Soluzioni esami 21/01/2020, 04/06/2020, 18/06/2020

Alessandra Raffaetà

Esercizio 1. Si vuole costruire una base di dati per memorizzare informazioni su un museo d'arte. Il museo ha una collezione di oggetti d'arte. Ogni oggetto d'arte ha un numero indetificativo, un autore (se noto), un anno (in cui è stato creato), un titolo e una descrizione. Gli oggetti d'arte sono classificati in base al loro tipo. I tre tipi principali sono: Dipinto, Scultura e Statua, più un altro tipo detto Altro per accogliere oggetti che non rientrano in uno dei tre tipi principali. Un dipinto ha un tipo-pittura (olio, acquarello, etc), un materiale su cui è stato steso (carta, tela di canapa, legno, etc.), e uno stile (moderno, astratto etc.). Una scultura o una statua ha un materiale con cui è stata creata (legno, pietra, etc.), un'altezza, un peso e uno stile. Un oggetto d'arte nella categoria Altro ha un tipo (stampa, foto etc) e uno stile. Il museo memorizza informazioni sugli artisti: nome, data di nascita, data di morte (se non è vivente), paese di origine e lo stile principale. Nel museo sono organizzate delle esposizioni, ognuna delle quali ha un nome, una data di inizio e una data di fine, ed è collegata a tutti gli oggetti d'arte che sono stati esposti durante l'esposizione.

Si dia uno schema grafico a oggetti (secondo la notazione del libro di testo) della base di dati e si trasformi nello schema relazionale mostrandone la rappresentazione grafica (anche questa secondo la notazione del libro di testo, indicando la chiave primaria e le chiavi esterne). Sia per lo schema a oggetti che per lo schema relazionale si devono specificare, rispettivamente, i nomi e i tipi degli attributi di ciascuna classe e relazione.

Soluzione: Lo schema grafico a oggetti è illustrato in Figura 5.

La trasformazione dello schema a oggetti nello schema relazionale è mostrato in Figura 6.

Esercizio 2. Dati i seguenti schemi di relazione (le chiavi primarie sono sottolineate, le chiavi esterne sono date esplicitamente), che sono intesi rappresentare informazioni su di una edizione dei campionati mondiali.

- Squadre(Nazione, Allenatore)
- Partite(IdP, Squadra1*, Squadra2*)
  Squadra1, Squadra2 FK(Squadre)
- Reti(IdR, IdG*, IdP*, Auto)
  IdG FK(Giocatori), IdP FK(Partite)
- Giocatori(IdG, Nome, Nazione*, Nascita)
  Nazione FK(Squadre)

L'attributo `Auto` è un booleano che indica se la rete è o meno un autogoal.

(i) Scrivere in SQL le seguenti interrogazioni:

1. trovare le squadre contro cui ha giocato l'Italia;
2. trovare il nome dei giocatori della nazionale brasiliana che non hanno segnato alcun autogoal;
3. trovare le squadre e gli allenatori che hanno giocato il massimo numero di partite;
4. trovare per ogni squadra quanti giocatori hanno segnato un goal escludendo gli autogoal. Se una squadra ha segnato 0 goal deve ritornare 0 giocatori.
5. ritornare le partite concluse in parità, non zero a zero (ovvero con qualche rete);

