
6    `numeroSpettatori` int(20) NOT NULL,
7    ADD PRIMARY KEY (`IDCompetizione`),
8    ADD CONSTRAINT `test` FOREIGN KEY (`IDStruttura`) REFERENCES `Struttura`
   ↪  (`IDStruttura`)
9  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
10
```

TipologiaGara

```
1
2  CREATE TABLE `TipologiaGara` (
3    `IDTipologia` int(20) NOT NULL,
4    `Genere` varchar(5) NOT NULL,
5    `nomeTipologia` varchar(10) NOT NULL,
6    ADD PRIMARY KEY (`IDTipologia`)
7  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
8
```

Gara

```
1
2  CREATE TABLE `Gara` (
3    `IDGara` int(20) NOT NULL,
4    `IDCompetizione` int(20) NOT NULL,
5    `IDTipologia` int(20) NOT NULL,
6    `IDCategoria` int(20) NOT NULL,
7    ADD PRIMARY KEY (`IDGara`),
8    ADD CONSTRAINT `testt1` FOREIGN KEY (`IDCategoria`) REFERENCES `Categoria`
   ↪  (`IDCategoria`),
9    ADD CONSTRAINT `testt2` FOREIGN KEY (`IDCompetizione`) REFERENCES
   ↪  `Competizione` (`IDCompetizione`),
10   ADD CONSTRAINT `testt3` FOREIGN KEY (`IDTipologia`) REFERENCES `TipologiaGara`
   ↪  (`IDTipologia`)
11 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
12
```

Posizionamento

```
1
2  CREATE TABLE `Posizionamento` (
3    `IDGara` int(20) NOT NULL,
4    `IDAtleta` varchar(16) NOT NULL,
5    `Posizione` int(5) NOT NULL,
6    `Punteggio` float NOT NULL,
7    ADD PRIMARY KEY (`IDGara`,`IDAtleta`),
8    ADD CONSTRAINT `t1` FOREIGN KEY (`IDAtleta`) REFERENCES `Atleta` (`IDAtleta`),
9    ADD CONSTRAINT `t2` FOREIGN KEY (`IDGara`) REFERENCES `Gara` (`IDGara`)
10 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
11
```

Regolamento

```
1
2  CREATE TABLE `Regolamento` (
3    `IDTipologia` int(20) NOT NULL,
4    `IDCategoria` int(20) NOT NULL,
5    `annoRegolamento` int(10) NOT NULL,
6     ADD PRIMARY KEY (`IDTipologia`,`IDCategoria`,`annoRegolamento`),
7     ADD CONSTRAINT `t11` FOREIGN KEY (`IDCategoria`) REFERENCES `Categoria`
   ↪  (`IDCategoria`),
8    ADD CONSTRAINT `t22` FOREIGN KEY (`IDTipologia`) REFERENCES `TipologiaGara`
   ↪  (`IDTipologia`)
9  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
10
```

Struttura

```
1
2  CREATE TABLE `Struttura` (
3    `IDStruttura` int(20) NOT NULL,
4    `Nome` varchar(30) NOT NULL,
5    `Luogo` varchar(40) NOT NULL,
6    ADD PRIMARY KEY (`IDStruttura`)
7  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
8
9
```

11 Inserimento righe

Adesso si popolano le varie tabelle con dati non autentici.
L'obiettivo è fornire un esempio dei record che fanno parte della base di dati.

Persona

