AA 22/23

Basi di Dati

Documentazione progetto Basi di dati

Università di Catania

Alfio Spoto

Progetto Basi di Dati

Documentazione relativa al progetto di Basi di Dati

Si vuole implementare un database relativo alla gestione di eventi sportivi

Indice

Descrizione e Specifiche sui dati

Si vuole realizzare il progetto della base di dati relativa alla gestione di gare sportive, partendo da un insieme di requisiti.

Le fasi da svolgere vanno dall'analisi dei requisiti, alle varie fasi dell'analisi fino all'implementazione delle operazioni previste. Durante il progetto è necessario produrre un insieme di documenti, che costituiscono appunto la documentazione del progetto:

- Analisi dei requisiti;
- Lo schema concettuale, tramite il modello E-R, presentato a diversi gradi di raffinamento(progettazione top-down);
- Una descrizione delle operazioni previste e le relative tavole di carico;
- Lo schema ottenuto per ristrutturazione dalla prima fase della progettazione logica e lo schema logico finale;
- Un listato delle interrogazioni e delle istruzioni (aggiornamenti, inserimenti, cancellazioni) SQL relative alle operazioni previste;
- Contenuto di test della base di dati e nella stampa dei risultati delle interrogazioni su tali dati;

Descrizione e Specifiche sui dati

Si vuole realizzare il progetto della base di dati relativa alla gestione di gare sportive. Diversi atleti partecipano a delle competizioni sportive. Ogni partecipante fa parte di uno solo dei team presenti alla gara ed ogni squadra si compone di 12 giocatori. Si rende necessario iscrivere dei team alla gara (circa 4 volte al mese). Ogni team viene allenato da un rispettivo coach che si occupa dell'amministrazione e allenamento della squadra, tuttavia ogni coach può allenare più team. Si consideri possibile accettare modifiche ai dati relativi al team(2 volte al mese). Si renda possibile inoltre assegnare un altro team ad un allenatore . In media un coach allena due team. I concorrenti praticano un determinato tipo di sport che pregiudica le gare a cui possono prendere parte, in particolare un atleta può partecipare solo ad una gara relativa alla categoria di sport che pratica. Si ha la necessità di conteggiare il numero di team iscritti ad una gara (circa 1 volta al giorno per mantenere le statistiche sempre aggiornate). Considerato che ogni atleta pratica un solo sport, esso potrà collocarsi in al più una classifica. Le performance degli atleti vengono valutati da un giudice esperto, che attribuisce una valutazione alla prestazione dell'atleta. Per ogni gara vi sono tre giudici e la valutazione verrà considerata positiva solo se almeno due giudici su tre danno un giudizio superiore al 18 (i voti sono in trentesimi). Data la possibilità di cambio di giudice si rende necessario considerare la possibilità di poter aggiornare i dati relativi ai giudici. Vengono effettuate 4 gare l'anno, quindi una competizione ogni 3 mesi. La classifica viene stilata sulla base del punteggio cumulativo di ogni atleta sulle quattro performance fornite durante la gara(ogni performance ha infatti un progressivo da 1 a 3), ed il punteggio di ogni squadra verrà determinato come somma dei punteggi di tutti e 12 i componenti. Al termine della competizione tutte le squadre si vedranno assegnata una posizione in classifica. Si rende necessario visualizzare il posizionamento in classifica di un team alla fine di ogni gara. Si rende necessario visualizzare il posizionamento in classifica di un team alla fine di ogni gara. Ogni gara si svolge in una location ed è possibile accedervi solo dopo aver comprato un biglietto, il quale ricopre esattamente un posto all'interno di un settore della location. Uno stesso utente può acquistare più biglietti ed il costo di ogni ticket dipende dal settore che si occupa. Se un utente ha un ripensamento, può comunque cambiare il biglietto acquistato precedentemente con quello relativo ad un altro settore, a fronte di un pagamento o rimborso sulla base della differenza di costo (in media non si hanno più di 3 rimborsi al mese). Vi sono in media 50.000 spettatori ad ogni gara. Si ha la necessità di conteggiare il numero di biglietti venduti per ogni settore (circa 1 volta al giorno per mantenere le statistiche sempre aggiornate).

Glossario dei termini

Termine	Descrizione	Sinonimi	Termini collegati
Atleta	individuo impegnato nell'attività sportiva	partecipante, concorrente, concorrente	team, valutazione
Performance	risultati conseguiti da un'atleta	prestazione	atleta
Gara	competizione tra vari atleti	competizione	atleta, location
Team	squadra di persone che gareggiano	squadra	classifica
Location	luogo dove si svolge la gara		biglietto
Spettatore	persona che assiste alla gara	utente	biglietto
Biglietto	stampato dato come certificazione di un pagamento	ticket	location
Classifica	graduatoria dove figurano le posizioni dei vari teams		valutazione, teams
Giudice	persona autorizzata a valutare una performance		valutazione
Valutazione	punteggio attribuito ad una performance da parte di un giudice	giudizio, punteggio	giudice
Coach	persona che allena un team	allenatore	team

Analisi dei requisiti

Dati di carattere generale

Si vuole realizzare il progetto della base di dati relativa alla gestione di gare sportive. Diversi atleti partecipano a delle competizioni sportive.

Dati sugli atleti e sui team

Ogni partecipante fa parte di uno solo dei team presenti alla gara ed ogni squadra si compone di 12 giocatori. Si rende necessario iscrivere dei team alla gara (circa 4 volte al mese). I concorrenti praticano un determinato tipo di sport che pregiudica le gare a cui possono prendere parte, in particolare un atleta può partecipare solo ad una gara relativa allo sport che pratica. Si ha la necessità di conteggiare il numero di team iscritti ad una gara (circa 1 volta al giorno per mantenere le statistiche sempre aggiornate). Considerato che ogni atleta pratica un solo sport, esso potrà collocarsi in al più una classifica.

Dati sul coach

Ogni team viene allenato da un rispettivo coach che si occupa dell'amministrazione e allenamento della squadra, tuttavia ogni coach può allenare più team. Si consideri possibile accettare modifiche ai dati relativi al team(2 volte al mese). Si renda possibile inoltre assegnare un altro team ad un allenatore. In media un coach allena due team.

Dati sulla valutazione delle performance

Le performance degli atleti vengono valutati da un giudice esperto, che attribuisce una valutazione alla prestazione dell'atleta. Per ogni gara vi sono tre giudici e la valutazione verrà considerata positiva solo se almeno due giudici su tre danno un giudizio superiore al 18 (i voti sono in trentesimi). Data la possibilità di cambio di giudice si rende necessario considerare la possibilità di poter aggiornare i dati relativi ai giudici.

Dati sulla gara e classifica

Vengono effettuate 4 gare l'anno, quindi una competizione ogni 3 mesi. La classifica viene stilata sulla base del punteggio cumulativo di ogni atleta sulle quattro performance fornite durante la gara(ogni performance ha infatti un progressivo da 1 a 3), ed il punteggio di ogni squadra verrà determinato come somma dei punteggi di tutti e 12 i componenti. Si effettua il conteggio dei punti di ogni team alla fine di Al termine della competizione tutte le squadre si vedranno assegnata una posizione in classifica. Si rende necessario visualizzare il posizionamento in classifica di un team alla fine di ogni gara.