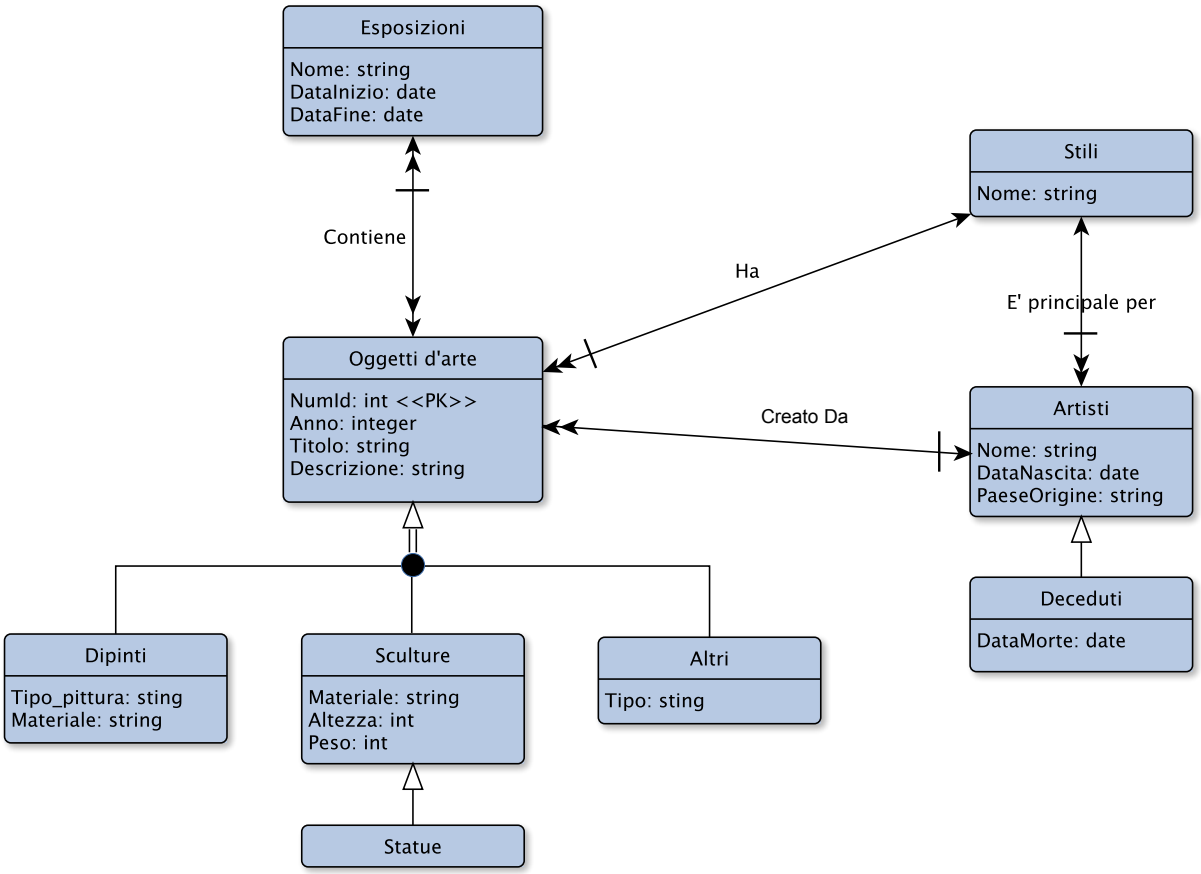

Figura 1: Schema a oggetti

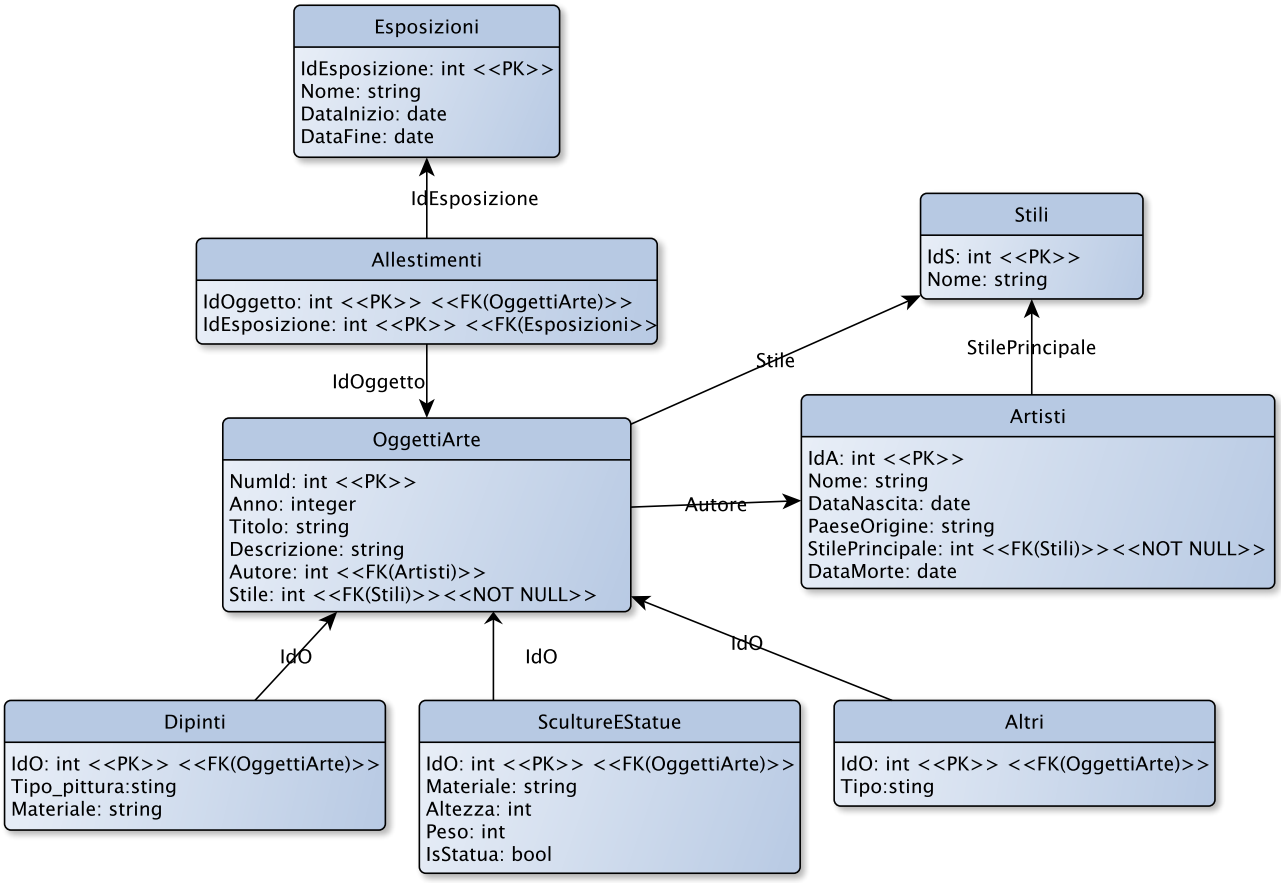

Figura 2: Schema relazionale

(*ii*) scrivere in Algebra relazionale la prima interrogazione.

Soluzione:

1. Trovare le squadre contro cui ha giocato l'Italia.

```
SELECT p.Squadra2 AS Avversario
FROM   Partite p
WHERE p.Squadra1 = 'Italia'
UNION
SELECT p.Squadra1 AS Avversario
FROM   Partite p
WHERE p.Squadra2 = 'Italia'
```

$$\rho_{\text{Squadra1}\to\text{Avversario}}(\pi_{\text{Squadra1}}(\sigma_{\text{Squadra2='Italia'}}(\text{Partite}))) \cup \rho_{\text{Squadra2}\to\text{Avversario}}(\pi_{\text{Squadra2}}(\sigma_{\text{Squadra1='Italia'}}(\text{Partite})))$$

2. Trovare il nome dei giocatori della nazionale brasiliana che non hanno segnato alcun autogoal.

```
SELECT g.Nome
FROM   Giocatori g
WHERE g.Nazione = 'Brasile' AND g.IdG NOT IN (SELECT r.IdG
                                              FROM   Reti r
                                              WHERE r.Auto)
```

3. Trovare le squadre e gli allenatori che hanno giocato il massimo numero di partite.

```
CREATE VIEW NumeroPartite(Nazione, Allenatore, NumP)
AS SELECT     s.Nazione, s.Allenatore, COUNT(*)
   FROM       Squadre s JOIN Partite p ON (s.Nazione = p.Squadra1) OR
                                          (s.Nazione = p.Squadra2)
   GROUP BY s.Nazione, s.Allenatore

SELECT np.Nazione, np.Allenatore