```
1
2  INSERT INTO `Persona` (`CF`, `Nome`, `Cognome`, `Telefono`, `Residenza`, `CAP`)
   ↪  VALUES
3  ('CDHVSM36H68D279F', 'Riccardo', 'Catania', '6372921977', 'Contrada Marini 95',
   ↪  16249),
4  ('DHYPGB66H08E707I', 'Samuele', 'Calanna', '4395538665', 'Via Davide 911',
   ↪  58320),
5  ('DXDFGD68L17F398P', 'Tino', 'Scardaci', '4631160981', 'Borgo Giacinta 142',
   ↪  23569),
6  ('FMPCVD63A29L384Q', 'Martina', 'Rodriguez', '3184263748', 'Rotonda Neri 189',
   ↪  56289),
7  ('GFQPDR63A59G459V', 'Valerio', 'Abbruscato', '3978223027', 'Rotonda Rizzi 77',
   ↪  84769),
8  ('GHBVBV46P22L217T', 'Francesca', 'Sapienza', '0432772808', 'Strada Lazzaro 838',
   ↪  13222),
9  ('HFLPCL63C62L673B', 'Raffaele', 'Cinquegrani', '5365555599', 'Piazza Loredana
   ↪  782', 23757),
10 ('HHTZCD57D61L554M', 'Rossana', 'Fourier', '2915230551', 'Via Armellina 24',
   ↪  24704),
11 ('JRMFZC69T06L537B', 'Patrizio', 'Pepe', '8871907941', 'Strada Ercole 025',
   ↪  78000),
12 ('LBHDRX82D44E892E', 'Giuseppe', 'Blanco', '8085202794', 'Via Orlando 444 Piano
   ↪  4', 43218),
13 ('NDSHHS58A54D218L', 'Marco', 'Pulvirenti', '0491671103', 'Via Amos 9', 46897),
14 ('NRQLPB51D02Z600V', 'Gianluca', 'Di Stefano', '6287796171', 'Contrada Alan 11',
   ↪  85200),
15 ('NVOYDS94R15G536D', 'Alessandro', 'La Rosa', '6666545217', 'Strada Rocco 507
   ↪  Piano 4', 75230),
16 ('PNFSCH55P28B739N', 'Salvatore', 'Coglitoro', '0628839159', 'Strada Marieva 3',
   ↪  27836),
17 ('PRVBCL46S52G382G', 'Federico', 'Finocchiaro', '2023408146', 'Contrada Esposito
   ↪  999', 12345),
18 ('PZTYCZ72P16E468Q', 'Chiara', 'Arrivabene', '2104618368', 'Incrocio Barbieri 9',
   ↪  74163),
19 ('RVRDFB70R08I381R', 'Maya', 'Silva', '3673069696', 'Incrocio Galli 1 Piano 2',
   ↪  12500),
```

```
('SPSPNQ37M22G616K', 'Teo', 'Ravenna', '4691886957', 'Contrada Ione 780', 55851),
('WQKJTF77A69B869L', 'Emanuele', 'Majeli', '3771314891', 'Rotonda Marina 279',
↪  95012),
('YFGDYB83M08L696Z', 'Luca', 'Camiola', '4526352197', 'Via Grazia 8', 48216);

```

Atleta

```

INSERT INTO `Atleta` (`IDAtleta`, `Genere`, `Età`, `Peso`, `Altezza`) VALUES
('FMPCVD63A29L384Q', NULL, 'F', 33, 60, 171),
('LBHDRX82D44E892E', 'JRMFZC69T06L537B', 'M', 32, 82, 159),
('NRQLPB51D02Z600V', 'JRMFZC69T06L537B', 'M', 24, 67, 174),
('NVOYDS94R15G536D', 'PRVBCL46S52G382G', 'M', 27, 102, 176),
('RVRDFB70R08I381R', 'PRVBCL46S52G382G', 'F', 18, 53, 165),
('WQKJTF77A69B869L', 'JRMFZC69T06L537B', 'M', 24, 62, 175);

```

Allenatore

```

INSERT INTO `Allenatore` (`IDAllenatore`, `IDAtleta`, `nomePalestra`) VALUES
('JRMFZC69T06L537B', 'MC Fit'),
('PRVBCL46S52G382G', 'Kayoka');


```

Spettatore