Dati sulla location e sui biglietti

Ogni gara si svolge in una location ed è possibile accedervi solo dopo aver comprato un biglietto, il quale ricopre esattamente un posto all'interno di un settore della location. Uno stesso utente può acquistare più biglietti ed il costo di ogni ticket dipende dal settore che si occupa. Se un utente ha un ripensamento, può comunque cambiare il biglietto acquistato precedentemente con quello relativo ad un altro settore, a fronte di un pagamento o rimborso sulla base della differenza di costo (in media non si hanno più di 3 cambi al mese). Vi sono in media 50.000 spettatori ad ogni gara. Si ha la necessità di conteggiare il numero di biglietti venduti per ogni settore (circa 1 volta al giorno per mantenere le statistiche sempre aggiornate).

Progettazione Concettuale

Per la strategia implementativa si opta per una strategia top-down. Le primitive di trasformazione *top-down* sono regole che operano su un singolo concetto dello schema e lo trasformano in una struttura più complessa che descrive il concetto con maggiore dettaglio. Si redige in breve tempo il progetto concettuale della base dati senza preoccuparsi dei dettagli descrivendone in modo astratto e generale la realtà di interesse. In questo modo si ha una visione generale delle componenti dell'intero sistema.

Di seguito vengono riportate le fasi di progettazione, aggiungendo sempre più entità e relazioni :

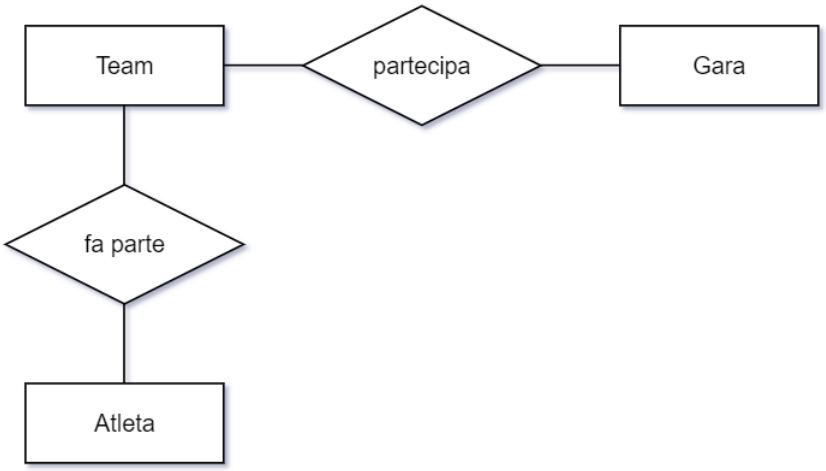
Primitive di trasformazione possibili

1. Si applica quando un'entità descrive due concetti diversi legati fra di loro. Consiste nello scomporre l'entità in due diverse entità legate da un'opportuna relazione.
2. Un entità è composta da due sotto-entità distinte. Consiste nello specificare l'entità in due entità differenti attraverso una gerarchia.
3. Una relazione in realtà descrive due relazioni diverse tra le stesse entità. Consiste nel collegare le due entità attraverso due relazioni distinte.
4. Una relazione descrive un concetto con esistenza autonoma. In questo caso essa va sostituita con un' entità.
5. Si applica per aggiungere attributi ad entità.
6. Si applica per aggiungere attributi alle relazioni.

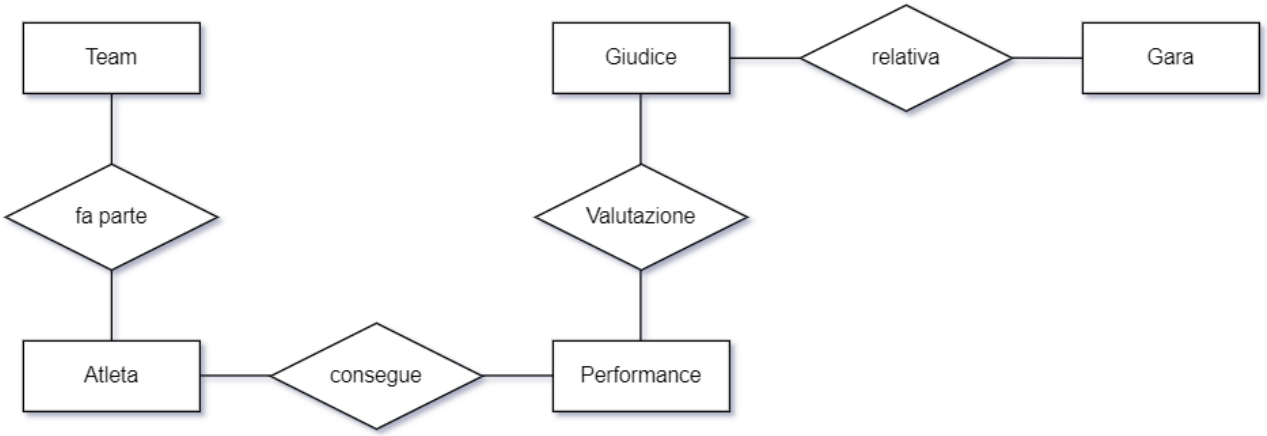
Schema scheletro

Si identifica l'entità gara. Si vuole quindi modellare che un atleta ve ne prende parte.

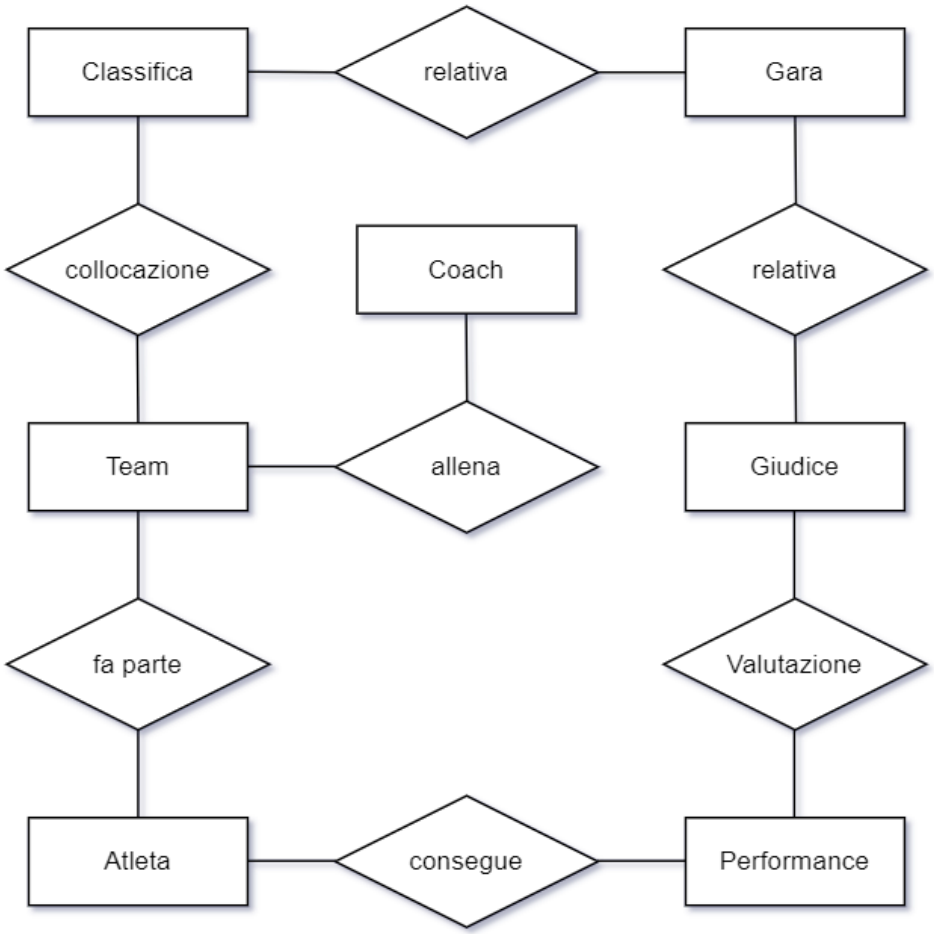
Ci si rende subito conto che non è un atleta a partecipare direttamente alla gara ma un atleta fa di certo parte di un team per quanto richiesto dai requisiti, per tanto l'entità atleta va scomposta nelle entità atleta e team (trasformazione 1) :