FROM   NumPartite np
WHERE np.NumP = (SELECT MAX(NumP)
                 FROM   NumPartite)
```

4. Trovare per ogni squadra quanti giocatori hanno segnato un goal escludendo gli autogoal. Se una squadra ha segnato 0 goal deve ritornare 0 giocatori.

```
SELECT     s.Nazione, COUNT(DISTINCT r.IdG) AS NumG
FROM       Squadre s NATURAL LEFT JOIN
           (Giocatori g JOIN Reti r ON g.IdG = r.IdG AND NOT r.Auto)
GROUP BY s.Nazione
```

   Poiché la squadra è l'insieme dei suoi giocatori, la query può essere semplificata in questo modo:

```
SELECT     g.Nazione, COUNT(DISTINCT r.IdG) AS NumG
FROM       Giocatori g LEFT JOIN Reti r ON g.IdG = r.IdG AND NOT r.Auto
GROUP BY g.Nazione
```

5. Ritornare le partite concluse in parità, non zero a zero (ovvero con qualche rete).

```
CREATE VIEW RisultatoPartite(Partita, Sq1, NumGoalSq1, Sq2, NumGoalSq2)
AS SELECT    p.IdP, p.Squadra1,
             SUM(CASE WHEN (NOT r.Auto AND g.Nazione = p.Squadra1) OR
             (r.Auto AND g.Nazione = p.Squadra2) THEN 1 ELSE 0 END),
             p.Squadra2,
             SUM(CASE WHEN (NOT r.Auto AND g.Nazione = p.Squadra2) OR
             (r.Auto AND g.Nazione = p.Squadra1) THEN 1 ELSE 0 END),
  FROM       Partite p NATURAL JOIN Reti r NATURAL JOIN Giocatori g
  GROUP BY p.IdP, p.Squadra1, p.Squadra2

SELECT rp.Partita
FROM   RisultatoPartite rp
WHERE rp.NumGoalSq1 = rp.NumGoalSq2
```