```
1
2  INSERT INTO `Spettatore` (`IDSpettatore`, `IDCompetizione`) VALUES
3  ('CDHVSM36H68D279F', '2003-01-02'),
4  ('GFQPDR63A59G459V', '1992-09-15'),
5  ('GHBVBV46P22L217T', '1987-01-16'),
6  ('HFLPCL63C62L673B', '2008-06-20'),
7  ('HHTZCD57D61L554M', '2006-07-06'),
8  ('PZTYCZ72P16E468Q', '1985-11-14'),
9  ('YFGDYB83M08L696Z', '2005-04-14');
10
```

SpettaCompetizione

```
1
2  INSERT INTO `SpettaCompetizione` (`IDSpettatore`, `IDCompetizione`) VALUES
3  ('CDHVSM36H68D279F', 1),
4  ('GFQPDR63A59G459V', 1),
5  ('GFQPDR63A59G459V', 3),
6  ('GHBVBV46P22L217T', 1),
7  ('GHBVBV46P22L217T', 5),
8  ('HFLPCL63C62L673B', 4),
9  ('HHTZCD57D61L554M', 4),
10 ('PZTYCZ72P16E468Q', 6),
11 ('YFGDYB83M08L696Z', 1),
12 ('YFGDYB83M08L696Z', 5);
13
```

OrganizzatoreEvento

```
1
2  INSERT INTO `OrganizzatoreEvento` (`IDOrganizzatore`, `IDCompetizione`,
   ↪  `numEventiOrganizzati`) VALUES
3  ('NDSHHS58A54D218L', 'FIC'),
4  ('PNFSCH55P28B739N', 'FIC');
5
```

Organizza

```
INSERT INTO `Organizza` (`IDOrganizzatore`, `IDCompetizione`) VALUES
('NDSHHS58A54D218L', 5),
('PNFSCH55P28B739N', 2),
('PNFSCH55P28B739N', 5),
('PNFSCH55P28B739N', 6);

```

Giudice

```
INSERT INTO `Giudice` (`IDGiudice`, `IDGara`) VALUES
('DHYPGB66H08E707I', 'Burningate'),
('SPSPNQ37M22G616K', 'Movimento Calisthenics');

```

GiudicaGara

```
INSERT INTO `GiudicaGara` (`IDGiudice`, `IDGara`) VALUES
('DHYPGB66H08E707I', 4),
('DHYPGB66H08E707I', 14),
('SPSPNQ37M22G616K', 5),
('SPSPNQ37M22G616K', 12);

```

Categoria

```
1
2  INSERT INTO `Categoria` (`IDCategoria`, `nomeCategoria`, `pesoMin`, `pesoMax`)
   ↪  VALUES
3  (1, 'Beginners', 50, 69),
4  (2, 'Beginners', 70, 89),
5  (3, 'Beginners ', 90, 109),
6  (4, 'Junior', 50, 69),
7  (5, 'Junior', 70, 89),
8  (6, 'Junior', 90, 109),
9  (7, 'Senior', 50, 69),
10 (8, 'Senior', 70, 89),
11 (9, 'Senior', 90, 109);
12
```

CategoriaAppartenenza

```
1
2  INSERT INTO `CategoriaAppartenenza` (`IDCategoria`, `IDAtleta`) VALUES
3  (1, 'FMPCVD63A29L384Q'),
4  (4, 'NVOYDS94R15G536D'),
5  (5, 'RVRDFB70R08I381R'),
6  (7, 'LBHDRX82D44E892E'),
7  (7, 'WQKJTF77A69B869L'),
8  (9, 'NRQLPB51D02Z600V');
```

Competizione

```
1
2  INSERT INTO `Competizione` (`IDCompetizione`, `Data`, `IDStruttura`,
   ↪  `numeroSpettatori`) VALUES
3  (1, '2017-12-13', 1, 301),
4  (2, '2022-12-23', 2, 56),
5  (3, '2015-12-17', 3, 2101),
6  (4, '2018-10-17', 3, 780),
7  (5, '2014-04-23', 5, 120),
8  (6, '2020-08-21', 6, 670);
9
```

TipologiaGara

