Basi di dati A.A. 2023/24
Esercitazione N. 4

Luca Andolfi, Maurizio Lenzerini

1. Creazione della base di dati e delle tabelle

Occorre definire una nuova base di dati, di nome Esercitazione-4 e poi, operando sullo schema di default "public", occorre produrre il codice SQL per creare e popolare le seguenti tabelle.

automobile

targa	*cilindrata*	*citta*
A1	500	Roma
A2	1200	Roma
A3	900	Milano
A4	1000	Firenze
A5	2000	Palermo
A6	3000	Torino
A7	2000	Torino
A8	4000	Roma
A9	4000	Napoli
A10	2500	Siena

custodita

codice	*citta*
G1	Roma
G2	Firenze
G3	Firenze
G4	Milano
G5	Milano
G6	Palermo
G7	Roma
G8	Palermo
G9	Roma
G10	Milano
G11	Roma
G12	Siena

garage

targaauto	*codgarage*	*numero*
A1	G1	2
A1	G2	1
A3	G5	1
A2	G6	15
A2	G7	17
A5	G6	18
A5	G1	2
A4	G3	1
A6	G1	1
A5	G8	1
A8	G9	NULL
A8	G2	3
A10	G12	5

Si osservi che:

- La tabella "automobile" ha una tupla per ogni automobile di interesse per la base di dati, dove ogni automobile è identificata dalla targa (che non può avere valori nulli).
- L'attributo città di "automobile" non può avere valori nulli e per ogni automobile indica il nome della città in cui l'automobile è stata immatricolata.

- La tabella “garage” ha una tupla per ogni garage di interesse per la base di dati, dove ogni garage è identificato dal codice targa (che non può avere valori nulli).
- L’attributo città di “garage” indica il nome della città in cui ogni garage si trova ed il valore nullo significa che tale città non è nota.
- La tabella “custodita” indica quali automobili sono state custodite in quali garage quante volte. Più precisamente, ogni tupla ci dice quante volte (con l’attributo numero, che ha valori positivi) una certa automobile (denotata dalla targa) è stata custodita in un certo garage (denotato dal codice). Il valore nullo per “numero” significa che, sebbene il numero di volte in cui l’automobile è stata custodita nel garage sia maggiore di 0, tale numero non è noto.

In questa parte di esercitazione si richiede agli studenti di ragionare sulla base di dati da realizzare e decidere quali istruzioni SQL e in quale ordine sottoporre al sistema al fine di definire esattamente le tabelle mostrate sopra.

2. Aggiornamenti della base di dati

Una volta definita e popolata la base di dati, svolgiamo un semplice esercizio di utilizzo delle operazioni delete e update. Infatti, in questa parte di esercitazione si richiede agli studenti di aggiornare la base di dati definita e popolata al punto 1, eseguendo queste due operazioni:

1. eliminare le automobili immatricolate a Siena;
2. per ogni automobile A e per ogni garage G che si trova a Roma, aggiungere 1 al numero di volte in cui A è stata custodita in G.

3. Scrittura e test delle query

In quest’ultima parte di esercitazione si richiede agli studenti di scrivere e provare le query SQL corrispondenti alle seguenti esigenze informative.

1. Per ogni automobile e per ogni garage in cui l’automobile è stata custodita esattamente una volta, mostrare la targa dell’automobile, il codice del garage e la città in cui il garage si trova. L'output deve essere ordinato in ordine crescente sulla targa dell'auto.
2. Per ogni città e per ogni automobile con cilindrata maggiore di 100 custodita in almeno un garage di quella città per più di 10 volte, mostrare il nome della città e la targa dell’automobile, evitando ripetizioni nel risultato.
3. Chiamiamo attivi i garage che hanno custodito almeno un’automobile. Per ogni garage attivo mostrare il codice del garage ed il numero complessivo di custodie di automobili in quel garage.
4. Per ogni garage attivo mostrare il codice del garage, ed il numero complessivo di custodie di automobili in quel garage, ma solo se il numero complessivo è maggiore di 10 oppure se il suo calcolo fornisce null.
5. Per ogni automobile che è stata custodita almeno una volta in almeno un garage “di casa” (ossia un garage che si trova nella città di immatricolazione dell’automobile),

mostrare la targa ed il numero complessivo di volte in cui l'automobile è stata custodita nei garage di casa ed il numero di tali garage.

6. Mostrare il codice dei garage che non hanno mai custodito automobili immatricolate a Roma.
7. Mostrare il codice dei garage che hanno custodito solo automobili immatricolate nella stessa città in cui si trova il garage.
8. Mostrare tutte le coppie (A,B) dove A è la targa di un'automobile, B è il nome di una città e l'automobile di targa A non è mai stata custodita in un garage che si trova nella città di nome B.
9. Per ogni automobile mostrare il numero di città nei garage delle quali è stata custodita.
10. Per ogni automobile mostrare il codice di ogni garage nel quale è stata custodita il maggior numero di volte (ovviamente maggiore di 0). Ad esempio, se i garage g1, g2 e g3 sono quelli in cui è stata custodita l'automobile t, ed in particolare t è stata custodita 3 volte nel garage g1, 1 volta nel garage g2 e 3 volte nel garage g3, allora nel risultato dovranno comparire le tuple <t,g1> e <t,g3>.