Si necessita di introdurre un giudice autorizzato a valutare la performance dell'atleta. Si necessita quindi di modellare che un atleta da una performance in gara, la quale sarà seguita da una valutazione del giudice :

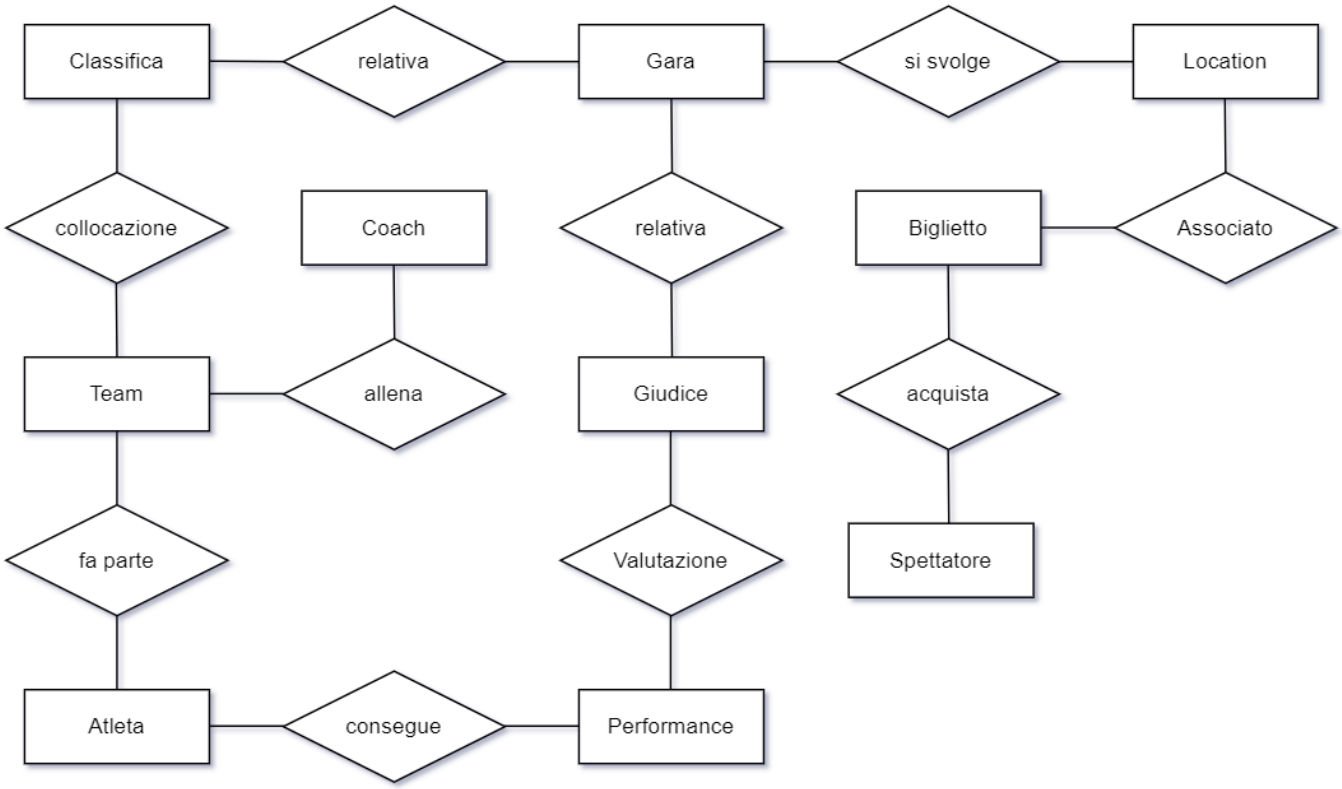


Per quanto richiesto dai requisiti, bisogna apportare delle modifiche affinché , a seguito della valutazione dei giudici sulle performance degli atleti si ha una collocazione dei team in una precisa posizione in classifica ed in oltre si richiede di modellare un'entità coach che si occupa della formazione della squadra :