```
1
2  INSERT INTO `TipologiaGara` (`IDTipologia`, `Genere`, `nomeTipologia`) VALUES
3  (1, 'M', 'Skills'),
4  (2, 'F', 'Skills'),
5  (3, 'M', 'Endurance'),
6  (4, 'F', 'Endurance'),
7  (5, 'M', 'Strength'),
8  (6, 'F', 'Strength');
9
```

Gara

```
1
2  INSERT INTO `Gara` (`IDGara`, `IDCompetizione`, `IDTipologia`, `IDCategoria`)
   ↪  VALUES
3  (3, 1, 1, 2),
4  (4, 1, 4, 6),
5  (5, 2, 3, 4),
6  (6, 3, 5, 6),
7  (7, 4, 4, 7),
8  (8, 3, 5, 8),
9  (9, 5, 2, 4),
10 (10, 3, 5, 6),
11 (11, 5, 3, 8),
12 (12, 6, 6, 4),
13 (13, 1, 2, 6),
14 (14, 6, 4, 3),
15 (15, 4, 1, 8);
16
```

Posizionamento

```
1
2  INSERT INTO `Posizionamento` (`IDGara`, `IDAtleta`, `Posizione`, `Punteggio`)
   ↪  VALUES
3  (3, 'WQKJTF77A69B869L', 2, 12.8),
4  (4, 'FMPCVD63A29L384Q', 1, 27.9),
5  (5, 'NVOYDS94R15G536D', 6, 35.9),
6  (5, 'WQKJTF77A69B869L', 1, 52.7),
7  (6, 'LBHDRX82D44E892E', 8, 24),
8  (8, 'NVOYDS94R15G536D', 1, 59.1),
9  (9, 'RVRDFB70R08I381R', 5, 44.5),
10 (10, 'NRQLPB51D02Z600V', 6, 56.3),
11 (11, 'LBHDRX82D44E892E', 4, 38),
12 (14, 'FMPCVD63A29L384Q', 2, 27),
13 (14, 'RVRDFB70R08I381R', 1, 34.9),
14 (15, 'WQKJTF77A69B869L', 1, 60.9);
15
```

Regolamento

```
1
2  INSERT INTO `Regolamento` (`IDRegolamento`, `IDTipologia`, `IDCategoria`,
   ↪  `annoRegolamento`) VALUES
3  (1, 6, 2020),
4  (2, 1, 2019),
5  (3, 5, 2017),
6  (3, 5, 2018),
7  (3, 5, 2019),
8  (3, 5, 2021),
9  (3, 5, 2022),
10 (4, 6, 2020),
11 (5, 9, 2021),
12 (6, 9, 2019);
13
```

Struttura

```

INSERT INTO `Struttura` (`IDStruttura`, `Nome`, `Luogo`) VALUES
(1, 'Fitness And Fight', 'Acireale'),
(2, 'Stefano\'s gym', 'Acitrezza'),
(3, 'Tiger Club', 'Lamezia Terme'),
(4, 'Fitness World', 'Monte Sacro'),
(5, 'Athlon', 'San Gregorio'),
(6, 'Virgin', 'Cannizzaro');


```

12 Creazione Trigger

Un trigger è un pezzo di codice che viene eseguito automaticamente dal sistema di gestione del database (DBMS) in risposta a un evento specifico (come una insert, delete o update).

I trigger vengono spesso utilizzati per far rispettare le regole aziendali o eseguire altre attività quando si verificano determinati eventi nel database, come ad esempio quando viene inserito o aggiornato un nuovo record o quando viene eliminata una tabella.

I trigger sono di solito associati ad una tabella specifica e possono essere attivati prima o dopo un'operazione di inserimento, aggiornamento o eliminazione su quella tabella.

Essi vengono utilizzati per garantire dei vincoli d'integrità. I trigger vengono implementati utilizzando SQL e vengono memorizzati nel database insieme agli altri oggetti del database.

Ecco elencato le principali componenti di cui è composto un trigger:

- **Nome**
- **Target (tabella controllata)**
- **Modalità (before o after)**
- **Evento (insert, delete o update)**
- **Granularità (statement-level o row-level)**
- **Alias dei valori o tabelle di transizione**
- **Azione**
- **Timestamp di creazione**

Per il progetto in esame, sono stati scritti diversi trigger per introdurre dei vincoli d'integrità.

Il primo Trigger in questione è un trigger di inserimento.
Esso in seguito all'inserimento di un nuovo regolamento, controlla se esistono già altri regolamenti per l'attuale categoria e tipologia. Se vi sono più di 5 regolamenti passati, rimuove il regolamento con l'anno meno recente.

