Università degli Studi di Trieste - Dipartimento di Ingegneria e Architettura

Progetto - Basi di Dati
"Olimpiadi"

Lorenzo Martin Diaz Avalos
IN0501010

Anno Accademico 2022-2023

Indice

1 Progettazione

Ogni due anni si organizza l' edizione delle olimpiadi e si vuole realizzare un semplice database per gestire i dati.

Le olimpiadi si svolgono in un determinato periodo identificato da una data di apertura e chiusura ed è presente una cerimonia di inizio e fine edizione, inoltre hanno un simbolo rappresentativo, identificato tramite la mascotte.
Le edizioni delle olimpiadi sono composte da diversi giochi olimpici, identificati dalla disciplina, che hanno una piccola descrizione del gioco stesso. Ad ogni gioco possono competere più atleti, anche rispetto a più edizioni ed ogni atleta riceve una posizione rispetto alla gara a cui compete (può succedere che più atleti effettuino lo stesso risultato e che quindi abbiano la stessa posizione).

Di ogni atleta si vuole conoscere il nome, cognome, la data di nascita e la nazione da cui proviene, inoltre devono aver compiuto almeno il 18-mo anno di età prima della data di inizio dell' edizione a cui vogliono partecipare.

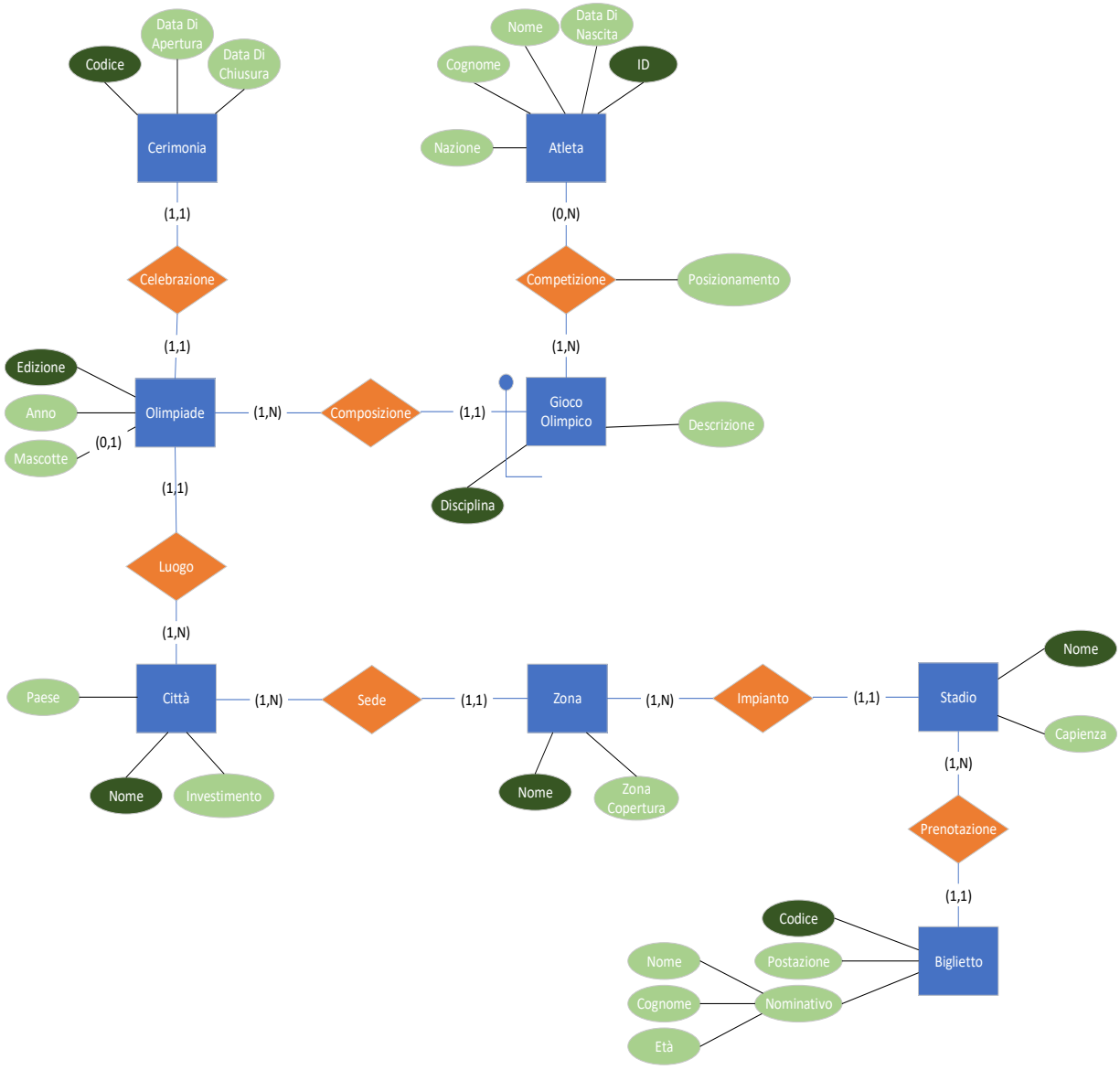
Le olimpiadi hanno luogo in una città, identificata dal loro nome e le città si trovano in un paese. Ogni città mette a disposizione un investimento in denaro per l' organizzazione dell' olimpiade. Le città sono divise per zone, in modo da riuscire a coprire tutta la città e ad assegnare i vari stadi in ogni zona.
Degli stadi si vuole sapere il nome e la capienza massima.

