ESAME_07_2021

Esercizi SQL

Dato il seguente schema logico relazionale

EDITORI (codice, nome, indirizzo, citta)
LIBRERIE (codice, nome, indirizzo, citta)
AUTORI (nome, nascita, morte*, nazione)
PUBBLICAZIONI (codice, titolo, data_stampa, autore_{fk}, editore_{fk})
VENDITE (librerie_{fk}, $\text{pubblicazione}_{fk}$, data, copie_vendute)

codificare le seguenti richieste in linguaggio SQL.

1. Scrivere l'istruzione DDL per la creazione della relazione **PUBBLICAZIONI**, codificando tutti i vincoli indicati nello schema.

```sql
CREATE TABLE pubblicazioni (
   codice NUMERIC NOT NULL PRIMARY KEY,
   titoli VARCHAR(200) NOT NULL,
   data_stampa DATE NOT NULL,
   autore VARCHAR(100) NOT NULL REFERENCES autori(nome)
   editore NUMERIC NOT NULL REFERENCES editori(codice));
```

2. Modificare la relazione **LIBRERIE** per impedire che possano esistere librerie con lo stesso nome situate nella stessa citta'.

```sql
ALTER TABLE librerie
ADD CONSTRAINT nome_citta_uq
UNIQUE(nome,citta)
```

3. Per ogni citta, mostrare il numero di pubblicazioni vendute nelle librerie di quella citta nell'anno solare 2020 (non occorre elencare le citta nelle quali non si e' venduta alcuna pubblicazione).

```sql
SELECT lib.citta SUM(v.copie_vendute)
FROM librerie lib, vendite v
WHERE v.libreria = lib.codice
   AND v.data BETWEEN '2020-01-01' AND '2020-31-12';
   -- EXTRACT (GROUP BY lib.citta);
```

4. Per ogni editore, indicare il numero di autori ancora in vita (con riferimento alla data odierna) che hanno pubblicato almeno una volta con quell'editore dal 2020 a oggi; ordinare il risultato in base al nome dell'editore.

```sql
SELECT e.nome, COUNT(a.nome), AS numero_autori_in_vita
FROM editori e, autori a, pubblicazioni p
WHERE e.codice = p.editore
   AND a.nome = p.autore
   AND a.morte IS NULL
   AND p.data ⩾ '2000-01-01'
GROUP BY e.nome
ORDER BY e.nome;
```

5. Eliminare dal database gli editori che non hanno associata alcuna pubblicazione.

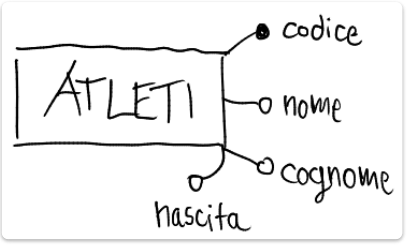
```sql
DELETE FROM editori
WHERE codice NOT IN (SELECT editore FROM pubblicazioni);
-- WHERE NOT EXISTS (SELECT * FROM pubblicazioni p, editori e
--                                     WHERE p.editori = e.codice);
```

Esercizi E-R

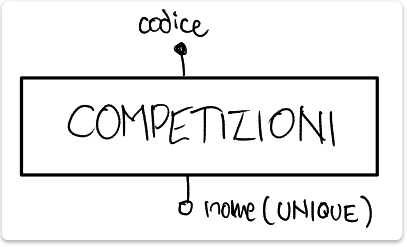
Mostrare lo schema concettuale Entita-Relazione per un database che codifica informazioni relative a varie edizioni di una manifestazione sportiva. Si richiede di modellare le informazioni seguenti.

1. Ogni *edizione* della manifestazione sportiva e' identificata dall'anno di svolgimento; si tiene anche traccia delle date d'inizio e fine dell'edizione.

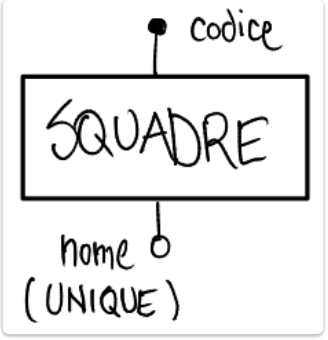
2. Gli *atleti* sono identificati da un codice; per essi si registrano nome, cognome e dati di nascita.



3. Le *competizioni* sportive sono identificate da un codice e caratterizzate da un nome, anche esso univoco; si considerano solo competizioni individuali.



