Seconda prova di verifica, del 18/12/2008

1. Si consideri il seguente schema relazionale:

   Impiegati (IdImp, Nome, AnnoNascita, Sesso, Stipendio)
   ImpProg (IdImp *, IdProg*)
   Progetti(IdProg, Nome, DataInizio, DataFine, Budget)

   Si scrivano le interrogazioni SQL che restituiscono le seguenti informazioni.

   (a) Si ritorni il nome di quei progetti in cui lavora qualche impiegato maschio nato dopo il 1984

   (b) Si ritorni il nome di quei progetti in cui lavorano solo impiegati maschi e nati dopo il 1984

   (c) Si ritorni il nome di quei progetti in cui tutti gli impiegati maschi che vi lavorano sono nati dopo il 1984

2. Si consideri lo schema relazionale R(A,B,C,D,E) con le seguenti DF, già in forma canonica:

   $AE \rightarrow B$, $C \rightarrow D$, $C \rightarrow E$, $DA \rightarrow B$, $B \rightarrow C$

   (a) La decomposizione R1(ABC) e R2(BDE) preserva i dati?

   (b) Si trovino le chiavi di R.

   (c) Dire se lo schema è in 3FN o in FNBC.

   (d) Si applichi allo schema l'algoritmo di sintesi, e si dica se la decomposizione risultante è in FNBC.

   (e) (Opzionale) Si applichi allo schema l'algoritmo di analisi e si dica se la decomposizione risultante preserva le dipendenze.

   (f) Si porti il seguente schema relazionale in forma canonica: $R<(A, B, C, D, E), \{A \rightarrow BC, BDE \rightarrow C, BE \rightarrow A, C \rightarrow A\}>$.

3. Si consideri lo schema relazionale:

   R(Rpk: integer, RA: integer, RB: integer)
   S(Spk: integer, SA: string, SB: integer, SRfk*: integer)

   Si supponga che la relazione S sia **ordinata** sulla chiave primaria Spk. Si consideri l'interrogazione

```
SELECT      DISTINCT s.SA, AVG(r.RA) AS M
FROM        R r , S s
WHERE       r.Rpk = s.SRfk  AND r.RA > 10 AND s.SA > 'b'
GROUP BY    s.Spk, s.SA
HAVING      COUNT(*) > 3
ORDER BY    s.SA ;
```

(a) Disegnare l'albero di sintassi astratta di un'espressione algebrica (albero logico) per l'interrogazione.

(b) Disegnare un piano d'accesso efficiente per la stessa interrogazione che non faccia uso di indici.

(c) Disegnare un piano d'accesso efficiente supponendo che siano presenti tutti gli indici che si desidera.

Seconda prova di verifica del 18/12/2008 - Soluzioni

1. Si consideri il seguente schema relazionale:

   Impiegati (IdImp, Nome, AnnoNascita, Sesso, Stipendio)
   ImpProg (IdImp *, IdProg*)
   Progetti(IdProg, Nome, DataInizio, DataFine, Budget)

   Si scrivano le interrogazioni SQL che restituiscono le seguenti informazioni.

   (a) Si ritorni il nome di quei progetti in cui lavora qualche impiegato maschio nato dopo il 1984

   ```
   SELECT p.Nome
   FROM   Progetti p, ImpProg ip, Impiegati i
   WHERE  p.IdProg = ip.IdProg AND ip.IdImp = i.IdImp
          AND i.Sesso = 'M' AND i.AnnoNascita > 1984
   ```

   (b) Si ritorni il nome di quei progetti in cui lavorano solo impiegati maschi e nati dopo il 1984

   ```
   SELECT   p.Nome
   FROM     Progetti p
   WHERE    FOR ALL ip IN ImpProg, i IN Impiegati
            WHERE p.IdProg = ip.IdProg AND ip.IdImp = i.IdImp :
                  i.Sesso = 'M' AND i.AnnoNascita > 1984;
   ```

   ```
   SELECT p.Nome
   FROM   Progetti p
   WHERE  NOT EXISTS ( SELECT *
                       FROM   ImpProg ip, Impiegati i
                       WHERE  p.IdProg = ip.IdProg AND ip.IdImp = i.IdImp
                              AND NOT (i.Sesso = 'M' AND i.AnnoNascita > 1984));
   ```

   (c) Si ritorni il nome di quei progetti in cui tutti gli impiegati maschi che vi lavorano sono nati dopo il 1984

   ```
   SELECT   p.Nome
   FROM     Progetti p
   WHERE    FOR ALL ip IN ImpProg, i IN Impiegati
            WHERE p.IdProg = ip.IdProg AND ip.IdImp = i.IdImp AND i.Sesso = 'M' :
                  i.AnnoNascita > 1984;
   ```

   ```
   SELECT p.Nome
   FROM   Progetti p
   WHERE  NOT EXISTS ( SELECT *
                       FROM   ImpProg ip, Impiegati i
                       WHERE  p.IdProg = ip.IdProg AND ip.IdImp = i.IdImp
                              AND i.Sesso = 'M' AND NOT (i.AnnoNascita > 1984));
   ```

2. Si consideri lo schema relazionale R(A,B,C,D,E) con le seguenti DF, già in forma canonica:

   AE → B, C → D, C → E, DA → B, B → C

   (a) La decomposizione R1(ABC) e R2(BDE) preserva i dati?
   Sì, dato che $B^+ = BCDE \supseteq BDE$

   (b) Si trovino le chiavi di R.
   Tutte le chiavi contengono A, che non è a destra di nessuna dipendenza. Le chiavi sono AB, AC, AD, AE.

   (c) Dire se lo schema è in 3FN o in FNBC.
   Lo schema è in 3FN, dato che tutti gli attributi sono primi. Non è in FNBC, dato che esistono dipendenze che a sinistra non hanno una superchiave.