Esercizio 3. Si vuole costruire una base di dati per la gestione di una scuola di canto. La scuola di canto dà la possibilità di certificare i livelli di preparazione per ciascun studente. Ogni livello è identificato da un codice e possiede una descrizione che specifica gli obiettivi didattici che devono essere raggiunti. Per ogni livello ci sono dei brani musicali che lo studente dovrà interpretare per poter ricevere il certificato che attesta di aver raggiunto un certo livello di preparazione. Per ogni studente della scuola, identificato dal codice fiscale, è necessario sapere quando è stato superato un certo livello di preparazione. Ogni studente ha dei recapiti telefonici per contattarlo e un proprio repertorio di brani, che non necessariamente devono coincidere con quelli dei livelli di preparazione. Ogni brano deve avere il titolo, la durata e l'autore che l'ha composto. Gli studenti della scuola si possono esibire in alcuni locali, di cui si vuole memorizzare il nome e l'indirizzo e le serate organizzate. Per ogni studente si vuole registrare la serata in cui si è esibito e le canzoni che ha cantato. Per ogni serata si vuole sapere il numero di consumazioni vendute per poter permettere al locale di richiamare i cantanti che attirano più pubblico.

Si dia uno schema grafico a oggetti (secondo la notazione del libro di testo) della base di dati e si trasformi nello schema relazionale mostrandone la rappresentazione grafica (anche questa secondo la notazione del libro di testo, indicando la chiave primaria e le chiavi esterne). Sia per lo schema a oggetti che per lo schema relazionale si devono specificare, rispettivamente, i nomi e i tipi degli attributi di ciascuna classe e relazione.

Soluzione: Lo schema grafico a oggetti è illustrato in Figura 5.

La trasformazione dello schema a oggetti nello schema relazionale è mostrato in Figura 6.

Esercizio 4. Dati i seguenti schemi di relazione riguardanti gli atleti e le medaglie vinte alle olimpiadi (le chiavi primarie sono sottolineate, le chiavi esterne sono date esplicitamente):

- Atleti(<u>IdAtleta</u>, Nome, Cognome, Nazione*, Sesso)
  Nazione FK(Nazioni)
- Medaglie(<u>Codice</u>, Tipo, Sport, Disciplina, IdAtleta*, Anno)
  IdAtleta FK(Atleti)
- Nazioni(<u>Nome</u>, Estensione, NumAbitanti)

(i) Scrivere in SQL le seguenti interrogazioni:

1. trovare il numero di medaglie vinte dall'Italia nell'anno 2016;
2. trovare il nome, cognome e la nazione degli atleti cha hanno vinto almeno due medaglie in sport differenti;
3. trovare la nazione con il minor numero di abitanti che ha vinto almeno una medaglia d'oro femminile nello sport Nuoto;