4. Soluzioni

4.1 Codice SQL per la creazione della base di dati e delle tabelle

La base di dati si crea operando con PgAdmin oppure eseguendo questo comando da linea di comando:

```
$psql -U postgres -c "CREATE DATABASE Esercitazione-4;"
```

Il sistema chiederà la password associata all'utente *postgres*. Sul DBMS della macchina virtuale tale password è "biar".

Per creare le tabelle occorre accedere al database appena creato ed eseguire i seguenti comandi. Ricordiamo che tutti gli attributi che formano la chiave primaria di una tabella sono implicitamente definiti come NOT NULL. Ricordiamo anche che la clausola CHECK serve a dichiarare un vincolo di dominio (in particolare se coinvolge un attributo, come ad esempio nel caso di cilindrata qui sotto) oppure un vincolo di tupla. La soluzione che ora presentiamo definisce tutti i vincoli con dichiarazioni ad hoc di tipo CONSTRAINT ed assegna ad ognuno un nome.

```
CREATE TABLE automobile (
  targa VARCHAR(20),
  cilindrata INTEGER,
  citta VARCHAR(20),
  CONSTRAINT automobile_pk PRIMARY KEY (targa),
  CONSTRAINT nonulloCitta check (citta IS NOT NULL),
  CONSTRAINT cilindrataPositiva CHECK (cilindrata > 0)
);

CREATE TABLE garage (
  codice VARCHAR(10),
```

```
  citta VARCHAR(20),
  CONSTRAINT garage_pk PRIMARY KEY (codice)
);

CREATE TABLE custodita (
  targaauto VARCHAR(20),
  codgarage VARCHAR(20),
  numero INTEGER,
  CONSTRAINT custodita_pk PRIMARY KEY (targaauto, codgarage),
  CONSTRAINT targaauto_fk foreign key (targaauto) REFERENCES automobile,
  CONSTRAINT codgarage_fk foreign key (codgarage) REFERENCES garage,
  CONSTRAINT valoreNumero CHECK (numero > 0)
);
```

In PostgreSQL si possono assegnare nomi ai vincoli anche quando sono dichiarati accanto all'attributo coinvolto. Quindi, un modo alternativo per definire lo schema visto prima è:

```
CREATE TABLE automobile (
  targa VARCHAR(20) CONSTRAINT automobile_pk PRIMARY KEY,
  cilindrata INTEGER CONSTRAINT cilindrataPositiva CHECK (cilindrata > 0),
  citta VARCHAR(20) CONSTRAINT nonnulloCitta NOT NULL
);

CREATE TABLE garage (
  codice VARCHAR(10) CONSTRAINT garage_pk PRIMARY KEY,
  citta VARCHAR(20)
);

CREATE TABLE custodita (
  targaauto VARCHAR(20) CONSTRAINT targaauto_fk REFERENCES automobile,
  codgarage VARCHAR(20) CONSTRAINT codgarage_fk REFERENCES garage,
  numero INTEGER CONSTRAINT valoreNumero CHECK (numero > 0),
  CONSTRAINT custodita_pk PRIMARY KEY (targaauto, codgarage)
);
```

4.2 Codice SQL per l'inserimento di dati

Per popolare la base di dati occorre eseguire le opportune operazioni di inserimento, stando attenti ad inserire prima le tuple di "automobile" e "garage" e poi le tuple di "custodita", così da non violare i vincoli di integrità referenziale durante l'inserimento su "custodita".

```
insert into automobile values
('A1',500,'Roma'),
('A2',1200,'Roma'),
('A3',900,'Milano'),
```

```
('A4',1000,'Firenze'),
('A5',2000,'Palermo'),
('A6',3000,'Torino'),
('A7',2000,'Torino'),
('A8',4000,'Roma'),
('A9',4000,'Napoli'),
('A10',2500,'Siena');

insert into garage values
('G1','Roma'),
('G2','Firenze'),
('G3','Firenze'),
('G4','Milano'),
('G5','Milano'),
('G6','Palermo'),
('G7','Roma'),
('G8','Palermo'),
('G9','Roma'),
('G10','Milano'),
('G11','Roma'),
('G12','Siena');

insert into custodita values
('A1','G1',2),
('A1','G2',1),
('A3','G5',1),
('A2','G6',15),
('A2','G7',17),
('A5','G6',18),
('A5','G1',2),
('A4','G3',1),
('A6','G1',1),
('A5','G8',1),
('A8','G9',null),
('A8','G2',3),
('A10','G12',5);
```

4.3 Codice SQL per le query

Query 1
Per ogni automobile e per ogni garage in cui l'automobile è stata custodita esattamente una volta, mostrare la targa dell'automobile, il codice del garage e la città in cui il garage si trova. L'output deve essere ordinato in ordine crescente sulla targa dell'auto.

```
select custodita.targaauto, garage.codice, garage.citta
from custodita join garage on custodita.codgarage = garage.codice
where custodita.numero = 1
order by custodita.targaauto
```