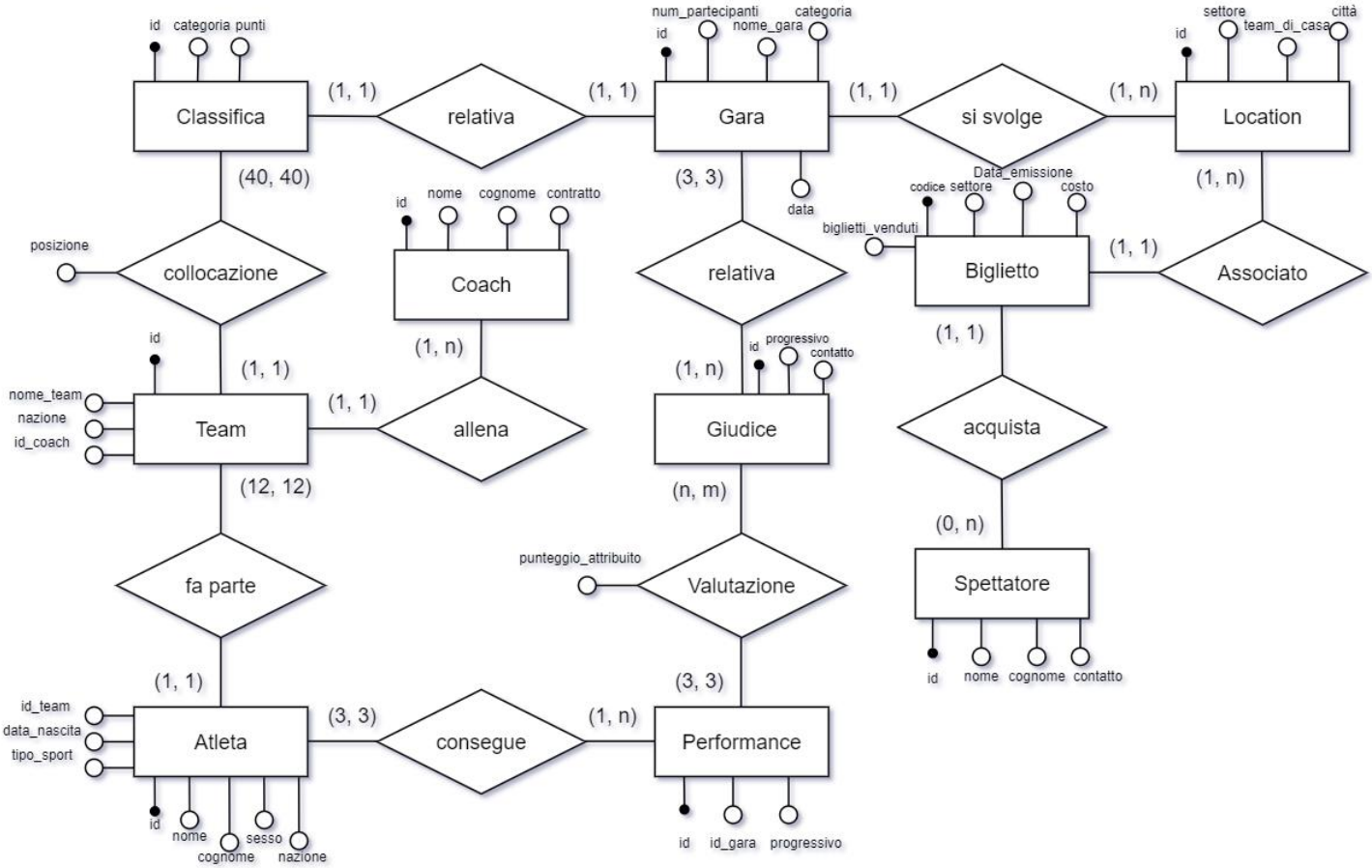


Schema scheletro completo

Si necessita di apportare ulteriori modifiche al diagramma in modo tale da modellare che un evento sportivo si svolge in una esatta location alla quale si ha accesso soltanto comprato un biglietto associato . L'acquisto del biglietto avviene da parte di uno spettatore che si colloca, attraverso l'acquisto del biglietto, in un preciso posto all'interno della location :



Schema scheletro completo



Business rules (vincoli non esprimibili dallo schema ER)

Vincoli sugli atleti

Un atleta può prendere parte solo ad una gara la cui categoria coincide con il tipo di sport praticato dall'atleta . Un atleta ha gareggiato correttamente se ha dato tutte e tre le performance e tutte e tre sono state valutate dai tre giudici. Nonostante atleti possano partecipare a più eventi sportivi, essi non possono partecipare a più eventi nello stesso giorno.

Vincoli sulla classifica

La posizione in classifica, raggiungibile mediante la tabella collocazione, rappresenta la posizione in classifica di un team per il quale tutti gli atleti hanno gareggiato correttamente. I punteggi degli atleti devono essere registrati in modo accurato e i risultati devono essere resi pubblici.

Vincoli sulle gare

Un evento sportivo può avere una sola data e un solo luogo. Ogni evento sportivo deve avere esattamente tre giudici. Gli eventi sportivi devono essere organizzati in modo da garantire la sicurezza degli atleti e degli spettatori. Gli eventi sportivi possono essere annullati o posticipati in caso di maltempo o altre circostanze impreviste. Gli eventi sportivi devono essere pubblicizzati in modo adeguato per attirare atleti e spettatori.

Vincoli sui biglietti

Gli spettatori devono acquistare un biglietto per accedere all'evento sportivo e il numero di spettatori ammessi può essere limitato al numero di biglietti disponibili per quel settore.

Porzione del dizionario dati – Entità

Entità	Descrizione	Attributi	Identificatore
Classifica	graduatoria dove figurano le posizioni dei vari teams	Id, categoria, punti	id
Team	squadra di persone che gareggiano	Id, nome_team, nazione, id_coach	id
Atleta	individuo impegnato nell'attività sportiva	Id, nome, cognome, sesso, id_team, nazione, data_nascita, id_team, tipo_sport	id
Coach	persona che allena un team	Id, nome, cognome, contatto	id
Gara	competizione tra vari atleti	Id, num_partecipanti, nome_gara, categoria	id
Giudice	persona autorizzata a valutare una performance	Id, progressivo, contatto	id
Performance	risultati conseguiti da un'atleta	Id, id_gara, progressivo	id
Biglietto	stampato dato come certificazione di un pagamento	Codice, settore, data_emissione, costo, biglietti_venduti	Codice
Spettatore	persona che assiste alla gara	Id, nome, cognome, contatto	id
Location	luogo dove si svolge la gara	Id, settore, città, team_di_casa	id

Porzione del dizionario dati – Relazioni

Relazione	Entità partecipanti	Descrizione	Attributi
Collocazione	Team - Classifica	un team si colloca in una determinata posizione nella classifica	posizione
Fa parte	Atleta - Team	ogni atleta fa parte di un team	
Consegue	Atleta - Performance	gli atleti in gara danno tre performance	
Allena	Coach - Team	ogni squadra ha un allenatore che si occupa della gestione sportiva del team	
Relativa (valutazione-gara)	Valutazione - Gara	ogni valutazione è relativa ad una certa gara	
Relativa (classifica-gara)	Classifica - Gara	ogni classifica è relativa ad una certa gara	
Si svolge	Gara - Location	un evento sportivo si svolge in una location	
Associato	Location - Biglietto	l'accesso ad una location avviene tramite l'acquisto di un biglietto da parte di uno spettatore	
Acquista	Spettatore -Biglietto	uno spettatore deve acquistare un biglietto per poter assistere alla competizione	
Valutazione	Giudice - Performance	le tre performance conseguite dagli atleti vengono valutate da tre giudici	punteggio_attribuito

Specifiche sulle operazioni

Si analizzano i requisiti richiesti in modo da evidenziare le principali operazioni che dovrà effettuare il database :

Si vuole realizzare il progetto della base di dati relativa alla gestione di gare sportive. Diversi atleti partecipano a delle competizioni sportive. Ogni partecipante fa parte di uno solo dei team presenti alla gara ed ogni squadra si compone di 12 giocatori**. Si rende necessario iscrivere dei team alla gara (circa 4 volte al mese).** Ogni team viene allenato da un rispettivo coach che si occupa dell'amministrazione e allenamento della squadra, tuttavia ogni coach può allenare più team. **Si consideri possibile accettare modifiche ai dati relativi al team(2 volte al mese). Si renda possibile inoltre assegnare un altro team ad un allenatore.** In media un coach allena due team. I concorrenti praticano un determinato tipo di sport che pregiudica le gare a cui possono prendere parte, in particolare un atleta può partecipare solo ad una gara relativa alla categoria di sport che pratica**.** Considerato che ogni atleta pratica un solo sport, esso potrà collocarsi in al più una classifica. Le performance degli atleti vengono valutati da un giudice esperto, che attribuisce una valutazione alla prestazione dell'atleta. Per ogni gara vi sono tre giudici e la valutazione verrà considerata positiva solo se almeno due giudici su tre danno un giudizio superiore al 18 (i voti sono in trentesimi). **Data la possibilità di cambio di giudice si rende necessario considerare la possibilità di poter aggiornare i dati relativi ai giudici**. Vengono effettuate 4 gare l'anno, quindi una competizione ogni 3 mesi. La classifica viene stilata sulla base del punteggio cumulativo di ogni atleta sulle quattro performance fornite durante la gara(ogni performance ha infatti un progressivo da 1 a 3), ed il punteggio di ogni squadra verrà determinato come somma dei punteggi di tutti e 12 i componenti. Al termine della competizione tutte le squadre si vedranno assegnata una posizione in classifica. **Si rende necessario visualizzare il posizionamento in classifica di un team alla fine di ogni gara**. Ogni gara si svolge in una location ed è possibile accedervi solo dopo aver comprato un biglietto, il quale ricopre esattamente un posto all'interno di un settore della location. Uno stesso utente può acquistare più biglietti ed il costo di ogni ticket dipende dal settore che si occupa. **Se un utente ha un ripensamento, può comunque cambiare il biglietto acquistato precedentemente con quello relativo ad un altro settore, a fronte di un pagamento o rimborso sulla base della differenza di costo (in media non si hanno più di 3 rimborsi al mese).** Vi sono in media 50.000 spettatori ad ogni gara. **Si ha la necessità di conteggiare il numero di biglietti venduti per ogni settore (circa 1 volta al giorno per mantenere le statistiche sempre aggiornate).**