Le postazioni dello stadio possono essere prenotate attraverso i biglietti, che hanno un loro codice identificativo, una loro postazione, che non può superare la capienza dello stadio ed il nominativo della persona che acquista il biglietto, quindi nome, cognome ed età. Solo i maggiorenni possono acquistare il biglietto, quindi gli acquirenti possono acquistare più di un biglietto.

2 Operazioni

1. Si vuole ottenere la classifica delle nazioni con più atleti sul podio rispetto ad una data edizione (una volta ogni due anni)
2. Si vogliono ottenere gli investimenti ed il numero di stadi messi a disposizione da ogni città per ogni edizione (una volta ogni due anni)
3. Si vogliono ottenere il numero di biglietti rimanenti di tutti gli stadi di una data edizione (20 volte al mese)
4. Si vogliono ottenere gli atleti più premiati (sul podio) rispetto a tutte le edizioni (una volta ogni due anni)

3 Diagramma Entity - Relationship



In BLU ci sono le entità, in ARANCIONE le relazioni, in VERDE gli attributi ed in VERDE SCURO gli identificatori delle entità.

4 Dizionario dei dati

4.1 Dizionario delle entità

Entità	Descrizione	Attributi	Identificatore
Olimpiade	Edizione dell' olimpiade	edizione, anno, mascotte	edizione
Cerimonia	Celebrazione di apertura e chiusura	codice, data di apertura, data di chisura	codice
Atleta	Atleti partecipanti all' olimpiade	ID, nome, cognome, nazione, data di nascita	ID
Gioco	Gara di uno sport specifico	disciplina, descrizione	disciplina
Città	Città in cui si svolge l' olimpiade	nome, paese, investimento	nome
Zona	Coperture della città	nome, copertura	nome
Stadio	Impianti a disposizione dell' olimpiade	nome, capienza	nome
Biglietto	Biglietto	codice, postazione, nominativo	codice

4.2 Dizionario delle relazioni

Relazione	Descrizione	Componenti	Attributi
Celebrazione	celebrazione di un' edizione dell' olimpiade	codice cerimonia, edizione olimpiade	-
Composizione	gioco di una specifica edizione dell' olimpiade	edizione olimpiade, disciplina	-
Competizione	Gara a cui partecipano gli atleti	ID, disciplina, edizione olimpiade	posizionamento
Luogo	luogo in cui si svolge una specifica edizione dell' olimpiade	edizione olimpiade, nome città	-
Sede	luoghi di una specifica città	nome città, nome zona	-
Impianto	impianti a disposzione di una specifica zona	nome zona, nome stadio	-
Prenotazione	prenotazione del biglietto per uno specifico stadio	nome stadio, codice biglietto	-

