Sistemi Informativi T
14 gennaio 2026
Risoluzione

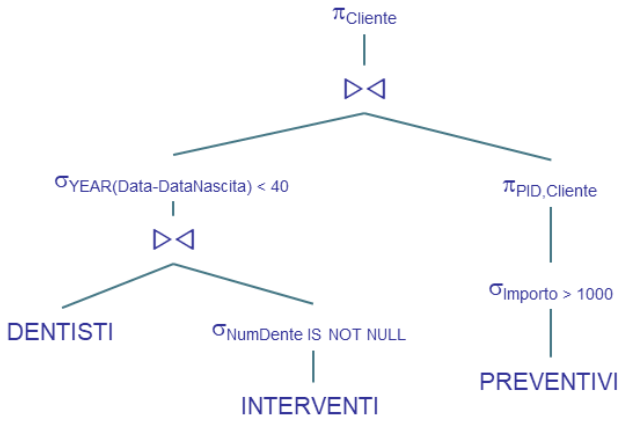
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

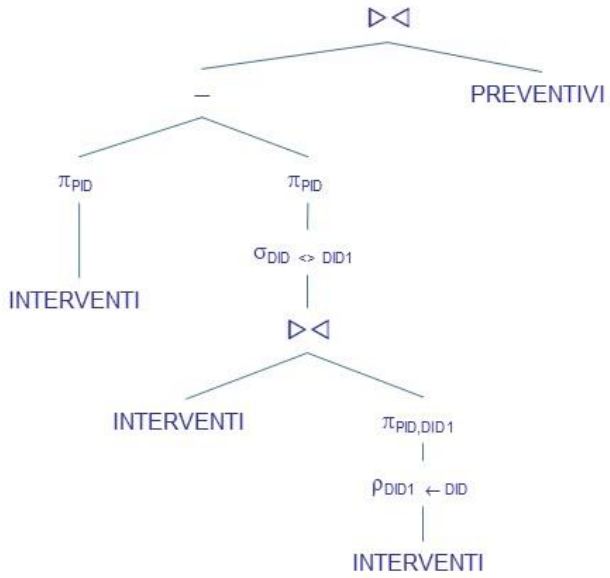
```
DENTISTI(DID,Nome,DataNascita);
PREVENTIVI(PID,Cliente,Data,Importo);
INTERVENTI(CodI,PID,DID,Data,NumDente*),
      DID REFERENCES DENTISTI,
      PID REFERENCES PREVENTIVI;
-- Importo è di tipo DEC(8,2)
-- NumDente è un intero (valori compresi tra 1 e 32) che identifica
-- uno specifico dente. Se è NULL allora l'intervento è di tipo
-- generico (ad es. ablazione tartaro)
-- Ogni intervento fa riferimento a uno specifico preventivo, il quale
-- può richiedere anche più interventi
```

si esprimano in algebra relazionale le seguenti interrogazioni:

1.1) **[1 p.]** I clienti con almeno un preventivo di importo maggiore di 1000€ e in cui almeno un intervento relativo a un dente specifico è stato eseguito da un dentista con meno di 40 anni (al momento dell'intervento)



1.2) **[2 p.]** I dati dei preventivi in cui tutti gli interventi (1 o più) sono stati eseguiti da uno stesso dentista



L'operando destro della differenza trova i codici dei preventivi con interventi di più di un dentista e l'operando sinistro i codici dei preventivi con almeno un intervento.

Sistemi Informativi T
14 gennaio 2026
Risoluzione

2) SQL (5 punti totali)

Con riferimento al DB dell'esercizio 1, si esprimano in SQL le seguenti interrogazioni:

2.1) [2 p.] I nomi dei dentisti che hanno eseguito tutti gli interventi (almeno 3) di un preventivo e in meno di 30 giorni

```
SELECT   DISTINCT D.Nome
FROM     DENTISTI D JOIN INTERVENTI I ON (D.DID = I.DID)
WHERE    I.PID IN
         ( SELECT    I1.PID
           FROM      INTERVENTI I1
           GROUP BY  I1.PID
           HAVING    COUNT(*) >= 3                  -- almeno 3 interventi
           AND       COUNT(DISTINCT I1.DID) = 1     -- un solo dentista
           AND       DAYS(MAX(I1.DATA)) - DAYS(MIN(I1.DATA)) < 30  ) ;

-- La subquery restituisce i preventivi che rispettano le specifiche e
-- il blocco esterno restituisce i dentisti intervenuti in almeno uno
-- di tali preventivi
```