4. Le *squadre*, identificate da un codice, sono dotate di un nome, anche esso univoco.

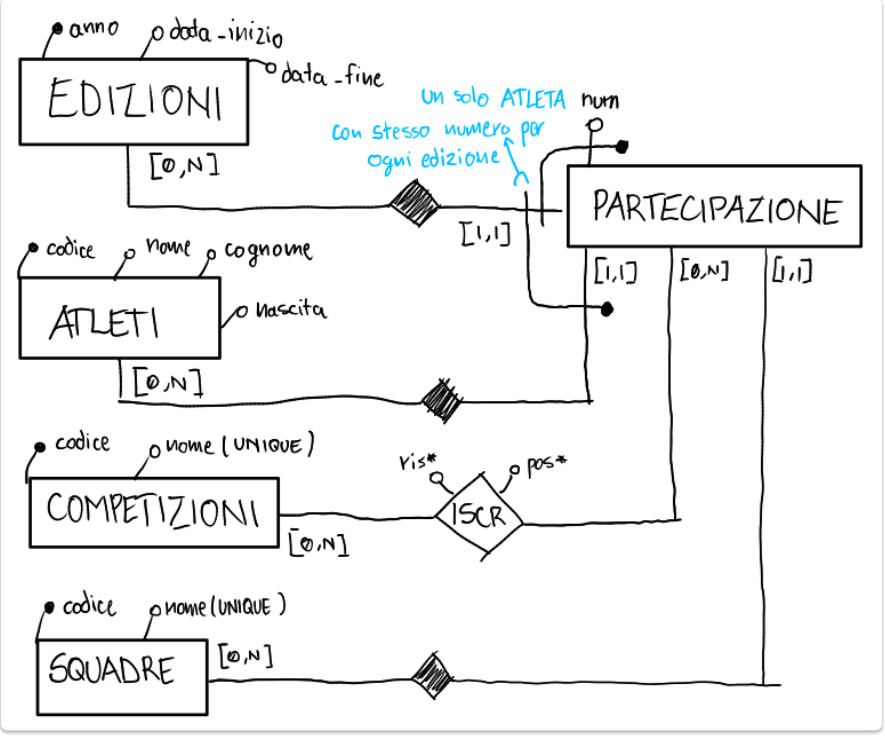


5. *Per ogni edizione* della manifestazione, si registrano le informazioni seguenti:

    1. La *partecipazione* degli atleti a quell'edizione; ogni atleta puo' partecipare a piu' edizioni, ma al piu' una volta per ogni edizione; la partecipazione e' identificata dall'edizione e da un numero progressivo

(il numero di pettorina dell'atleta in quella edizione); ogni partecipazione e' associata ad una sola squadra (un atleta puo' far parte di squadre diverse, ma in edizione diverse).

2. Ogni partecipante ad una edizione puo' *iscriversi* ad una o piu' competizioni distinte; un atleta che partecipa a piu' edizioni puo' iscriversi piu' volte alla stessa competizione.
3. Per ogni iscrizione si possono registrare il *risultato*, cioe' una misura della prestazione ottenuta, e la *posizione* in classifica; entrambi opzionali.



Traduzione E-R in logico relazionale

Tradurre lo schema ER in uni schema logico relazionale, codificando opportunamente i vincoli dello schema.

`EDIZIONI` (anno, inizio, fine)
`ATLETI` (codice, nome, cognome, nascita)
`SQUADRE` (codice, $\text{nome}_{\text{UNIQUE}}$)
`COMPETIZIONI` (codice, $\text{nome}_{\text{UNIQUE}}$)
`PARTECIPAZIONE` ([$\text{edizione}_{\text{fk}}$, num], [$\text{edizione}_{\text{fk}}$, $\text{atleta}_{\text{fk}}$]$_{\text{UNIQUE}}$, $\text{squadra}_{\text{fk}}$)
`ISCRIZIONI` (competizione$_{\text{fk}}$. [$\text{edizione}_{\text{fk}}$, num_{fk}], ris*, pos*)

`NOTE` : le associazioni *molti-a-molti* vanno sempre codificate, inoltre, le relazioni associate diventano attributi `PRIMARY KEY` e `FOREIGN KEY` automaticamente.

Esercizi teoria

1. Data la relazione $R(X)$, sotto quali condizioni l'operatore di proiezione $\pi_Y(R)$ e' ben definito? Sotto quali condizioni tale espressione dell'algebra relazionale e' equivalente all'istruzione SQL `SELECT Y FROM R`?

   $\pi_Y(R)$: Y e' un sottoinsieme di X, l'eliminazione dei duplicati non avviene automaticamente; nell'algebra relazionale avviene fintanto che Y superchiave di R.

2. Spiegare brevemente in cosa consistono le anomalie dette *lettura sporca* e *lettura non ripetibile*.

   Nella *lettura sporca*, la transazione T_1 ha letto un dato x in corso di manipolazione dalla transazione T_2 che tuttavia non ha ancora fatto `commit`.

Nella *lettura non ripetibile*, viene riletto un x che pero' viene successivamente modificato da una transazione. (Entrambi i casi sono una violazione dell'isolamento)

3. Spiegare brevemente come si costruisce il grafo dei conflitti a partire da uno schedule di un insieme di transazioni; per cosa puo' essere utile tale grafo?

   Un grafo dei conflitti e' utile allo scopo d'identificare se uno schedule di transazioni e' serializzabile o meno in base ai conflitti.