5 Vincoli non esprimibili graficamente

I vincoli non esprimbili graficamente sono:

- Gli atleti devono avere almeno 18 anni di età rispetto all' edizione dell' olimpiade a cui partecipano
- Il numero di biglietti venduti per uno stadio deve essere minore della capienza dello stesso stadio
- Il numero che rappresenta la postazione di un biglietto per uno stadio non può essere maggiore della capienza dello stesso stadio
- Il numero che rappresenta la postazione di un biglietto per uno stadio non può essere duplicato
- L' età di chi acquista un biglietto per un qualsiasi stadio deve essere almeno di 18 anni
- La data di apertura di un' edizione di un' olimpiade deve essere precedente alla data di chiusura della stessa edizione dell' olimpiade
- Il numero che rappresenta il posizionamento di un atleta deve essere positivo

6 Tavola dei volumi

Concetto	Tipo	Volume
Olimpiade	E	33
Celebrazione	R	33
Cerimonia	E	33
Gioco Olimpico	E	660
Composizione	R	660
Atleta	E	40000
Competizione	R	80000
Sede	R	33
Città	E	25
Luogo	R	125
Zona	E	125
Impianto	R	625
Stadio	E	625
Prenotazione	R	6250000
Biglietto	E	6250000

7 Analisi generale

Si può osservare che l' attributo "Nominativo" dell' entità "Biglietto" è un attributo multivalore, contiene altri attributi che, rispetto a questa scelta di modellazione, non sono attributi che riguardano l' entità "Biglietto", quindi si è deciso di creare una nuova entità "Spettatore" che conterrà i suoi attributi "Nome", "Cognome", "Età" e "CF" aggiungendo la relazione "Acquisto" che collega le due entità di modo che ogni spettatore possa acquistare più biglietti ed ogni biglietto è acquistato da un solo spettatore.

Un' altra osservazione riguarda l' attributo "Anno" dell' entità "Olimpiade". Questo attributo si può calcolare direttamente attraverso l' attributo "Data Di Apertura" dell' entità "Cerimonia", perciò sarebbe un campo calcolato e non sarebbe rispettata la terza forma normale. Si è quindi deciso di toglierlo.

Osserviamo poi che l' entità "Gioco Olimpico" è identificato tramite la disciplina e l' identificatore esterno "Olimpiade", questo perché un gioco olimpico può essere ripetuto su più edizioni.