```
1
2  CREATE TRIGGER limitiRegolamento
3  AFTER INSERT ON Regolamento
4  FOR EACH ROW
5  BEGIN
6     DECLARE numRegolamenti INT;
7
8     SELECT COUNT(*) INTO numRegolamenti
9     FROM Regolamento
10    WHERE IDCategoria = NEW.IDCategoria AND IDTipologia = NEW.IDTipologia;
11
12    IF numRegolamenti >= 5 THEN
13       DELETE FROM Regolamento
14       WHERE IDCategoria = NEW.IDCategoria AND IDTipologia = NEW.IDTipologia AND
   ↪  annoRegolamento <> NEW.annoRegolamento
15       ORDER BY annoRegolamento ASC
16       LIMIT 1;
17    END IF;
18 END
```

Un ulteriore trigger, si occupa di aggiornare il numero di spettatori in una Competizione in seguito ad un inserimento sulla tabella Spettatore:

```
1
2  CREATE TRIGGER inserimentoSpettatore
3  AFTER INSERT ON SpettaCompetizione
4  FOR EACH ROW
5  BEGIN
6     UPDATE Competizione
7     SET numeroSpettatori = numeroSpettatori + 1
8     WHERE IDCompetizione = NEW.IDCompetizione;
9  END;
```

Il seguente vincolo d'integrità, vieta a due competizioni di tenersi nella stessa struttura contemporaneamente, il seguente codice garantisce ciò :

```
1
2  CREATE TRIGGER cancellaCompetizioniSimultanee
3  AFTER INSERT ON Competizione
4  FOR EACH ROW
5  BEGIN
6    DECLARE numCompetizioni INT;
7    SELECT COUNT(*) INTO numCompetizioni FROM Competizione
8    WHERE IDStruttura = NEW.IDStruttura AND Data = NEW.Data AND IDCompetizione <>
   ↪  NEW.IDCompetizione;
9
10   IF numCompetizioni > 0 THEN
11     DELETE FROM Competizione WHERE IDCompetizione = NEW.IDCompetizione;
12   END IF;
13 END;
```

Un altro trigger viene creato per evitare che due atleti, nella stessa gara ricevino le stesse posizioni.

Ciò può essere evitato, attraverso un Trigger *after insert* che rilevando l'incongruenza, rimuove il nuovo record.

```
1
2  CREATE TRIGGER eliminaAtletaPosizionamento
3  AFTER INSERT ON Posizionamento
4  FOR EACH ROW
5  BEGIN
6     DECLARE numAtleti INT;
7
8     SELECT COUNT(*) INTO numAtleti
9     FROM Posizionamento
10    WHERE IDGara = NEW.IDGara AND posizione = NEW.posizione AND IDAtleta <>
   ↪  NEW.IDAtleta;
11
12    IF numAtleti > 0 THEN
13       DELETE FROM Posizionamento
14       WHERE IDGara = NEW.IDGara AND posizione = NEW.posizione AND IDAtleta =
   ↪  NEW.IDAtleta;
15    END IF;
16 END;
```

Il seguente trigger invece, si assicura che il giudice di una determinata gara, non sia l'allenatore di un partecipante.

