Basi di Dati – Appello del 15/06/2023

Cognome e nome: _____ Matricola: _____ Turno: ___

Riportare sui fogli i seguenti dati: cognome, nome, matricola e turno di laboratorio.

Esame di SQL

Punteggi massimi:

- *Domande 1 e 2 svolte perfettamente: 23;*
- *Domande 1 e 3 svolte perfettamente: 25;*
- *Domande 2 e 3 svolte perfettamente: 28;*
- *Domande 1, 2 e 3 svolte perfettamente: 33.*

Lo svolgimento corretto di una sola domanda non permette il raggiungimento della sufficienza.

Le seguenti relazioni definiscono una base di dati "**DoveStudio**" per gestire una piattaforma di prenotazione aule studio per gli atenei della città. Gli attributi sottolineati sono le chiavi primarie delle relazioni.

STUDENTE(Matricola, Ateneo, Cognome, Nome, DataNascita, Gruppo*)
AULA(Codice, Nome, Indirizzo, NumPosti)
PRENOTAZIONE(Matricola, Ateneo, Data, Aula, FasciaOraria, Gruppo*)
GRUPPO(NomeGruppo, DataCreazione)

Valgono i seguenti vincoli UNIQUE:

- UNIQUE(Data, Gruppo) in PRENOTAZIONE
- UNIQUE(Matricola, Ateneo, Gruppo) in STUDENTE

Vincoli di integrità referenziale:
STUDENTE(Gruppo) referenzia GRUPPO(NomeGruppo)
PRENOTAZIONE(Matricola, Ateneo) referenziano STUDENTE(Matricola, Ateneo),
PRENOTAZIONE(Aula) referenzia AULA(Codice),
PRENOTAZIONE(Matricola, Ateneo, Gruppo) referenzia STUDENTE(Matricola, Ateneo, Gruppo).

FasciaOraria può assumere i valori "mattino", "pomeriggio", "intera giornata".
Ogni studentessa o studente può prenotare un'aula a proprio nome o a nome di un gruppo di studio. In questo ultimo caso, l'attributo Gruppo conterrà il riferimento al suo gruppo di studio e verranno riservati un numero di posti nell'aula pari al numero di componenti del gruppo. In PRENOTAZIONE ci sarà, però, solo la tupla relativa alla persona che ha effettuato la prenotazione.

Le date rappresentate da stringhe nel formato 'AAAAMMGG'.

I rimanenti attributi sono autoesplicativi.

Con riferimento alla base di dati "DoveStudio" esprimere in SQL le seguenti interrogazioni.

Domanda 1 (bassa complessità).
Mostrare cognome, ateneo e data di nascita di studentesse e studenti che hanno prenotato un'aula studio solo per sé stess* per il 15/06/2023 e che si troveranno nell'aula almeno per la mattinata. Indicare anche il nome dell'aula prenotata. Elencare studentesse e studenti in ordine crescente di età.

Soluzione 1.

```
SELECT s.Cognome, s.Ateneo, s.DataNascita, a.Nome
FROM studente s JOIN prenotazione p
     ON s.Matricola=p.Matricola AND s.Ateneo=p.Ateneo JOIN aula a
```

```
     ON a.Codice=p.Aula
WHERE p.Data='20230615' AND p.Gruppo IS NULL AND
     (FasciaOraria='mattino' OR FasciaOraria='intera giornata')
ORDER BY DataNascita DESC;
```

Domanda 2 (media complessità).
Elencare i gruppi di studio creati nel 2020 che hanno prenotato almeno tre aule distinte nel 2021. Indicare anche il numero di prenotazioni di ciascun gruppo. **Non usare sottoquery.**

Soluzione 2.

```
SELECT g.NomeGruppo, COUNT(DISTINCT Data) AS NumPrenotazioni
FROM gruppo g JOIN prenotazione p ON (g.NomeGruppo=p.Gruppo)
     JOIN studente s ON (s.Gruppo=g.NomeGruppo)
WHERE g.DataCreazione LIKE '2020%' AND p.Data LIKE '2021%'
GROUP BY g.NomeGruppo
HAVING COUNT(DISTINCT p.Aula)>=3;
```

Domanda 3 (alta complessità).
Mostrare tutte le informazioni sulle aule in cui, in mattinata, i posti non sono mai stati esauriti. **È possibile utilizzare gli operatori insiemistici**.