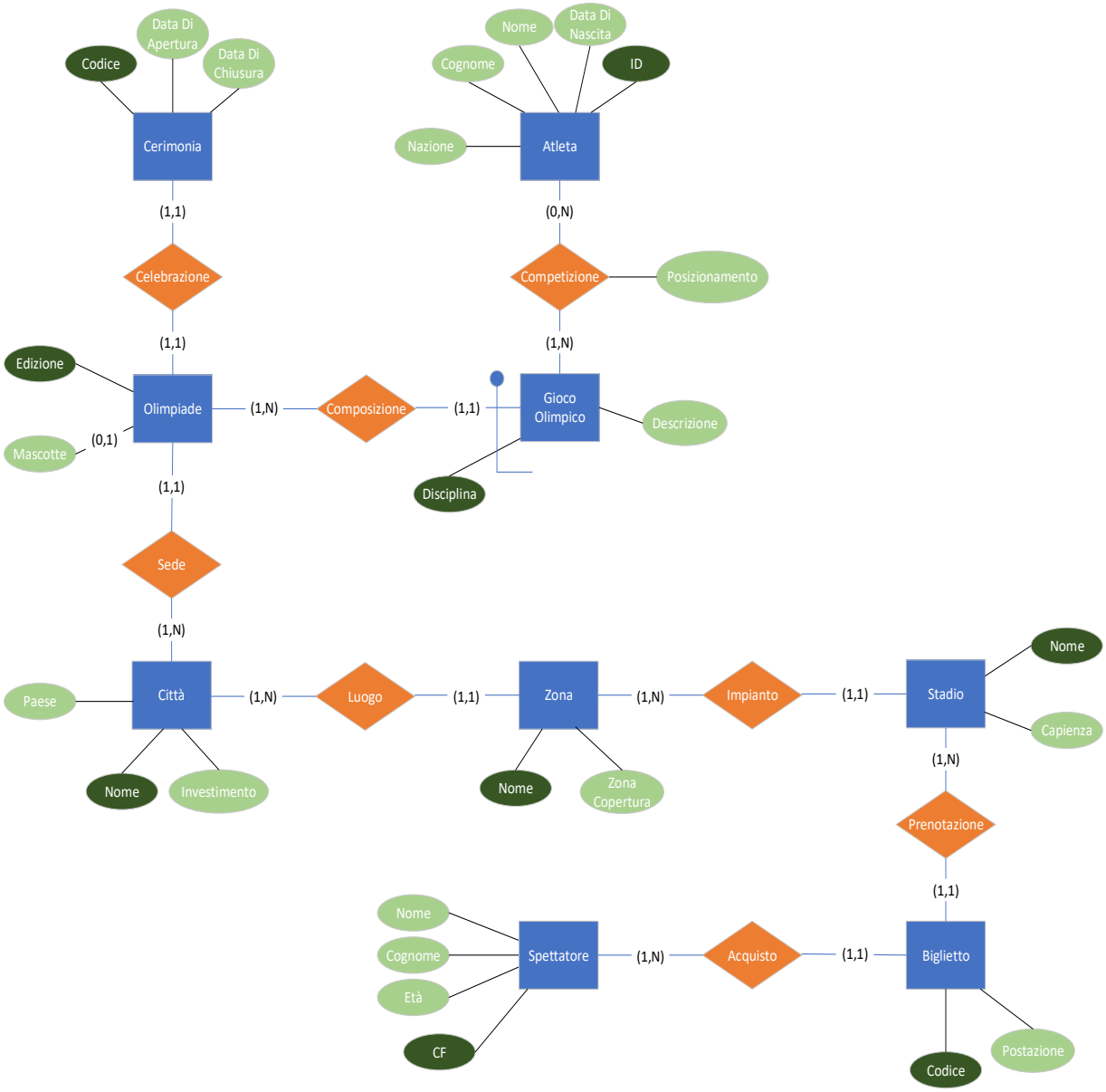
Per quanto riguarda la scelta degli identificatori primari si sono scelti in questo modo:

- Olimpiade: edizione dell' olimpiade
- Cerimonia: codice cerimonia
- Gioco Olimpico: disciplina e Olimpiade a cui riferisce
- Atleta: codice identificativo
- Città: nome della città
- Zona: nome della zona
- Stadio: nome dello stadio
- Biglietto: codice identificativo del biglietto
- Spettatore: codice fiscale dello spettatore

Ulteriore osservazione riguarda l' entità "Biglietto". Dato che l' attributo "Postazione" deve essere non nullo e numericamente diverso per tutti i dati rispetto ad uno specifico stadio, si poteva scegliere di usare l' attributo "Postazione" come chiave primaria insieme all' identificatore esterno attraverso la relazione "Prenotazione" ma si è comunque scelto di usare un codice come chiave primaria per rispettare più una visione reale nella quale un biglietto di uno stadio non è identificato dal numero della postazione ma da un codice univoco.

Riguardo la progettazione fisica, ho deciso di non usare indici secondari dato che non ci sono attributi utilizzati con alta frequenza nelle ricerche.