Le principali operazioni richieste dalla base di dati sono :

1. Iscrivere un team alla gara ;
2. Modificare i dati di un team già iscritto alla gara ;
3. Modificare il prezzo ed il settore a cui si riferisce il biglietto(cambiare il biglietto) ;
4. Conteggiare il numero di biglietti venduti per ogni settore;
5. Visualizzare il posizionamento in classifica di un team ;
6. Modificare i dati di un giudice di gara;
7. Modificare i team seguiti da un coach ;
8. Calcolare il punteggio complessivo di un team a fine gara;

Tavola dei volumi

Concetto	Tipo	Volume
Team	E	
Atleta	E	12 ogni team
Gara	E	4 ogni anno
Giudice	E	3
Biglietto	E	
Performance	E	3
Spettatore	E	50.000
Location	E	
Valutazione	R	3

Tavola delle frequenze

Operazione	**Tipo**	**Frequenza**
Iscrivere un team alla gara	I	4v/mese
Modificare i dati di un team già iscritto alla gara	I	2v/mese
Modificare il prezzo ed il settore a cui si riferisce il biglietto(cambiare il biglietto)	I	3v/mese
Conteggiare il numero di biglietti venduti per ogni settore	I	1v/g
Visualizzare il posizionamento in classifica di un team	B	4v/anno
Modificare i dati di un giudice di gara	B	
Modificare i team seguiti da un coach	B	
Calcolare il punteggio complessivo di un team a fine gara	B	4v/anno

Analisi dei costi dovuti alle ridondanze

L'attributo biglietti_venduti è ridondante in quanto è possibile risalire al numero di biglietti venduti effettuando letture nella relazione Acquista. Bisogna valutare il costo delle operazioni previste dal database in presenza e in assenza dell'attributo ridondante, così da decidere se conviene rimuoverlo.

Nota :

Durante l'analisi si considera il costo degli accessi in scrittura doppio rispetto a quello degli accessi in lettura

Analisi dei costi relativa all'operazione 1 :

Iscrivere un team alla gara (4v/mese)

Con ridondanza :

- 1 scrittura in "Team"
- 1 lettura in "fa parte"
- 12 scritture in "Atleta"
- 1 scrittura in "Gara" (per aggiornare il numero di team partecipanti)

28 letture * 4v/mese = 112 accessi/mese

Senza ridondanza :

- 1 scrittura in "Team"
- 1 lettura in "fa parte"
- 12 scritture in "Atleta"
- 1 scrittura in "Gara"

28 letture * 4v/mese = 112 accessi/mese

Analisi dei costi relativa all'operazione 2 :

Modificare i dati di un team già iscritto alla gara (2v/mese)

Si potrebbe supporre che la modifica riguardi anche il coach relativo al team

Con ridondanza :

- 1 scrittura in "Team"
- 1 lettura in "allena"
- 1 scrittura in "Coach"

5 letture * 2v/mese = 10 accessi/mese

Senza ridondanza :

- 1 scrittura in "Team"
- 1 lettura in "allena"
- 1 scrittura in "Coach"

5 letture * 2v/mese = 10 accessi/mese

Analisi dei costi relativa all'operazione 3 :

Modificare il prezzo ed il settore a cui si riferisce il biglietto(3v/mese)

Una modifica di questo tipo implica le seguenti operazioni:

- ***Si restituisce il biglietto di un settore , quindi si decrementa il numero di biglietti venduti per quel determinato settore***
- ***Si concede il biglietto del nuovo settore, quindi si ha un altro accesso per incrementare il numero di biglietti venduti per il nuovo settore***

Con ridondanza :

- 1 lettura in "Location"
- 1 lettura in "Associato"
- 2 scritture in "Biglietto"

6 letture * 3v/mese = 18 accessi/mese

Senza ridondanza :

- 1 scrittura in "Biglietto" (per modificare il settore di riferimento)

2 letture * 3v/mese = 6 accessi/mese

Analisi dei costi relativa all'operazione 4 :

Conteggiare il numero di biglietti venduti per ogni settore (1v/giorno)

Per un diretto confronto con l'analisi delle prime tre operazioni, consideriamo anche la frequenza di questa operazione in accessi al mese . Rapportiamo 1v/g a 30v/mese e proseguiamo con l'analisi dei costi.

Con ridondanza :

- 1 lettura in "Location"
- 1 lettura in "Associato"
- 1 lettura in "Biglietto"

3 letture * 30 v/mese = 90 accessi/mese

Dalla tabella dei volumi è noto che in media ad ogni gara assistono 50.000 persone. Sia X il numero di settori per la location in questione. Si avranno allora circa 50.000/X persone per ogni location, di conseguenza anche 50.000/x biglietti :

Senza ridondanza :

- 1 lettura in "Location"
- 1 lettura in "Associato"
- 50.000/x lettura in "Biglietto"

(50.000/X + 2) letture * 30 v/mese = 1.500.060/X accessi/mese

Conseguenze delle analisi dei costi

Procediamo sommando il numero si accessi e valutando se conviene mantenere o togliere l'attributo ridondante :

Con ridondanza :

- Operazione1 : 112v/mese
- Operazione2 : 10v/mese
- Operazione3 : 18v/mese
- Operazione4 : 90v/mese

Totale = 230 v/mese

Senza ridondanza :

- Operazione1 : 112v/mese
- Operazione2 : 10v/mese
- Operazione3 : 6v/mese
- Operazione4 : 1.500.060/X v/mese

Totale = 1.500.188/X v/mese

Per valutare se conviene mantenere o togliere la ridondanza si risolve la disuguaglianza :