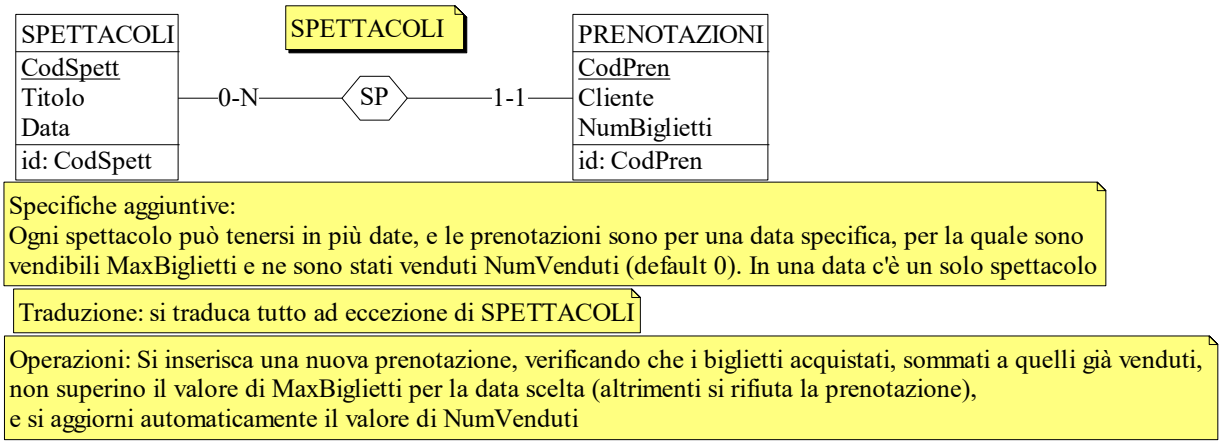
2.2) [3 p.] Considerando solo interventi relativi a specifici denti, il codice del preventivo in cui è passato il minor numero di giorni tra la data del preventivo e quella del primo intervento

```
WITH PRIMO_INTERVENTO(PID,PRIMADATA) AS (
   SELECT I.PID, MIN(I.DATA)
   FROM   INTERVENTI I
   WHERE  NUMDENTE IS NOT NULL
   GROUP BY I.PID )
SELECT   P.PID
FROM     PREVENTIVI P, PRIMO_INTERVENTO PI
WHERE    P.PID = PI.PID
AND      DAYS(PI.PRIMADATA) - DAYS(P.DATA) =
         ( SELECT MIN(DAYS(PI1.PRIMADATA) - DAYS(P1.DATA))
           FROM   PREVENTIVI P1, PRIMO_INTERVENTO PI1
           WHERE  P1.PID = PI1.PID ) ;

-- La c.t.e. restituisce per ogni preventivo la data del primo intervento
-- su uno specifico dente
```

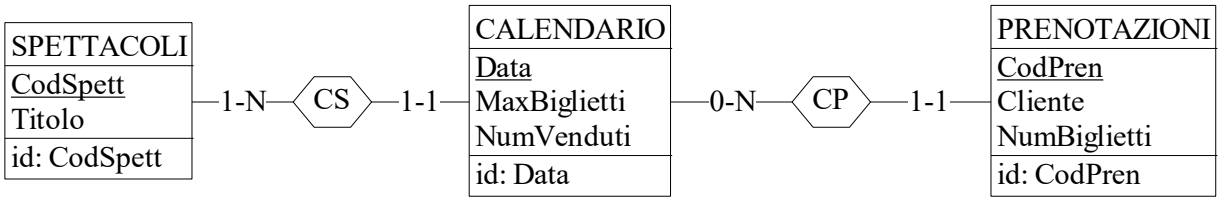
Sistemi Informativi T
14 gennaio 2026
Risoluzione

3) Modifica di schema E/R e del DB (6 punti totali)

Dato il file ESE3.lun fornito, in cui è presente lo schema ESE3-input in figura:



3.1) [2 p.] Si copi lo schema ESE3-input in uno schema ESE3-modificato e si modifichi quest'ultimo secondo le Specifiche aggiuntive;



3.2) [1 p.] Si copi lo schema modificato in uno schema ESE3-tradotto. Mediante il comando Transform/Quick SQL, si traduca la parte di schema specificata, modificando lo script SQL in modo da essere compatibile con DB2 e permettere l'esecuzione del punto successivo, ed eventualmente aggiungendo quanto richiesto dalle Specifiche aggiuntive;
Si veda il relativo file .sql