```
1
2  CREATE TRIGGER verificaAllenatore
3  AFTER INSERT ON Giudice
4  FOR EACH ROW
5  BEGIN
6     DECLARE Allenatore INT;
7
8     SELECT COUNT(*) INTO Allenatore
9     FROM Allenatore
10    WHERE IDAllenatore = NEW.IDGiudice;
11
12    IF Allenatore > 0 THEN
13       DELETE FROM Giudice
14       WHERE IDGiudice = NEW.IDGiudice AND IDGara = NEW.IDGara;
15    END IF;
16 END;
```

Un ultimo trigger viene realizzato per evitare di inserire in Posizionamento, un atleta di Genere discorde alla Tipologia di Gara della gara inserita nel nuovo record :

```
1
2  CREATE TRIGGER verificaAllenatore
3  AFTER INSERT ON Giudice
4  FOR EACH ROW
5  BEGIN
6      DECLARE genereAtleta VARCHAR(5) DEFAULT '';
7      DECLARE genereTipologia VARCHAR(5) DEFAULT '';
8      SELECT Genere INTO genereAtleta
9      FROM Atleta
10     WHERE IDAtleta = NEW.IDAtleta;
11
12     SELECT TipologiaGara.Genere INTO genereTipologia
13     FROM TipologiaGara, Gara
14     WHERE TipologiaGara.IDTipologia = Gara.IDTipologia
15     AND Gara.IDGara = NEW.IDGara;
16
17     IF genereAtleta <> genereTipologia THEN
18         DELETE FROM Posizionamento
19         WHERE IDGara = NEW.IDGara AND IDAtleta = NEW.IDAtleta;
20     END IF;
21 END
```

13 Query in SQL

Le query sono istruzioni utilizzate per interagire con il database, come ad esempio estrarre, modificare o eliminare dati. In quest'ultima sezione, verranno scritte alcune query riguardanti operazioni interattive/batch elencate nella tabella delle operazioni (nella sezione della progettazione logica).

Le seguenti query rispettano la sintassi di SQL e sono state testate prima di essere inserite nel nell'attuale progetto.

Se qualcuna tra esse, in una base di dati analoga a questa ma con righe differenti, dovesse tornare un insieme nullo di elementi, vi può essere una mancanza di dati all'interno delle tabelle;

in tal caso, qualche inserimento aggiuntivo, risolverà il "problema".

Operazione 1

Visualizzazione del nome degli atleti e delle gare in cui si sono posizionati in cui, la tipologia è Skills ed essa è tenuta in una struttura di nome "FitnessAndFight"

```

SELECT P.Nome, G.IDGara
FROM Gara G, Competizione Co, Struttura S1, TipologiaGara T, Posizionamento Po,
↪  Persona P
WHERE G.IDCompetizione = Co.IDCompetizione AND Co.IDStruttura = S1.IDStruttura
↪  AND S1.Nome = "Fitness And Fight"
AND T.IDTipologia = G.IDTipologia AND T.nomeTipologia = 'Skills' AND Po.IDGara =
↪  G.IDGara
AND P.CF = Po.IDAtleta;

```

Operazione 4:

Visualizzazione dell'allenatore con più allievi

```

SELECT A.IDAllenatore
FROM Atleta A
GROUP BY A.IDAllenatore
HAVING COUNT(A.IDAtleta) >= (SELECT MAX(conteggioAtleti)
                                FROM (SELECT COUNT(A1.IDAtleta) as
↪  conteggioAtleti
                                      FROM Atleta A1
                                      GROUP BY A1.IDAllenatore) subquery);

```

Operazione 6

Visualizzazione di tutti i posizionamenti ottenuti dall'atleta "Emanuele Majeli" in tutte le gare effettuate