Soluzione 3.

```
WITH PrenotazioniStudMattino AS (
     SELECT Matricola, Ateneo, Aula, Data
     FROM prenotazione p
     WHERE p.Gruppo IS NULL AND (p.FasciaOraria='mattina' OR p.FasciaOraria='intera
giornata')
          UNION
     SELECT s.Matricola, s.Ateneo, p.Aula, p.Data
     FROM studente s JOIN prenotazione p ON (s.Gruppo=p.Gruppo)
     WHERE p.FasciaOraria='mattina' OR p.FasciaOraria='intera giornata'
),
NumPrenotazioniAula AS (
     SELECT Aula, Data, COUNT(*) AS NumStud
     FROM PrenotazioniStudMattino
     GROUP BY Aula, Data
)
SELECT *
FROM Aula
WHERE Codice NOT IN (
     SELECT a.Codice
     FROM Aula a JOIN NumPrenotazioniAula npa ON (a.Codice=npa.Aula)
     WHERE npa.NumStud >= a.NumPosti
);
```

Esame di Teoria (rispondere su fogli separati rispetto alla parte di SQL)

Domanda 1 (9 punti).
Con riferimento alla base di dati "**DoveStudio**":

A. (4 punti) Esprimere in Algebra Relazionale l'interrogazione
Elencare i gruppi che hanno studiato in tutte le aule con più di 100 posti.

B. (5 punti) Esprimere, nel calcolo relazionale su tuple con dichiarazione di range, la seguente domanda:
Mostrare cognome e nome delle studentesse/degli studenti che hanno sempre prenotato la stessa aula. Indicare anche il nome dell'aula.

Soluzione 1.

A. Una possibile soluzione in algebra relazionale è la seguente:

$$\pi_{Gruppo,Aula}(prenotazione) \div \rho_{Aula \leftarrow Codice}(\pi_{Codice}(\sigma_{NumPosti>100}(aula)))$$

B. Una possibile soluzione è la seguente:

{s.Cognome, s.Nome, a.Nome | s(STUDENTE),a(AULA) | ∃p(prenotazione)(p.Matricola=s.Matricola ∧ p.Ateneo=s.Ateneo ∧ a.Codice=p.Aula ∧ ¬∃p'(prenotazione)(p.Matricola=p'.Matricola ∧ p.Ateneo=p'.Ateneo ∧ p.Aula≠p'.Aula))}

oppure

{s.Cognome, s.Nome, a.Nome | s(STUDENTE),a(AULA) | ∀p(prenotazione)((s.Matricola=p.Matricola ∧ s.Ateneo=p.Ateneo) ⇒ p.Aula=a.Aula)}

Domanda 2 (8 punti).

A. Enunciare la condizione necessaria e sufficiente affinché una decomposizione sia senza perdita di informazioni.

B. Dati: R(A, B, C, D, E, F, G) e F = { D → B G, C → A B F, B C → D E, G → C D }
dire se R è in 3FN e se non lo è decomporla in relazioni in 3FN esplicitando tutti i passaggi. Il risultato è BCNF?

Soluzione 2.

A. Si vedano gli appunti/testo/slide.

B. Per prima cosa è necessario identificare le chiavi candidate. In questo caso abbiamo tre chiavi K1={C}, K2={D}, K3={G} (il calcolo della chiusura lo dimostra).

La relazione R è già in 3FN e in BCNF, in quanto tutte le dipendenze funzionali sono del tipo "superchiave". Non è necessario decomporre.