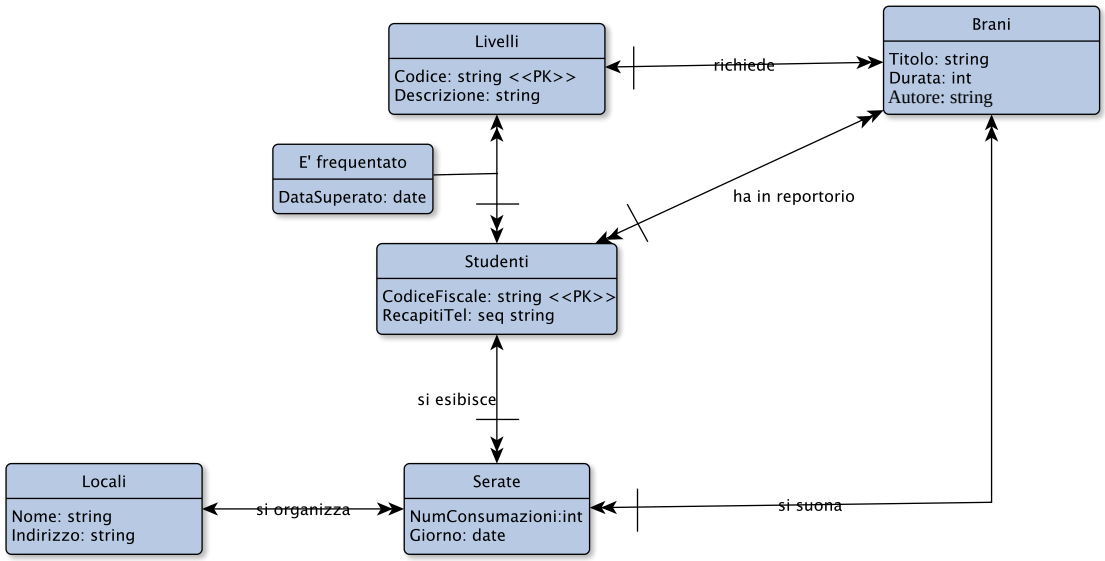

Figura 3: Schema a oggetti

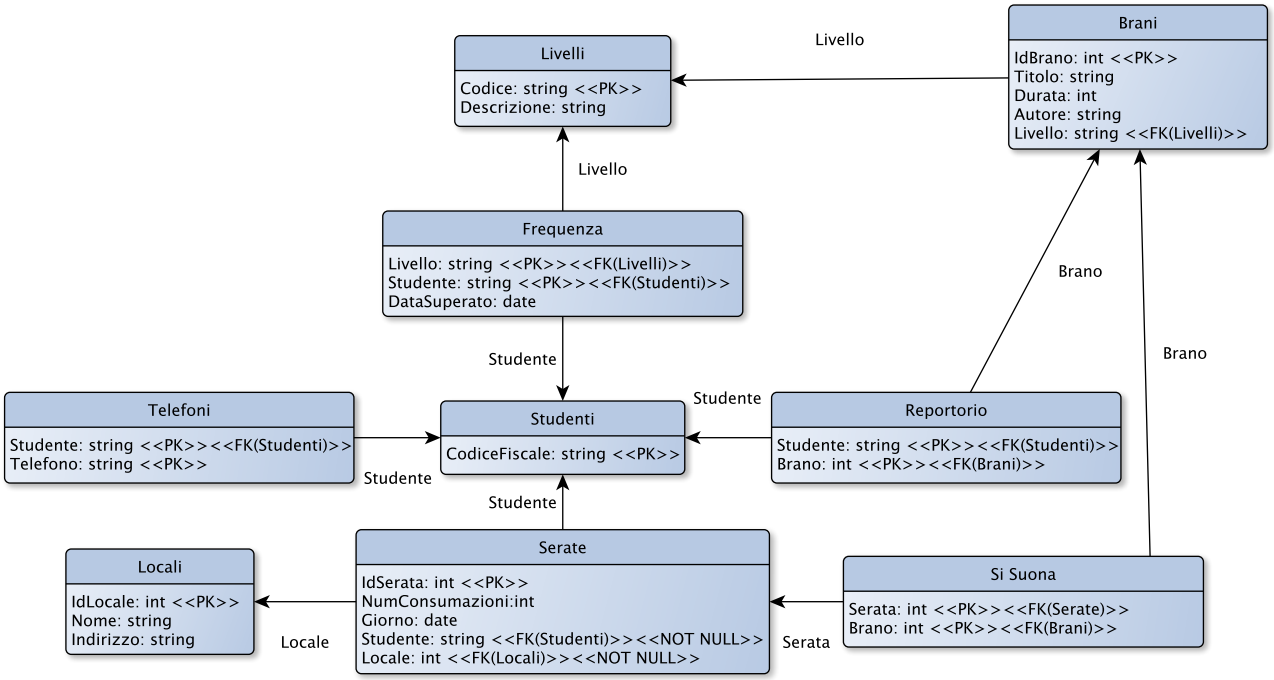

Figura 4: Schema relazionale

4. per ogni sport, trovare il nome e il cognome dell'atleta che ha vinto più medaglie e il numero di edizioni (anni) in cui le ha vinte.
5. per ogni nazione restituire il numero di medaglie d'oro, d'argento e di bronzo vinte negli anni dal 1960 al 2000. Se una nazione non ha vinto medaglie di un certo tipo restituire 0.

(*ii*) scrivere in Algebra relazionale la prima interrogazione.

Soluzione:

1. Trovare il numero di medaglie vinte dall'Italia nell'anno 2016.

```
SELECT COUNT(*)
FROM   Atleti a NATURAL JOIN Medaglie m
WHERE a.Nazione = 'Italia' AND m.Anno = 2016
```

$$\gamma count(*)(\sigma_{\text{Nazione='Italia'} \wedge \text{Anno=2016}}(\text{Atleti} \bowtie \text{Medaglie}))$$

2. Trovare il nome, cognome e la nazione degli atleti cha hanno vinto almeno due medaglie in sport differenti.

```
SELECT DISTINCT a.IdAtleta, a.Nome, a.Cognome, a.Nazione
FROM   Atleti a NATURAL JOIN Medaglie m1
       JOIN Medaglie m2 USING (IdAtleta)
WHERE m1.Sport <> m2.Sport
```

3. Trovare la nazione con il minor numero di abitanti che ha vinto almeno una medaglia d'oro femminile nello sport Nuoto.

```
WITH NazioneDonneOroNuoto AS (
        SELECT DISTINCT n.Nome, n.NumAbitanti
        FROM   Atleti a NATURAL JOIN Medaglie m
               JOIN Nazioni n ON a.Nazione = n.Nome
        WHERE a.Sesso = 'F' AND m.Tipo = 'Oro' AND m.Sport = 'Nuoto'
     )
SELECT n.Nome
FROM   NazioneDonneOroNuoto nd
WHERE nd.NumAbitanti = (SELECT MIN(NumAbitanti)
                        FROM   NazioneDonneOroNuoto)
```