```
1
2  SELECT P.Nome, G.IDGara, Po.posizione
3  FROM Gara G, Posizionamento Po, Atleta A, Persona P
4  WHERE G.IDGara = Po.IDGara AND Po.IDAtleta = A.IDAtleta AND A.IDAtleta = P.CF
5  AND P.Nome = 'Emanuele' AND P.Cognome = 'Majeli';
6
```

Operazione 10

Visualizza per ogni categoria di atleti,l'atleta con peso maggiore a quello medio

```
1
2  SELECT *
3  FROM CategoriaAppartenenza CA, Atleta A
4  WHERE CA.IDAtleta = A.IDAtleta
5  GROUP BY CA.IDCategoria
6  HAVING A.Peso >= (SELECT AVG(A1.Peso) FROM Atleta A1, CategoriaAppartenenza CA1
7  WHERE CA.IDCategoria = CA1.IDCategoria);
8
```

Operazione 12:

Visualizzazione del nome e cognome dell'atleta femminile che ha gareggiato in almeno una gara di Endurance e il quale nome inizi per M e l'id del suo allenatore se presente.

```
1
2  SELECT DISTINCT P.Nome, P.Cognome, A.IDAllenatore
3  FROM Posizionamento Po, TipologiaGara T, Gara G, Persona P, Atleta A
4  WHERE Po.IDAtleta = P.CF AND Po.IDGara = G.IDGara AND G.IDTipologia =
   ↪  T.IDTipologia
5  AND T.nomeTipologia = 'Endurance' AND T.Genere = 'F' AND P.Nome LIKE 'M%' AND
   ↪  A.IDAtleta = Po.IDAtleta;
6
```

Operazione Bonus:

Visualizzazione degli atleti di ogni tipologia di gare, che hanno ottenuto il punteggio più alto.

```
1
2  SELECT Po.IDAtleta, Po.IDGara, Po.Punteggio
3  FROM Atleta A, Gara G, Posizionamento Po, TipologiaGara T
4  WHERE A.IDAtleta = Po.IDAtleta AND Po.IDGara = G.IDGara AND G.IDTipologia =
   ↪  T.IDTipologia
5  AND Po.Punteggio >= (
6     SELECT MAX(Po1.Punteggio)
7     FROM Posizionamento Po1, Gara G1
8     WHERE Po1.IDGara = G1.IDGara AND G1.IDTipologia = T.IDTipologia
9  )
10 GROUP BY T.IDTipologia;
11
```

Part V
Conclusioni

Il progetto appena svolto ha avuto come finalità la creazione di una base di dati per la gestione delle competizioni di calisthenics.

Esso è stato suddiviso in tre parti, fondamentali nello sviluppo di un DB.

Nella fase iniziale, sono stati raccolti i requisiti, studiate i ruoli di ciascuna entità e le interazioni tra di essi. Sono state effettuate delle scelte progettuali per avere un buon trade off tra dimensione della base di dati e la capacità di essa.

E' stato creato un glossario dei termini per evitare ambiguità e costruito un diagramma Entità-Relazione ad hoc. Lo strumento utilizzato per il modello è *draw.io*.

In seguito è stata svolta una ristrutturazione dello schema che presentava delle incompatibilità con il modello logico; sono stati illustrati i due dizionari dei dati (delle entità e delle relazioni) ed è stata effettuata un'attenta analisi delle ridondanze dopo aver costruito le tabelle dei volumi e delle operazioni. In seguito sono state costruite le tabelle (entità e relazioni), definite le chiavi primarie ed esterne e ridefiniti gli attributi in esse.

Il tutto è stato tradotto in linguaggio SQL per la fase di progettazione fisica, con cui sono state effettuate diverse operazioni quali la creazione delle tabelle, l'inserimento dei record in esse, la creazione di trigger per stabilire dei vincoli d'integrità e delle query che rispondessero alle operazioni definite nella sezione precedente.