Domanda 3 (7 punti).
Con riferimento alla base di dati "**DoveStudio**" e tenendo conto dei seguenti dati quantitativi:

CARD(aula) = 20
CARD(prenotazione) = 30 000 000 (trenta milioni)
VAL(Gruppo, prenotazione) = 1000
VAL(Aula, prenotazione) = 20
MIN(NumPosti,aula) = 30
MAX(NumPosti,aula) = 230

disegnare gli alberi sintattici prima e dopo l'ottimizzazione logica e calcolare il numero di tuple “mosse” prima e dopo l’ottimizzazione logica della seguente query:

$$\sigma_{\text{Gruppo='Linuxare incallite'} \wedge \text{FasciaOraria='pomeriggio'} \wedge \text{NumPosti}<130} (\text{aula} \bowtie_{\text{AULA.Codice=PRENOTAZIONE.Aula}} \text{prenotazione})$$

Soluzione 3.

La query ottimizzata dividendo la selezione e portandola verso le foglie è:

$(\sigma_{\text{NumPosti}<130} (\text{aula}))$
$\bowtie_{\text{AULA.Codice=PRENOTAZIONE.Aula}} (\sigma_{\text{Gruppo=’Linuxare incallite’} \wedge \text{FasciaOraria='pomeriggio'}} (\text{prenotazione}))$

Prima dell’ottimizzazione:
• Costo r1 = (aula $\bowtie_{\text{AULA.Codice=PRENOTAZIONE.Aula}}$ prenotazione): $20 \cdot 3 \cdot 10^7 = 6 \cdot 10^8$.
• Cardinalità $|r1|$= CARD(prenotazione) = $3 \cdot 10^7$ (equijoin attraverso la chiave esterna)
• Costo della selezione = $|r1|$
• Costo totale = $6 \cdot 10^8 + 3 \cdot 10^7 \sim 6 \cdot 10^8$.

Dopo l’ottimizzazione:
• Costo $\sigma_1 = \sigma_{\text{NumPosti}<130} (\text{aula})$ = CARD(aula) = 20.
• Costo $\sigma_2 = \sigma_{\text{Gruppo=’Linuxare incallite’} \wedge \text{FasciaOraria='pomeriggio'}} (\text{prenotazione}) = 3 \cdot 10^7$
• Tuple prodotte dalla selezione $|\sigma_1|$ =
(130 - MIN(NumPosti, aula))/(MAX(NumPosti, aula) - MIN(NumPosti, aula)) · CARD(aula) = 100/200 · 20 = 10.
• Tuple prodotte dalla selezione $|\sigma_2|$ = 1/VAL(Gruppo, prenotazione) · 1/VAL(FasciaOraria, prenotazione) · CARD(prenotazione) = $1/1000 \cdot 1/3 \cdot 3 \cdot 10^7 = 10^4$
• Costo join r = $\sigma_1 \bowtie_{\text{AULA.Codice=PRENOTAZIONE.Aula}} \sigma_2 = 10 \cdot 10^4 = 10^5$.
• Costo totale = $20 + 3 \cdot 10^7 + 10^5 \sim 3 \cdot 10^7$

Domanda 4 (9 punti).
Date le seguenti transazioni:
T1: r(x), r(y), w(y), w(z)
T2: r(x), r(y), w(z)

Per ognuna delle seguenti storie (schedule) di T1 e T2 dire se è compatibile con il 2PL (lock a due fasi) o no **spiegando perché**.

- **a.** r2(x), r2(y), w2(z), r1(x), r1(y), w1(y), w1(z)
- **b.** r1(x), r2(x), r1(y), w1(y), r2(y), w2(z), w1(z)
- **c.** r1(x), r2(x), r1(y), w1(y), r2(y), w1(z), w2(z)

Soluzione 4.

Risposte:

- **a.** È compatibile perché è una storia seriale.
- **b.** Non è compatibile perché T1 dovrebbe rilasciare il lock sull’oggetto y per permettere a T2 di leggerlo e poi dovrebbe acquisire un nuovo lock per scrivere l’oggetto z.
- **c.** È compatibile e un esempio di storia con i relativi lock è:
  LS1(x), r1(x), LS2(x), r2(x), LX1(y), LX1(z), r1(y), w1(y), UN1(x,y), LS2(y), r2(y), w1(z), UN1(z), LX2(z), w2(z), UN2(x,y,z).