   (d) Si applichi allo schema l'algoritmo di sintesi, e si dica se la decomposizione risultante è in FNBC.
   Raggruppo le dipendenze: AE → B, C → DE, DA → B, B → C
   Creo le relazioni:
   Nessuna relazione è inclusa in un'altra. Almeno una relazione è superchiave (lo sono R1 ed R3), quindi ho finito. R1(AEB), R2(CDE), R3(DAB), R4(BC).
   Controllo FNBC: per prima cosa devo proiettare le dipendenze sulle relazioni, ottenendo i seguenti schemi. R1(AEB,{AE → B, B → E }), R2(CDE,{C → DE }), R3(DAB,{DA → B, B → D }), R4(BC,{B → C }). *Le dipendenze* B → E in R1 *e* B → D in R3 *violano entrambe la FNBC*

   (e) (Opzionale) Si applichi allo schema l'algoritmo di analisi e si dica se la decomposizione risultante preserva le dipendenze.
   AE → B *ha a sinistra una chiave.* C → DE *non ha una chiave a sinistra per cui genero due relazioni* R1(CDE), R2(CAB).
   Due dipendenze non sono incluse né in R1 né in R2, per cui devo calcolare la proiezione, ottenendo il seguente schema. R1(CDE,{C → DE }), R2(CAB,{B → C, AC → B}).
   B → C *non ha una chiave a sinistra, per cui divido ancora R2 e proietto le dipendenze, ottenendo* R21(BC,{B → C}), R22(BA,{}).
   Lo schema risultante è quindi: R1(CDE,{C → DE }), R21(BC,{B → C}), R22(BA,{}).
   Le dipendenze AE → B, DA → B *sono andate entrambe perdute, dato che* AE^+=AE *e* DA^+=DA.

   (f) Si porti il seguente schema relazionale in forma canonica: R<(A, B, C, D, E),{A → BC, BDE → C, BE → A, C → A}>.
   Elimino gli attributi estranei: {A → B, A → C, BE → C, BE → A, C → A}.
   Elimino le dipendenze ridondanti: {A → B, A → C, BE → A, C → A}.

3. Si consideri lo schema relazionale:

   R(Rpk: integer, RA: integer, RB: integer)
   S(Spk: integer, SA: string, SB: integer, SRfk*: integer)

   Si supponga che la relazione S sia **ordinata** sulla chiave primaria Spk. Si consideri l'interrogazione

```
SELECT      DISTINCT s.SA, AVG(r.RA) AS M
FROM        R r , S s
WHERE       r.Rpk = s.SRfk  AND r.RA > 10 AND s.SA > 'b'
GROUP BY    s.Spk, s.SA
HAVING      COUNT(*) > 3
ORDER BY    s.SA ;
```

   (a) Disegnare l'albero di sintassi astratta di un'espressione algebrica (albero logico) per l'interrogazione.

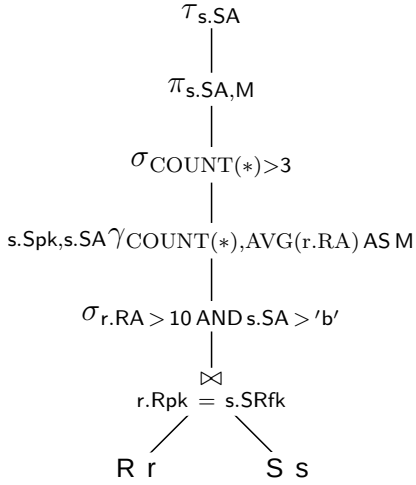

Figura 1: Albero logico

   (b) Disegnare un piano d'accesso efficiente per la stessa interrogazione che non faccia uso di indici.

   (c) Disegnare un piano d'accesso efficiente supponendo che siano presenti tutti gli indici che si desidera.

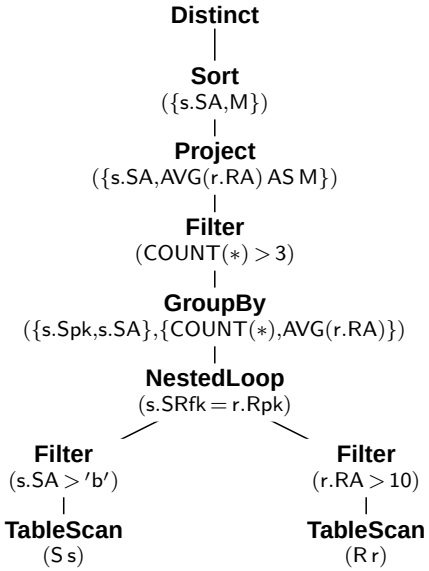

Figura 2: Piano d'accesso con anticipazione delle restrizioni

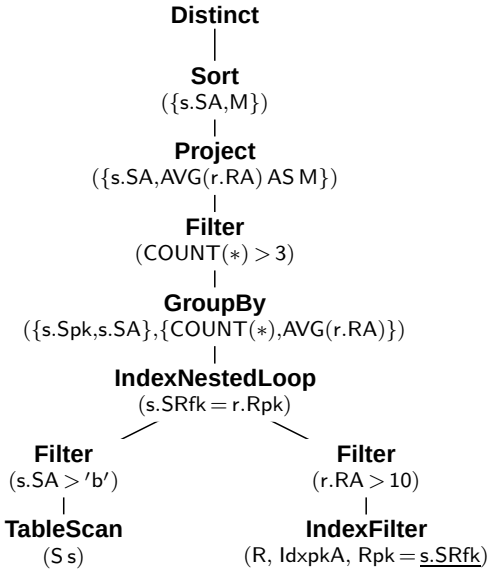

Figura 3: Piano d'accesso con un indice su Rpk