8 Diagramma Entity - Relationship Ristrutturato



9 Schema Logico

- città (nome, paese, investimento)
- cerimonia (codice, dataApertura, dataChiusura)
- olimpiade (edizione, mascotte, sede, cerimonia)
- zona (nome, zonaCopertura, città)
- stadio (nome, capienza, zona)
- spettatore (CF, nome, cognome, età)
- gioco (disciplina, edizione, descrizione)
- atleta (ID, nome, cognome, nazione, dataNascita)
- biglietto (codice, postazione, stadio, spettatore)

- competizione (atleta, gioco, edizione, posizione)

10 Normalizzazione

Riguardo la normalizzazione si ha che:

- la prima forma normale è rispettata dato che tutte le colonne sono atomiche
- la seconda forma normale è rispettata dato che la prima forma normale è rispettata e ciascuna colonna dipende dalla chiave primaria
- la terza forma normale è rispettata dato che la seconda forma normale è rispettata ed ogni attributo dipende solo dalla chiave primaria

Una spiegazione leggermente più dettagliata del perché la base di dati è in seconda forma normale è data dal fatto che, riguardo la tabella "gioco", questa presenta una chiave primaria composta dagli attributi "disciplina" ed "edizione"; l' attributo "descrizione" dipende dall' attributo "disciplina", in cui si ha una descrizione del gioco ma dipende anche dall' attributo "edizione" dato che può succedere che un gioco, nel corso delle edizioni, può cambiare regole, modalità e quindi a sua volta la descrizione stessa.

Osserviamo invece che la tabella "competizione" ha anch' essa una chiave primaria composta dagli attributi "atleta", "gioco" ed "edizione" e dato che l' atleta può competere sia in più giochi di una stessa edizione che in più edizioni, l' attributo "posizione", che rappresenta la posizone di un atleta nel gioco in cui partecipa, dipende da tutte le componenti della chiave primaria.