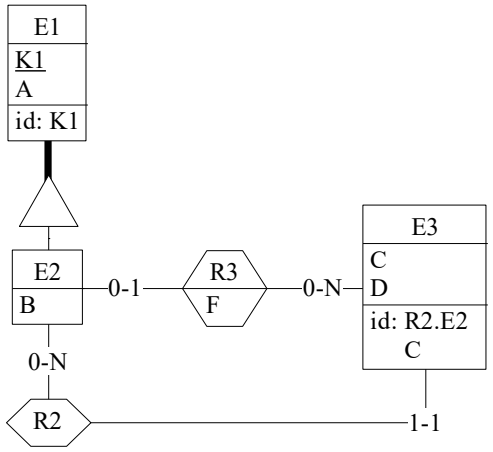
3.3) [3 p.] Si scriva l'istruzione SQL che modifica il DB come da specifiche (usare valori a scelta) e si definiscano i trigger necessari.

```
CREATE OR REPLACE TRIGGER NUOVA_PRENOTAZIONE
AFTER INSERT ON PRENOTAZIONI
REFERENCING NEW AS N
FOR EACH ROW
IF (N.NumBiglietti >
      (SELECT MaxBiglietti - NumVenduti
       FROM    CALENDARIO
       WHERE  Data = N.Data ) )
THEN SIGNAL SQLSTATE '70001' ('Posti insufficienti!') ;
ELSE UPDATE CALENDARIO
        SET       NumVenduti = NumVenduti + N.NumBiglietti
        WHERE  Data = N.Data ;
END IF

-- Possibili anche due trigger distinti

INSERT INTO PRENOTAZIONI VALUES
(:codPrenotazione,:cliente,:numbiglietti,:data) ;
```

Sistemi Informativi T
14 gennaio 2026
Risoluzione

4) Progettazione logica (6 punti totali)

Dato lo schema concettuale in figura e considerando che:



a) le entità E1 ed E2 vengono tradotte assieme;
b) nessuna associazione viene tradotta separatamente;
c) un'istanza di E2 non può essere associata tramite R3 a un'istanza di E3 con D > B;

4.1) [3 p.] Si progettino gli opportuni schemi relazionali e si definiscano tali schemi mediante uno script SQL compatibile con DB2

```
-- il tipo degli attributi non è necessariamente INT

CREATE TABLE E1 (
K1          INT NOT NULL PRIMARY KEY,
A           INT NOT NULL,
TIPO12      SMALLINT NOT NULL CHECK (TIPO12 IN (1,2)),      -- 2: appartiene anche a E2
B           INT,
K1E3        INT,
C           INT,
F           INT,
CHECK ((TIPO12 = 1 AND B IS NULL AND K1E3 IS NULL AND C IS NULL AND F IS NULL)
      OR  (TIPO12 = 2 AND B IS NOT NULL),
CHECK ((K1E3 IS NULL AND C IS NULL AND F IS NULL)
      OR   (K1E3 IS NOT NULL AND C IS NOT NULL AND F IS NOT NULL))    );
-- se TIPO12 = 1 e il primo CHECK è soddisfatto lo è anche il secondo, il quale discrimina i 2 comportamenti
-- possibili quando TIPO12 = 2 (ovvero, partecipare o meno a R3)

CREATE TABLE E3 (
K1          INT NOT NULL REFERENCES E1,
C           INT NOT NULL,
D           INT NOT NULL,
PRIMARY KEY (K1,C)                  );

ALTER TABLE E1
ADD CONSTRAINT FK_E3 FOREIGN KEY (K1E3,C) REFERENCES E3;
```

4.2) [3 p.] Per i vincoli non esprimibili a livello di schema si predispongano opportuni trigger che evitino **inserimenti di singole tuple non corrette**

```
-- Trigger che garantisce che E3.K1 referenzi un'istanza di E2
CREATE OR REPLACE TRIGGER R2
BEFORE INSERT ON E3
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( EXISTS ( SELECT *
                FROM    E1
                WHERE N.K1 = E1.K1 AND E1.TIPO12 = 1   ) )
SIGNAL SQLSTATE '70001' ('La tupla referenzia una tupla che non appartiene a E2!');

-- Il vincolo al punto c) può essere violato solo inserendo in E1 un'istanza che appartiene a E2 e partecipa a R3
CREATE TRIGGER PUNTO_C
BEFORE INSERT ON E1
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( N.B < ( SELECT E3.D
               FROM    E3
               WHERE (N.K1E3,N.C) = (E3.K1,E3.C)  )          )
SIGNAL SQLSTATE '70002' ('La tupla inserita in E3 non rispetta il vincolo del punto c)! ');
```