Query 2
Per ogni città e per ogni automobile con cilindrata maggiore di 100 custodita in almeno un garage di quella città per più di 10 volte, mostrare il nome della città e la targa dell'automobile, evitando ripetizioni nel risultato.

```
select distinct garage.citta, automobile.targa
from automobile join custodita on automobile.targa = custodita.targaauto
        join garage on custodita.codgarage = garage.codice
where automobile.cilindrata > 100 and custodita.numero > 10
```

Query 3
Chiamiamo attivi i garage che hanno custodito almeno un'automobile. Per ogni garage attivo mostrare il codice del garage ed il numero complessivo di custodie di automobili per quel garage.

```
select custodita.codgarage, sum(custodita.numero)
from custodita
group by custodita.codgarage
```

Notiamo che possiamo ottenere NULL nel secondo attributo del risultato, in particolare quando per un certo garage abbiamo solo valori nulli nel campo numero.

Query 4
Per ogni garage attivo mostrare il codice del garage ed il numero complessivo di custodie di automobili in quel garage, ma solo se il numero complessivo è maggiore 10 oppure se il suo calcolo fornisce null.

```
select garage.codice, sum(custodita.numero)
from garage join custodita on garage.codice = custodita.codgarage
group by garage.codice
having sum(custodita.numero) > 10 or sum(custodita.numero) is null
```

Query 5
Per ogni automobile che è stata custodita almeno una volta in almeno un garage che si trova nella città di immatricolazione dell'automobile, mostrare la targa, il numero complessivo di volte in cui l'automobile è stata custodita nei garage che si trovano nella città di immatricolazione dell'automobile ed il numero di tali garage.

```
select automobile.targa, sum(custodita.numero), count(garage.codice)
from automobile join custodita on automobile.targa = custodita.targaauto
        join garage on garage.codice = custodita.codgarage
where automobile.citta = garage.citta
group by automobile.targa
```

Si noti che nel calcolo del numero dei garage di casa di un'automobile non serve indicare la clausola "distinct" perché la coppia <automobile,garage> è chiave nella relazione "custodita".

Query 6
Mostrare il codice dei garage che non hanno mai custodito automobili immatricolate a Roma.

```
select garage.codice
from garage
except
select custodita.codgarage
from custodita join automobile on custodita.targaauto = automobile.targa
where automobile.citta = 'Roma'
```

Si noti che abbiamo usato il metodo già discusso in algebra relazionale: togliamo da tutti i garage quelli che hanno custodito almeno un'automobile immatricolata a Roma.

Query 7
Mostrare il codice dei garage che hanno custodito solo automobili immatricolate nella stessa città in cui si trova il garage.

```
select garage.codice
from garage
except
select custodita.codgarage
from custodita join automobile on custodita.targaauto = automobile.targa
        join garage on custodita.codgarage = garage.codice
where garage.citta != automobile.citta
```

Si noti che un garage che non ha custodito alcuna automobile correttamente appare nel risultato. Questo si capisce bene ricordando la semantica di "solo": i garage che hanno custodito solo automobili immatricolate nella stessa città in cui si trova il garage sono tutti i garage tranne quelli che hanno custodito almeno un'automobile immatricolata in una città diversa da quella in cui si trova il garage.

Query 8
Mostrare tutte le coppie (A,B) dove A è la targa di un'automobile, B è il nome di una città e l'automobile di targa A non è mai stata custodita in un garage che si trova nella città di nome B.

```
(select automobile.targa, garage.citta
 from automobile, garage
 union
 select a1.targa, a2.citta
 from automobile a1, automobile a2)
except
```

```
select automobile.targa, garage.citta
from automobile join custodita on codgarage = codice
```

Query 9
Per ogni automobile mostrare il numero di città nei garage delle quali è stata custodita.

```
select targa, count(distinct citta)
from automobile left join custodita on targa = targaauto join garage on codgarage = codice
group by targa
```

```
SELECT automobile.targa, count(distinct garage.citta)
FROM custodita JOIN garage ON (custodita.codgarage = garage.codice)
        RIGHT JOIN automobile ON (custodita.targaauto = automobile.targa)
GROUP BY automobile.targa
```

Query 10
Per ogni automobile mostrare il codice di ogni garage nel quale è stata custodita il maggior numero di volte (ovviamente maggiore di 0). Ad esempio, se i garage g1, g2 e g3 sono quelli in cui è stata custodita l'automobile t, ed in particolare t è stata custodita 3 volte nel garage g1, 1 volta nel garage g2 e 3 volte nel garage g3, allora nel risultato dovranno comparire le tuple <t,g1> e <t,g3>.

```
select custodita.targaauto, custodita.codgarage
from custodita
except
select c1.targaauto, c1.codgarage
from custodita c1 join custodita c2 on c1.targaauto = c2.targaauto
where c1.numero < c2. numero or c1.numero is null
```