11 Query Aggiuntive

```
01 | --SP per ottenere il numero di biglietti rimanenti degli stadi di una
       data edizione
02 |
03 | delimiter &&
04 | create procedure getBigliettiDisponibili(in edizioneInput varchar(8))
05 | begin
06 |     select s.nome as stadio, capienza - count(*) as biglietti_disponibili
07 |     from biglietto
08 |              inner join stadio s
09 |                         on biglietto.stadio = s.nome
10 |              inner join zona z
11 |                         on s.zona = z.nome
12 |              inner join citta c
13 |                         on z.citta = c.nome
14 |              inner join olimpiade o
15 |                         on c.nome = o.sede
16 |     where edizione like edizioneInput
17 |     group by s.nome;
18 | end &&
19 | delimiter ;
20 |
21 |
22 | --SP per ottenere la classifica delle nazioni con piu atleti sul podio in
       una data edizione
23 |
24 | delimiter $$
25 | create procedure getMedagliere(in edizioneInput varchar(8))
26 | begin
27 |     select nazione, count(*) as numMedaglie
28 |     from atleta a
29 |              inner join competizione c
30 |                         on a.ID = c.atleta
31 |              inner join gioco g
32 |                         on ( c.gioco = g.disciplina
33 |                         and c.edizione = g.edizione )
34 |     where ((posizione = 1 or posizione = 2 or posizione = 3)
35 |       and g.edizione like edizioneInput )
36 |     group by nazione
37 |     order by numMedaglie desc;
38 | end $$
39 | delimiter ;
40 |
41 |
42 | --SP per ottenere gli investimenti ed il numero di stadi messi a
       disposizione da ogni citta per ogni edizione
43 |
44 | delimiter $$
45 | create procedure getInvestimenti()
46 | begin
47 |     select edizione, c.nome as citta, investimento,
48 |         count(s.nome) as numeroStadi
49 |     from olimpiade o
```

```
50 |             inner join citta c on o.sede = c.nome
51 |             inner join zona z on c.nome = z.citta
52 |             inner join stadio s on z.nome = s.zona
53 |     group by edizione, c.nome, investimento
54 |     order by edizione;
55 | end $$
56 | delimiter ;
57 |
58 |
59 | --SP per ottenere gli atleti piu premiati (sul podio) in tutte le
       edizioni. Utilizzo una User Defined Function per evitare di ripetere
       righe di codice.
60 |
61 | delimiter $$
62 | create function getNumMedaglie(id varchar(8), posizioneInput int(5))
63 |     returns int(5)
64 |     deterministic
65 | begin
66 |     declare numeroMedaglie int(5);
67 |     select count(*) into numeroMedaglie
68 |     from competizione
69 |     where atleta = id and posizione = posizioneInput;
70 |     return numeroMedaglie;
71 | end $$
72 | delimiter ;
73 |
74 |
75 | delimiter $$
76 | create procedure getAtletiPiuPremiati()
77 | begin
78 |     select concat(a.nome, ' ', a.cognome) as nome,
79 |            getNumMedaglie(a.ID, 1)        as numeroOri,
80 |            getNumMedaglie(a.ID, 2)        as numeroArgenti,
81 |            getNumMedaglie(a.ID, 3)        as numeroBronzi
82 |     from atleta a
83 |              inner join competizione c on a.ID = c.atleta
84 |     where posizione = 1
85 |        or posizione = 2
86 |        or posizione = 3
87 |     group by ID
88 |     order by numeroOri desc, numeroArgenti desc, numeroBronzi desc;
89 | end $$
90 | delimiter ;
91 |
92 |
93 | --Trigger per controllare che non vengano venduti piu biglietti della
       capienza dello stadio
94 |
95 | delimiter $$
96 | create trigger checkCapienza
97 |     before insert
98 |     on biglietto
99 |     for each row
100 | begin
```

```
101 |     declare capienzaStadio int(11);
102 |     declare numeroBiglietti int(11);
103 |     select capienza into capienzaStadio
104 |     from stadio
105 |     where nome like NEW.stadio;
106 |     select count(*) into numeroBiglietti
107 |     from biglietto
108 |     where stadio like NEW.stadio;
109 |     if (numeroBiglietti >= capienzaStadio) then
110 |         set @signal = concat('Capienza piena per lo stadio ', NEW.stadio)
    ;
111 |         SIGNAL SQLSTATE '45001' SET MESSAGE_TEXT = @signal;
112 |     end if;
113 | end $$
114 | delimiter ;
115 |
116 |
117 | --Trigger per controllare che il numero della postazione di un biglietto
    non sia maggiore della capienza dello stadio
118 |
119 | delimiter $$
120 | create trigger checkPostazione
121 |     before insert
122 |     on biglietto
123 |     for each row
124 | begin
125 |     declare capienzaStadio int(11);
126 |     select capienza into capienzaStadio
127 |     from stadio
128 |     where nome like NEW.stadio;
129 |     if (NEW.postazione > capienzaStadio) then
130 |         set @signal = concat('La postazione ', NEW.postazione,
131 |                         ' per lo stadio ', NEW.stadio, ' non e valida');
132 |         SIGNAL SQLSTATE '45001' SET MESSAGE_TEXT = @signal;
133 |     end if;
134 | end $$
135 | delimiter ;