$230 \leq 1.500.188/X$

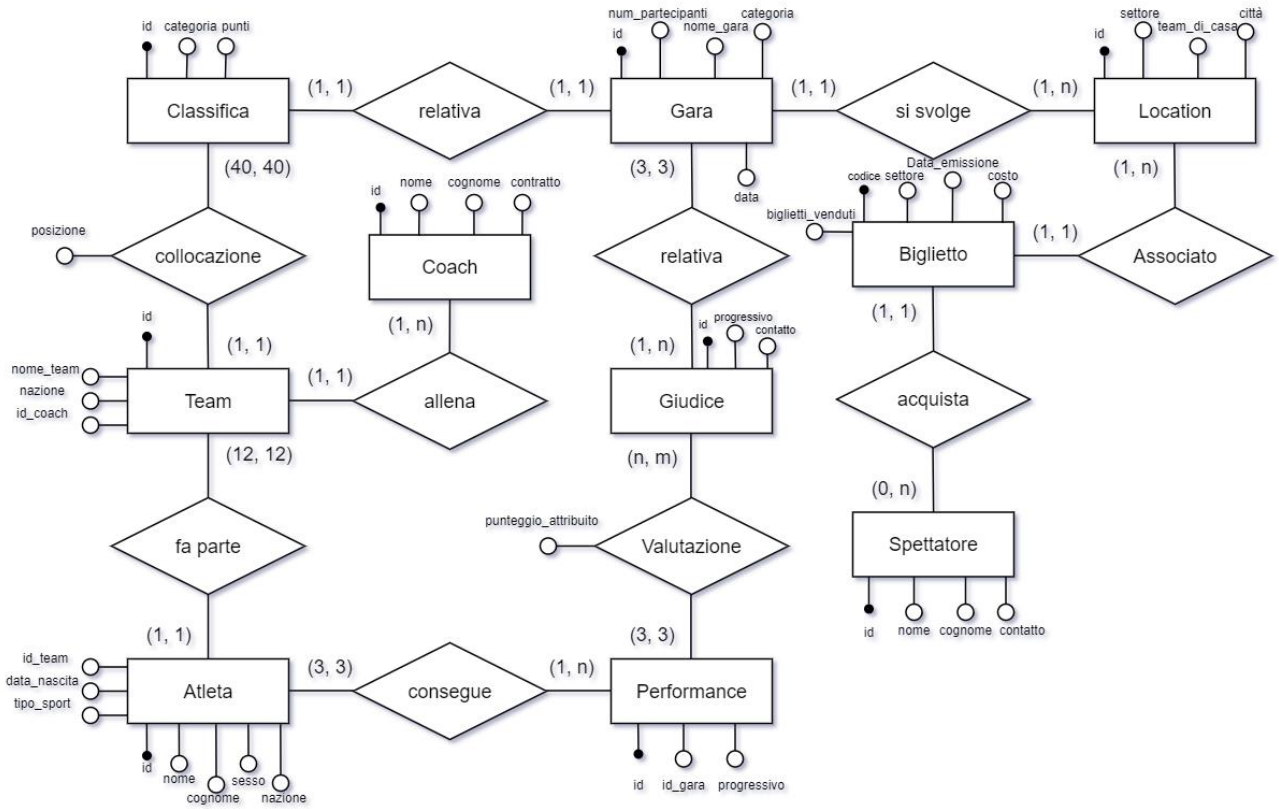
$230\ X \leq 1.500.188$

$X \leq 1.500.188 * 230$

Dato che X rappresenta il numero di settori nella location, che è di certo minore del secondo membro e quindi la disequazione è sempre verificata.

In definitiva conviene mantenere la ridondanza e quindi non si necessita di rimodellare lo schema ER.

Ristrutturazione dello schema ER



Traduzione nel modello logico

- Classifica (**id**, categoria, punti, id_gara)
- Gara (**id**, num_partecipanti, nome_gara, categoria, id_classifica, id_location)
- Collocazione (**team, classifica**, posizione)
- Team (**id**, nome_team, nazione, id_coach)
- Coach (**id**, nome, cognome, contatto)
- Atleta (**id**, nome, cognome, sesso, nazione, id_team, data_nascita, tipo_sport)
- Performance (**id**, progressivo, gara, id_atleta)
- Valutazione (**id_performance, giudice**, progressivo, punteggio_attribuito)
- Giudice (**id**, progressivo, contatto)
- Location (**id**, settore, team_di_casa, città)
- Biglietto (**codice**, biglietti_venduti, settore, data_emissione, costo, location, id_spettatore)
- Spettatore (**id**, nome, cognome, contatto)

Implementazione database in SQL

Creazione della tabella “Classifica”

```
CREATE TABLE IF NOT EXISTS `database`.`Classifica` (
  `id` INT NOT NULL,
  `categoria` VARCHAR(15) NULL DEFAULT NULL,
  `punti` INT NOT NULL,
  `id_gara` INT NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `id_gara`
    FOREIGN KEY (`id_gara`)
    REFERENCES `database`.`Gara` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Team”

```
CREATE TABLE IF NOT EXISTS `database`.`Team` (
  `id` INT NOT NULL,
  `nome_team` VARCHAR(15) NOT NULL,
  `nazione` VARCHAR(15) NULL DEFAULT NULL,
  `id_coach` INT NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `id_coach`
    FOREIGN KEY (`id_coach`)
    REFERENCES `database`.`Coach` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Coach”

```
CREATE TABLE IF NOT EXISTS `database`.`Coach` (
  `id` INT NOT NULL,
  `nome` VARCHAR(15) NULL DEFAULT NULL,
  `cognome` VARCHAR(15) NULL DEFAULT NULL,
  `contatto` VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Atleta”

```
CREATE TABLE IF NOT EXISTS `database`.`Atleta` (
  `id` INT NOT NULL,
  `nome` VARCHAR(15) NULL DEFAULT NULL,
  `congome` VARCHAR(15) NULL DEFAULT NULL,
  `sesso` VARCHAR(10) NULL DEFAULT NULL,
  `nazione` VARCHAR(15) NULL DEFAULT NULL,
  `id_team` INT NOT NULL,
  `data_nascita` DATE NULL DEFAULT NULL,
  `tipo_sport` VARCHAR(15) NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `id_team`
    FOREIGN KEY (`id_team`)
    REFERENCES `database`.`Team` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Performance”

```
CREATE TABLE IF NOT EXISTS `database`.`Performance` (
  `id` INT NOT NULL,
  `gara` INT NULL DEFAULT NULL,
  `progressivo` INT NULL DEFAULT NULL,
  `id_atleta` INT NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `id_atleta`
    FOREIGN KEY (`id_atleta`)
    REFERENCES `database`.`Atleta` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `gara`
    FOREIGN KEY (`gara`)
    REFERENCES `database`.`Gara` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Giudice”

```
CREATE TABLE IF NOT EXISTS `database`.`Giudice` (
  `id` INT NOT NULL,
  `progressivo` INT NULL DEFAULT NULL,
  `contatto` VARCHAR(15) NULL DEFAULT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Giudice”

```
CREATE TABLE IF NOT EXISTS `database`.`Gara` (
  `id` INT NOT NULL,
  `num_partecipanti` BIGINT NULL DEFAULT NULL,
  `nome_gara` VARCHAR(10) NULL DEFAULT NULL,
  `categoria` VARCHAR(15) NULL DEFAULT NULL,
  `id_classifica` INT NULL DEFAULT NULL,
  `id_giudice` INT NULL DEFAULT NULL,
  `id_location` INT NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `id_classifica`
    FOREIGN KEY (`id_classifica`)
    REFERENCES `database`.`Classifica` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `id_giudice`
    FOREIGN KEY (`id_giudice`)
    REFERENCES `database`.`Giudice` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `id_location`
    FOREIGN KEY (`id_location`)
    REFERENCES `database`.`Location` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Biglietto”

```
CREATE TABLE IF NOT EXISTS `database`.`Biglietto` (
  `codice` INT NOT NULL,
  `biglietti_venduti` MEDIUMINT NULL DEFAULT NULL,
  `data_emissione` DATE NULL DEFAULT NULL,
  `settore` VARCHAR(15) NULL DEFAULT NULL,
  `costo` INT NULL DEFAULT NULL,
  `location` INT NULL DEFAULT NULL,
  `id_spettatore` INT NULL DEFAULT NULL,
  PRIMARY KEY (`codice`),
  CONSTRAINT `location`
    FOREIGN KEY (`location`)
    REFERENCES `database`.`Location` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `id_spettatore`
    FOREIGN KEY (`id_spettatore`)
    REFERENCES `database`.`Spettatore` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Location”

```
CREATE TABLE IF NOT EXISTS `database`.`Location` (
  `id` INT NOT NULL,
  `settore` VARCHAR(15) NULL DEFAULT NULL,
  `team_di_casa` VARCHAR(15) NULL DEFAULT NULL,
  `città` VARCHAR(15) NULL DEFAULT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Spettatore”