4. Per ogni sport, trovare il nome e il cognome dell'atleta che ha vinto più medaglie e il numero di edizioni (anni) in cui le ha vinte.

```
CREATE VIEW MedSportAtleta(Sport, IdAtleta, NumM, NumEdizioni)
AS SELECT    m.Sport, m.IdAtleta, COUNT(*), COUNT(DISTINCT m.Anno)
   FROM      Medaglie m
   GROUP BY m.Sport, m.IdAtleta

SELECT a.Nome, a.Cognome, ms.NumEdizioni
FROM   MedSportAtleta ms NATURAL JOIN Atleti a
WHERE ns.NumM = (SELECT MAX(NumM)
                 FROM   MedSportAtleta ms1
                 WHERE ms1.Sport = ms.Sport)
```

5. Per ogni nazione restituire il numero di medaglie d'oro, d'argento e di bronzo vinte negli anni dal 1960 al 2000. Se una nazione non ha vinto medaglie di un certo tipo restituire 0.

```
SELECT    a.Nazione,
          SUM(CASE WHEN (m.Tipo IS NOT NULL AND m.Tipo = 'Oro')
          THEN 1 ELSE 0 END) AS Oro,
          SUM(CASE WHEN (m.Tipo IS NOT NULL AND m.Tipo = 'Argento')
          THEN 1 ELSE 0 END)AS Argento,
          SUM(CASE WHEN (m.Tipo IS NOT NULL AND m.Tipo = 'Bronzo')
          THEN 1 ELSE 0 END) AS Bronzo,
FROM      Atleti a LEFT JOIN Medaglie m ON a.IdAtleti = m.IdAtleta AND
          m.Anno BETWEEN 1960 AND 2000
GROUP BY  a.Nazione
```

Esercizio 5. Si vuole costruire una base di dati per la gestione degli stabilimenti balneari per la stagione 2020. Uno stabilimento balneare ha un nome, un indirizzo, e una bandiera di cui vogliamo memorizzare i colori. Si possono noleggiare tende, ombrelloni e cabine. Le tende e le cabine hanno un numero che le idenfica mentre gli ombrelloni sono identificati dal numero e dalla fila in cui si trovano. Tende, cabine e ombrelloni hanno un prezzo per l'intera stagione oppure per un singolo mese, gli ombrelloni hanno anche un prezzo al giorno. Per le cabine si vuole inoltre conoscere la dimensione espressa in m^2. I clienti si distinguono in abituali o occasionali, ed è memorizzato un documento di identità. Quelli abituali possono prenotare un ombrellone, una tenda o una cabina per un unico lungo periodo specificato con la data di inizio e quella di fine. I clienti occasionali noleggiano gli ombrelloni per un singolo giorno eventualmente anche più volte durante la stagione. Lo stabilimento organizza delle attività ludiche per i clienti abituali: ginnastica, yoga, corsi di nuoto, di vela etc. Per ciascuna attività si vuole memorizzare il nome e il numero massimo di partecipanti. I clienti possono prenotare più attività.

Si dia uno schema grafico a oggetti (secondo la notazione del libro di testo) della base di dati e si trasformi nello schema relazionale mostrandone la rappresentazione grafica (anche questa secondo la notazione del libro di testo, indicando la chiave primaria e le chiavi esterne). Sia per lo schema a oggetti che per lo schema relazionale si devono specificare, rispettivamente, i nomi e i tipi degli attributi di ciascuna classe e relazione.

Soluzione: Lo schema grafico a oggetti è illustrato in Figura 5.

La trasformazione dello schema a oggetti nello schema relazionale è mostrato in Figura 6.

Esercizio 6. Dati i seguenti schemi di relazione (le chiavi primarie sono sottolineate, le chiavi esterne sono date esplicitamente):

- Dipendenti(Id, Nome, AnnoNascita, Stipendio)
- Manager(Id*, Area)
  Id FK(Dipendenti)
- Staff(Id*, Cod*)
  Id FK(Dipendenti) Cod FK(Progetti)
- Progetti(Cod, Nome, Budget, Responsabile*)
  Responsabile FK(Manager)

(i) Scrivere in SQL le seguenti interrogazioni:

1. trovare il nome e il budget dei progetti il cui manager ha meno di 30 anni ed è dell'area "Centro";
2. trovare il nome e lo stipendio dei manager dei progetti che hanno più di 30 dipendenti e un budget superiore a 500000 euro.
3. trovare il codice e il nome dei manager che dirigono persone che sono tutte più anziane di loro o coetanei;