136 |
137 |
138 | --Trigger per controllare che un atleta abbia almeno 18 anni rispetto
    alla data di inizio dell edizione in cui compete
139 |
140 | delimiter $$
141 | create trigger checkAge
142 |     before insert
143 |     on competizione
144 |     for each row
145 | begin
146 |     declare dataNascitaAtleta date;
147 |     declare dataEdizione date;
148 |     select dataNascita into dataNascitaAtleta
149 |     from atleta
150 |     where id = NEW.atleta;
151 |     select dataApertura
```

```
152 |     into dataEdizione
153 |     from cerimonia c
154 |              inner join olimpiade o on c.codice = o.cerimonia
155 |     where o.edizione like NEW.edizione;
156 |     if (year(dataEdizione) - year(dataNascitaAtleta) < 18) then
157 |         set @signal =
158 |                 concat('L atleta ', (select concat(a.nome, ' ', a.cognome
    )
159 |                                      from atleta a
160 |                                      where id like NEW.atleta),
161 |                        ' e troppo giovane per competere nell edizione ',
162 |                        (select edizione
163 |                        from olimpiade o
164 |                        where o.edizione like NEW.edizione), '. Tra ',
165 |                        (18 - (year(sysdate()) - year(dataNascitaAtleta)))
    ,
166 |                        ' anni potra partecipare alla prossima edizione');
167 |         SIGNAL SQLSTATE '45001' SET MESSAGE_TEXT = @signal;
168 |     end if;
169 | end $$
170 | delimiter ;
171 |
172 |
173 | --Trigger per controllare che una postazione di un biglietto per uno
    stadio non sia duplicata
174 |
175 | delimiter $$
176 | create trigger checkPostazioniDuplicate
177 |     before insert
178 |     on biglietto
179 |     for each row
180 | begin
181 |     set @postazione = -1;
182 |     select postazione into @postazione
183 |     from biglietto
184 |     where postazione = NEW.postazione and stadio = NEW.stadio;
185 |     if (@postazione > 0) then
186 |         set @signal = concat('La postazione ', NEW.postazione,
187 |                          ' per lo stadio ', NEW.stadio,
188 |                          ' e gia occupata. Trovare un altra postazione!');
189 |         SIGNAL SQLSTATE '45001' SET MESSAGE_TEXT = @signal;
190 |     end if;
191 |
192 | end $$
193 | delimiter ;
194 |
195 |
196 | -- Vista per ottenere gli atleti che hanno partecipato alle edizioni
    delle olimpiadi, la relativa nazione, disciplina e posizione di gara
197 |
198 | create view atletiPartecipanti as
199 | select concat(a.nome, ' ', a.cognome) as atleta, nazione,
200 |             disciplina, posizione, c.edizione
201 | from atleta a
```

```
202 |          inner join competizione c on a.ID = c.atleta
203 |          inner join gioco g on (c.gioco = g.disciplina
204 |                                 and c.edizione = g.edizione)
205 | order by edizione, disciplina, posizione;
206 |
207 |
208 | -- Crezione della tabella "spettatore" per soddisfare il vincolo non
       esprimibile graficamente in cui l eta di chi acquista un biglietto per
        un qualsiasi stadio deve essere almeno di 18 anni
209 |
210 | drop table if exists spettatore;
211 |
212 | create table spettatore (
213 |     CF      varchar(20) not null,
214 |     nome    varchar(20) not null,
215 |     cognome varchar(20) not null,
216 |     eta     int(11)     not null,
217 |     primary key (CF),
218 |     constraint checkSpettatoreAge check (eta >= 18)
219 | );
220 |
221 |
222 | -- Crezione della tabella "cerimonia" per soddisfare il vincolo non
       esprimibile graficamente in cui la data di apertura di un edizione di
       un olimpiade deve essere precedente alla data di chiusura della stessa
        edizione dell olimpiade
223 |
224 | drop table if exists cerimonia;
225 |
226 | create table cerimonia (
227 |     codice       varchar(8) not null,
228 |     dataApertura date       not null,
229 |     dataChiusura date       not null,
230 |     primary key (codice),
231 |     constraint cerimonia_ibfk_1 check ( dataApertura < dataChiusura )
232 | );
233 |
234 |
235 | -- Crezione della tabella "competizione" per soddisfare il vincolo non
       esprimibile graficamente in cui il numero che rappresenta il
       posizionamento di un atleta deve essere positivo
236 |
237 | drop table if exists competizione;
238 |
239 | create table competizione (
240 |     atleta    varchar(8)  not null,
241 |     gioco     varchar(20) not null,
242 |     edizione  varchar(8)  not null,
243 |     posizione int(11)     not null,
244 |     primary key (atleta, gioco, edizione),
245 |     constraint competizione_ibfk_1 foreign key (atleta) references atleta
        (ID),
246 |     constraint competizione_ibfk_2 foreign key (gioco, edizione)
      references gioco (disciplina, edizione),
```

```
247 |     constraint competizione_ibfk_3 check ( posizione > 0 )
248 | );
```