```
CREATE TABLE IF NOT EXISTS `database`.`Spettatore` (
  `id` INT NOT NULL,
  `nome` VARCHAR(15) NULL DEFAULT NULL,
  `cognome` VARCHAR(15) NULL DEFAULT NULL,
  `contatto` VARCHAR(15) NULL DEFAULT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Collocazione”

```
CREATE TABLE IF NOT EXISTS `database`.`Collocazione` (
  `classifica` INT NOT NULL,
  `team` INT NOT NULL,
  `posizione` INT NOT NULL,
  PRIMARY KEY (`classifica`, `team`),
  CONSTRAINT `team`
    FOREIGN KEY (`team`)
    REFERENCES `database`.`Team` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `classifica`
    FOREIGN KEY (`classifica`)
    REFERENCES `database`.`Classifica` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Creazione della tabella “Valutazione”

```
CREATE TABLE IF NOT EXISTS `database`.`Vautazione` (
  `id_performance` INT NOT NULL,
  `giudice` INT NOT NULL,
  `punteggio_attribuito` INT NULL DEFAULT NULL,
  PRIMARY KEY (`id_performance`, `giudice`),
  CONSTRAINT `id_performance`
    FOREIGN KEY (`id_performance`)
    REFERENCES `database`.`Performance` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `giudice`
    FOREIGN KEY (`giudice`)
    REFERENCES `database`.`Giudice` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = UTF8MB4;
```

Interrogazioni in SQL

Si riportano qui di seguito alcuni esempi di query in SQL volte a testare il corretto funzionamento della base di dati :

1)Trovare le location dove non si è mai tenuta una gara della categoria B

```sql
SELECT l.id, l.settore, l.team_di_casa, l.città
FROM Location l
WHERE NOT EXISTS (
    SELECT *
    FROM Gara g
    WHERE g.id_location = l.id
    AND g.categoria = 'Categoria B'
);
```

2) Selezionare il nome della gara, il nome del giudice e il nome della città della location in cui si è svolta per tutte le gare della categoria C che hanno avuto un giudice con progressivo 2 :

```sql
SELECT G.nome_gara, J.contatto, L.città
FROM Gara G
JOIN Giudice J ON G.id_giudice = J.id
JOIN Location L ON G.id_location = L.id
WHERE G.categoria = 'Categoria C' AND J.progressivo = 2;
```

3) Restituire la città in cui si è svolta la gara con il maggior numero di partecipanti nella categoria "Categoria A" :

```sql
SELECT città
FROM Location
WHERE id = (
    SELECT id_location
    FROM Gara
    WHERE categoria = 'Categoria A' AND num_partecipanti = (
        SELECT MAX(num_partecipanti)
        FROM Gara
        WHERE categoria = 'Categoria A'
    )
)
```

4) Mostra i team di casa delle gare in cui ha partecipato un giudice che ha contato per una gara di categoria A :

```
SELECT DISTINCT team_di_casa
FROM Location
WHERE id IN (
  SELECT id_location
  FROM Gara
  WHERE id_giudice IN (
    SELECT id_giudice
    FROM Gara
    JOIN Classifica ON Gara.id_classifica = Classifica.id
    WHERE Classifica.categoria = 'Categoria A'
  )
);
```

5) Seleziona tutte le città che hanno ospitato gare di tutte le categorie esistenti :

```
SELECT L.città
FROM Location L
GROUP BY L.città
HAVING COUNT(DISTINCT (SELECT G.categoria
                       FROM Gara G
                       WHERE G.id_location = L.id)) = (SELECT COUNT(DISTINCT G.categoria)
                                                       FROM Gara G);
```

6) Selezionare l'atleta che ha preso sempre la valutazione più alta per le proprie performance in tutte le proprie performance , indicandone anche il nome del team di cui fa parte e il nome del coach che lo allena :

```
SELECT A.nome AS nome_atleta, T.nome_team AS nome_team, C.nome AS nome_coach
FROM Atleta A
    JOIN Performance P ON A.id = P.id_atleta
    JOIN Valutazione V ON P.id = V.id_performance
    JOIN (SELECT MAX(punteggio_attribuito) AS max_punteggio, id_performance
          FROM Valutazione
          GROUP BY id_performance) AS V2 ON V.punteggio_attribuito = V2.max_punteggio
                                        AND V.id_performance = V2.id_performance
    JOIN Team T ON A.id_team = T.id
    JOIN Coach C ON T.id_coach = C.id
GROUP BY A.id
HAVING COUNT(*) = (SELECT COUNT(*)
                   FROM Performance
                   WHERE id_atleta = A.id);
```

Analisi ed implementazione dei possibili trigger in SQL

1) Una prima idea potrebbe essere quella di gestire il calcolo dei punteggi degli atleti dopo aver conseguito una performance.
   Segue l'implementazione del trigger "*performance_insert_trigger*" che calcola il punteggio di un atleta e lo inserisce nella classifica ogni volta che viene inserita una nuova performance:

```
CREATE TRIGGER `performance_insert_trigger`
AFTER INSERT ON `performance`
FOR EACH ROW
BEGIN
    DECLARE punti INT;
    SELECT COUNT(*) INTO punti
    FROM Performance
    WHERE gara = NEW.gara AND progressivo <= NEW.progressivo AND id_atleta = NEW.id_atleta;
    UPDATE Classifica
    SET punti = punti + punti
    WHERE id = (SELECT id_classifica
                FROM Gara
                WHERE id = NEW.gara);
END
```

2) Un'altra possibilità riguarda il blocco nell'eliminazione di un atleta . In particolare il trigger "*atleta_delete_trigger*" consente l'eliminazione di un atleta se e solo se esso non ha preso parte a nessuna gara. Segue l'implementazione del trigger :

NB.

Se il valore della variabile "partecipazioni" è maggiore di zero, allora il programma emetterà un segnale SQLSTATE con il codice '45000' e imposterà il messaggio di testo del segnale su "impossibile eliminare l'atleta perché ha partecipato ad almeno una gara"

```
CREATE TRIGGER `atleta_delete_trigger`
BEFORE DELETE ON `Atleta`
FOR EACH ROW
BEGIN
    DECLARE partecipazioni INT;
    SELECT COUNT(*) INTO partecipazioni FROM Performance WHERE id_atleta = OLD.id;
    IF partecipazioni > 0 THEN
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Impossibile eliminare l''atleta perché ha partecipato ad almeno una gara';
    END IF;
END
```

3) Si necessità di creare anche un trigger "*team_insert_trigger*" per l'iscrizione di un nuovo team alla gara:

```
CREATE TRIGGER `team_insert_trigger`
BEFORE DELETE ON `Team`
FOR EACH ROW
BEGIN
    -- incrementa il contatore di team iscritti alla gara corrente
    UPDATE gara_attuale
    SET num_team_iscritti = num_team_iscritti + 1;
END;
```

4) Si implementa un trigger "*coach_update_trigger*" per l'assegnazione di un nuovo coach ad un team :

```
CREATE TRIGGER `coach_update_trigger`
AFTER UPDATE ON Team
FOR EACH ROW
BEGIN
    -- controlla se il nuovo allenatore allena già un altro team
    DECLARE num_team_allenati INT;
    SELECT COUNT(*) INTO num_team_allenati
    FROM Team
    WHERE Coach.id = NEW.Coach.id;

    -- se il nuovo allenatore allena meno di due team, aggiorna la tabella degli allenatori
    IF num_team_allenati < 2 THEN
        UPDATE allenatori
        SET num_team_allenati = num_team_allenati + 1
        WHERE id = NEW.Coach.id;
    END IF;
END;
```

5) Infine si implementa il trigger "*update_classifica_trigger*" per l'aggiornamento delle classifiche alla fine di ogni gara :