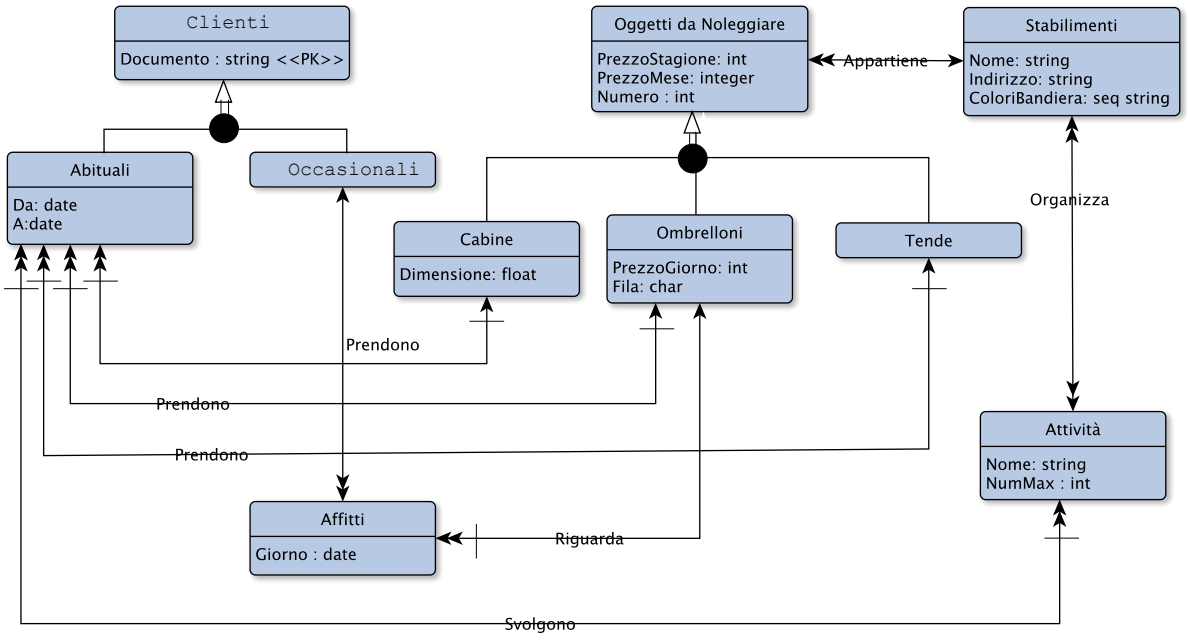

Figura 5: Schema a oggetti

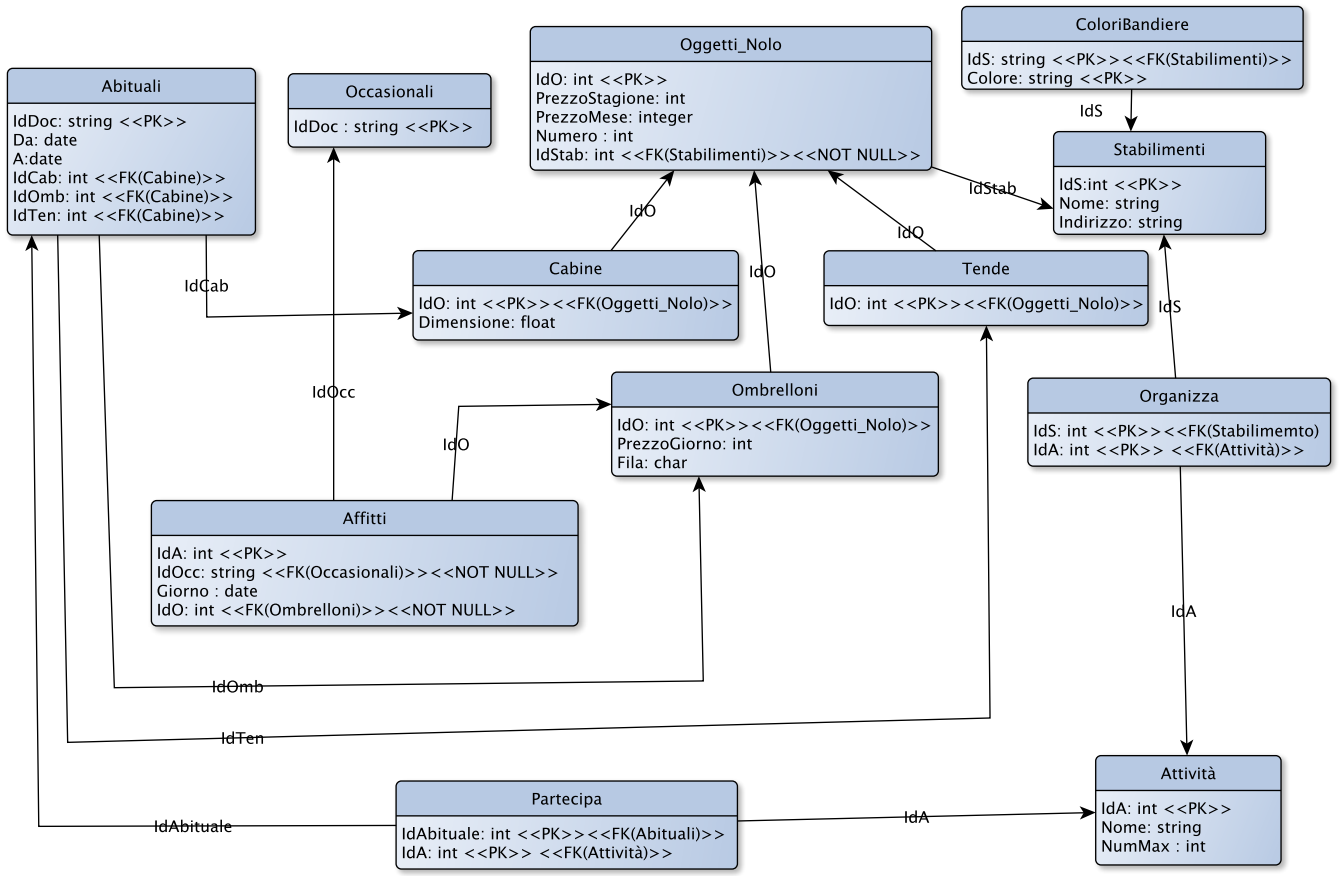

Figura 6: Schema relazionale

4. per ogni area, trovare il nome e il numero dei progetti del manager (o dei manager) che coordina il numero massimo di dipendenti. Se un dipendente occorre in più progetti deve essere contato una sola volta e il manager non coordina se stesso;
5. Aumentare del 20% lo stipendio dei dipendenti che lavorano in almeno due progetti e in almeno uno è il manager.

(*ii*) scrivere in Algebra relazionale la prima interrogazione.

Soluzione:

1. Trovare il nome e il budget dei progetti il cui manager ha meno di 30 anni ed è dell'area "Centro";

```
SELECT p.Nome, p.Budget
FROM   Progetti p JOIN Manager m ON p.Responsabile = m.Id
       JOIN Dipendenti d USING (Id)
WHERE m.Area = 'Centro' AND 2020 - d.AnnoNascita < 30
```

$$\pi_{\text{Nome,Budget}}(\sigma_{2020-\text{AnnoNascita}<30 \ \wedge \ \text{Area='Centro'}}(\text{Progetti} \underset{\text{Responsabile}=Id}{\bowtie} \text{Manager} \bowtie \rho_{\text{Nome}\rightarrow\text{NomeD}}(\text{Dipendeni})$$

2. Trovare il nome e lo stipendio dei manager dei progetti che hanno più di 30 dipendenti e un budget superiore a 500000 euro.

```
SELECT DISTINCT p.Responsabile, d.Nome, d.Stipendio
FROM        Progetti p JOIN Dipendenti d ON p.Responsabile = d.Id
            JOIN Staff s USING (Cod)
WHERE       p.Budget > 500000
GROUP BY p.Cod, p.Responsabile, d.Nome, d.Stipendio
HAVING      COUNT(*) > 30