```
CREATE TRIGGER `update_classifica_trigger`
AFTER INSERT ON `Collocazione` FOR EACH ROW
BEGIN
    UPDATE Gara
    SET num_partecipanti = num_partecipanti + 1
    WHERE id = (SELECT id_gara FROM Gara WHERE id_classifica = NEW.classifica);

    UPDATE Classifica
    SET punti = (
        SELECT SUM(punti)
        FROM (
            SELECT COUNT(*) AS punti
            FROM Collocazione c
            JOIN Team t ON c.team = t.id
            JOIN Performance p ON t.id = p.id_team AND (SELECT id_gara FROM Gara WHERE id_classifica = c.classifica) = p.gara
            WHERE c.classifica = NEW.classifica AND p.progressivo = 1
            GROUP BY t.id
        ) AS punti_per_team
    )
    WHERE categoria = (SELECT categoria FROM Gara WHERE id_classifica = NEW.classifica)
    AND id_gara = (SELECT id_gara FROM Gara WHERE id_classifica = NEW.classifica);

    UPDATE Team
    SET id_coach = (SELECT id_coach FROM Team WHERE id = NEW.team)
    WHERE id = NEW.team;
END;
```

All'interno del trigger, vengono eseguite tre query di aggiornamento:

- La prima query aggiorna il numero di partecipanti della Gara in cui è stata inserita la nuova collocazione. Viene incrementato il contatore dei partecipanti . La query utilizza l'identificatore della gara associato alla classifica della nuova collocazione.

- La seconda query aggiorna i punteggi nella tabella Classifica. La query calcola i punteggi per ciascuna squadra in base alle performance nella gara associata alla nuova collocazione e alla classifica corrispondente. La query recupera l'identificatore della gara e della categoria dalla tabella Gara utilizzando l'identificatore della classifica nella nuova collocazione.

- La terza query aggiorna l'identificatore dell'allenatore associato alla squadra appena inserita. La query recupera l'identificatore dell'allenatore dalla tabella Team utilizzando l'identificatore della squadra appena inserita.

In sintesi, il trigger aggiorna automaticamente le tabelle correlate (Gara, Classifica, Team) quando viene inserito un nuovo record nella tabella Collocazione.

Implementazione del database in XML

Si genera un file "database.xml" contente la traduzione in linguaggio xml del database creato :

Tabella "Classifica" :

```xml
<table name="Classifica">
  <column name="id" type="integer"/>
  <column name="categoria" type="text"/>
  <column name="punti" type="integer"/>
  <column name="id_gara" type="integer"/>
</table>
```

Tabella "Gara" :

```xml
<table name="Gara">
  <column name="id" type="integer"/>
  <column name="num_partecipanti" type="integer"/>
  <column name="nome_gara" type="text"/>
  <column name="categoria" type="text"/>
  <column name="id_classifica" type="integer"/>
  <column name="id_location" type="integer"/>
</table>
```

Tabella "Collocazione" :

```xml
<table name="Collocazione">
  <column name="team" type="integer"/>
  <column name="classifica" type="integer"/>
  <column name="posizione" type="integer"/>
</table>
```

Tabella "Team" :

```xml
<table name="Team">
  <column name="id" type="integer"/>
  <column name="nome_team" type="text"/>
  <column name="nazione" type="text"/>
  <column name="id_coach" type="integer"/>
</table>
```

Tabella “Coach” :

```
<table name="Coach">
  <column name="id" type="integer"/>
  <column name="nome" type="text"/>
  <column name="cognome" type="text"/>
  <column name="contatto" type="text"/>
</table>
```

Tabella “Atleta” :

```
<table name="Atleta">
  <column name="id" type="integer"/>
  <column name="nome" type="text"/>
  <column name="cognome" type="text"/>
  <column name="sesso" type="text"/>
  <column name="nazione" type="text"/>
  <column name="id_team" type="integer"/>
  <column name="data_nascita" type="date"/>
  <column name="tipo_sport" type="text"/>
</table>
```

Tabella “Performance” :

```
<table name="Performance">
  <column name="id" type="integer"/>
  <column name="progressivo" type="integer"/>
  <column name="gara" type="integer"/>
  <column name="id_atleta" type="integer"/>
</table>
```

Tabella “Valutazione” :

```
<table name="Valutazione">
  <column name="id_performance" type="integer"/>
  <column name="giudice" type="integer"/>
  <column name="progressivo" type="integer"/>
</table>
```

Tabella “Giudice” :

```
<table name="Giudice">
  <column name="id" type="integer"/>
  <column name="progressivo" type="integer"/>
  <column name="contatto" type="text"/>
</table>
```

Tabella “Location” :

```
<table name="Location">
  <column name="id" type="integer"/>
  <column name="settore" type="text"/>
  <column name="team_di_casa" type="integer"/>
  <column name="città" type="text"/>
</table>
```

Tabella “Biglietto” :

```
<table name="Biglietto">
  <column name="codice" type="varchar(255)"/>
  <column name="biglietti_venduti" type="int"/>
  <column name="settore" type="varchar(255)"/>
  <column name="data_emissione" type="date"/>
  <column name="costo" type="decimal(10,2)"/>
  <column name="location" type="int"/>
  <column name="id_spettatore" type="int"/>
</table>
```

Tabella “Spettatore” :

```
<table name="Spettatore">
  <column name="id" type="int"/>
  <column name="nome" type="varchar(255)"/>
  <column name="cognome" type="varchar(255)"/>
  <column name="contatto" type="varchar(255)"/>
</table>
```

Interrogazioni in XML

Come già fatto per SQL , dopo aver costruito il file XML adesso se ne verifica la consistenza attraverso delle interrogazioni :

1)Selezionare il valore del campo "punti" per tutti i record nella tabella "Classifica" in cui la categoria è "Categoria2" :

```
for $record in /database/table[@name="Classifica"]/record[categoria="Categoria2"]
return $record/punti
```

2)Selezionare il nome di tutte le gare che hanno avuto almeno 15 partecipanti:

```
for $gara in /database/table[@name="Gara"]/record[num_partecipanti>=15]
return $gara/nome_gara
```

3)Selezionare il nome della gara e il nome del team vincitore di ogni gara:

```
for $gara in /database/table[@name="Gara"]/record,
$classifica in /database/table[@name="Classifica"]/record[id_gara=$gara/id_classifica],
$collocazione in /database/table[@name="Collocazione"]/row[classifica=$classifica/id and posizione=1],
$team in /database/table[@name="Team"]/row[id=$collocazione/team]
return
    <result>
        <gara>{$gara/nome_gara/text()}</gara>
        <team>{$team/nome_team/text()}</team>
    </result>
```

4)Selezionare la categoria con il maggior numero di punti totali:

```
let $classifica := /database/table[@name='Classifica']/record
let $categorie := distinct-values($classifica/categoria)
for $categoria in $categorie
let $punti_tot := sum($classifica[categoria=$categoria]/punti)
order by $punti_tot descending
return $categoria[1]
```

5)Selezionare il nome e la nazione di tutte le squadre partecipanti alla gara con id 3:

```
for $g in /database/table[@name='Gara']/record[id=3]
return
  for $t in /database/table[@name='Team']/row
  where $t/id = $g/id_classifica
  return <team><nome>{$t/nome_team}</nome><nazione>{$t/nazione}</nazione></team>
```