```

3. Trovare il codice e il nome dei manager che dirigono persone che sono tutte più anziane di loro o coetanei

```
SELECT m.Id, d.Nome
FROM   Manager m NATURAL JOIN Dipendenti d
WHERE NOT EXISTS (SELECT *
                  FROM   Progetti p NATURAL JOIN Staff s
                         JOIN Dipendenti d1 USING (Id)
                  WHERE p.Responsabile = m.Id AND
                         d.AnnoNascita < d1.AnnoNascita)
```

4. Per ogni area, trovare il nome e il numero dei progetti del manager (o dei manager) che coordina il numero massimo di dipendenti. Se un dipendente occorre in più progetti deve essere contato una sola volta e il manager non coordina se stesso.

```
CREATE VIEW ManagerCoordina(Id, Area, NumProg, NumDip)
AS SELECT      m.Id, m.Area, COUNT(DISTINCT p.Cod), COUNT(DISTINCT s.Id)
   FROM        Progetti p JOIN Staff s ON s. Cod = p. Cod AND s.Id <> p.Responsabile
               JOIN Manager m ON p.Responsabile = m.Id
   GROUP BY m.Id, m.Area

SELECT mc.Area, d.Nome
FROM   ManagerCoordina mc NATURAL JOIN Dipendenti d
WHERE mc.NumDip = (SELECT MAX(NumDip)
                   FROM   ManagerCoordina mc1
                   WHERE mc.Area = mc1.Area)
```

5. Aumentare del 20% lo stipendio dei dipendenti che lavorano in almeno due progetti e in almeno uno è il manager.

```
UPDATE Dipendenti,
SET    Stipendio = Stipendio * 1.20
WHERE  Id IN (SELECT s.Id
              FROM   Progetti p JOIN Staff s ON p.Responsabile = s.Id
              WHERE s.Cod <> p.Cod)
```