

**Tempo a disposizione: 2:30 ore**

---

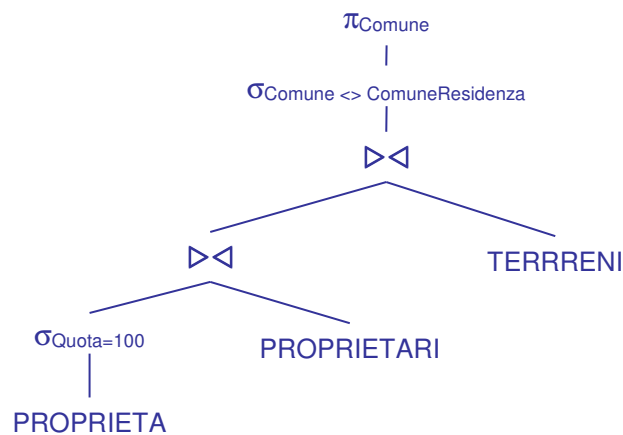
**1) Algebra relazionale (3 punti totali):**

Date le seguenti relazioni:

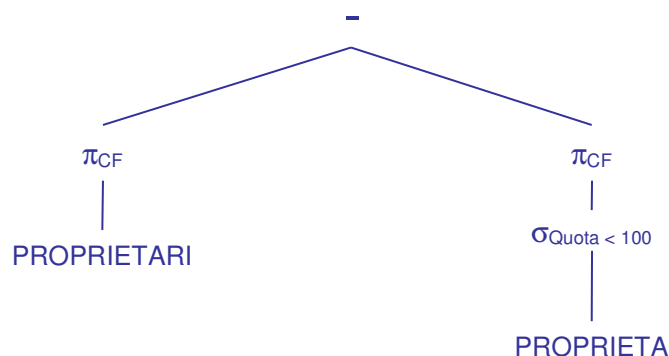
```
TERRENI (IDT, MetriQuadri, Comune);  
PROPRIETARI (CF, ComuneResidenza);  
PROPRIETA (CF, IDT, Quota),  
    CF references PROPRIETARI,  
    IDT references TERRENI;  
-- MetriQuadri è di tipo INT  
-- Quota è di tipo DEC(5,2) e rappresenta la percentuale di possesso  
-- di un terreno (0 < Quota ≤ 100); Quota = 100 se il terreno ha un  
-- solo proprietario  
-- Il valore di Quota*MetriQuadri/100 rappresenta i metri quadri  
-- effettivamente posseduti da un proprietario)
```

si scrivano in algebra relazionale le seguenti interrogazioni:

**1.1) [1 p.]** I comuni in cui c'è un terreno con un solo proprietario, il quale risiede in un altro comune



**1.2) [2 p.]** I codici fiscali (CF) dei proprietari che hanno solo terreni senza altri proprietari



Per definizione, ogni proprietario ha almeno un terreno, quindi sottraendo i proprietari di almeno un terreno con più proprietari ciò che resta sono i proprietari cercati

**Sistemi Informativi T**  
**6 luglio 2017**  
**Risoluzione**

**2) SQL (5 punti totali)**

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

- 2.1) [2 p.]** Per ogni comune, la dimensione media dei propri terreni, escludendo quelli di dimensione minore di 500 metri quadri e quelli con singolo proprietario

```
SELECT    T.Comune, AVG(CAST(T.MetriQuadri AS DEC(10,2))) AS MEDIA
FROM      TERRENI T
WHERE     T.MetriQuadri >= 500
AND       T.IDT IN ( SELECT P.IDT
                     FROM   PROPRIETA P
                     WHERE  P.Quota < 100 )

GROUP BY T.Comune
```

-- Si noti che la seguente soluzione non sarebbe corretta, in quanto ogni  
-- terreno verrebbe considerato tante volte nel calcolo della media quanti  
-- sono i suoi proprietari

```
SELECT    T.Comune, AVG(CAST(T.MetriQuadri AS DEC(10,2))) AS MEDIA
FROM      TERRENI T, PROPRIETA P
WHERE     T.IDT = P.IDT
AND       T.MetriQuadri >= 500
AND       P.Quota < 100
GROUP BY T.Comune
```

- 2.2) [3 p.]** Per ogni proprietario, il comune in cui possiede più terreni diversi (indipendentemente dalla quota e dai metri quadri posseduti)

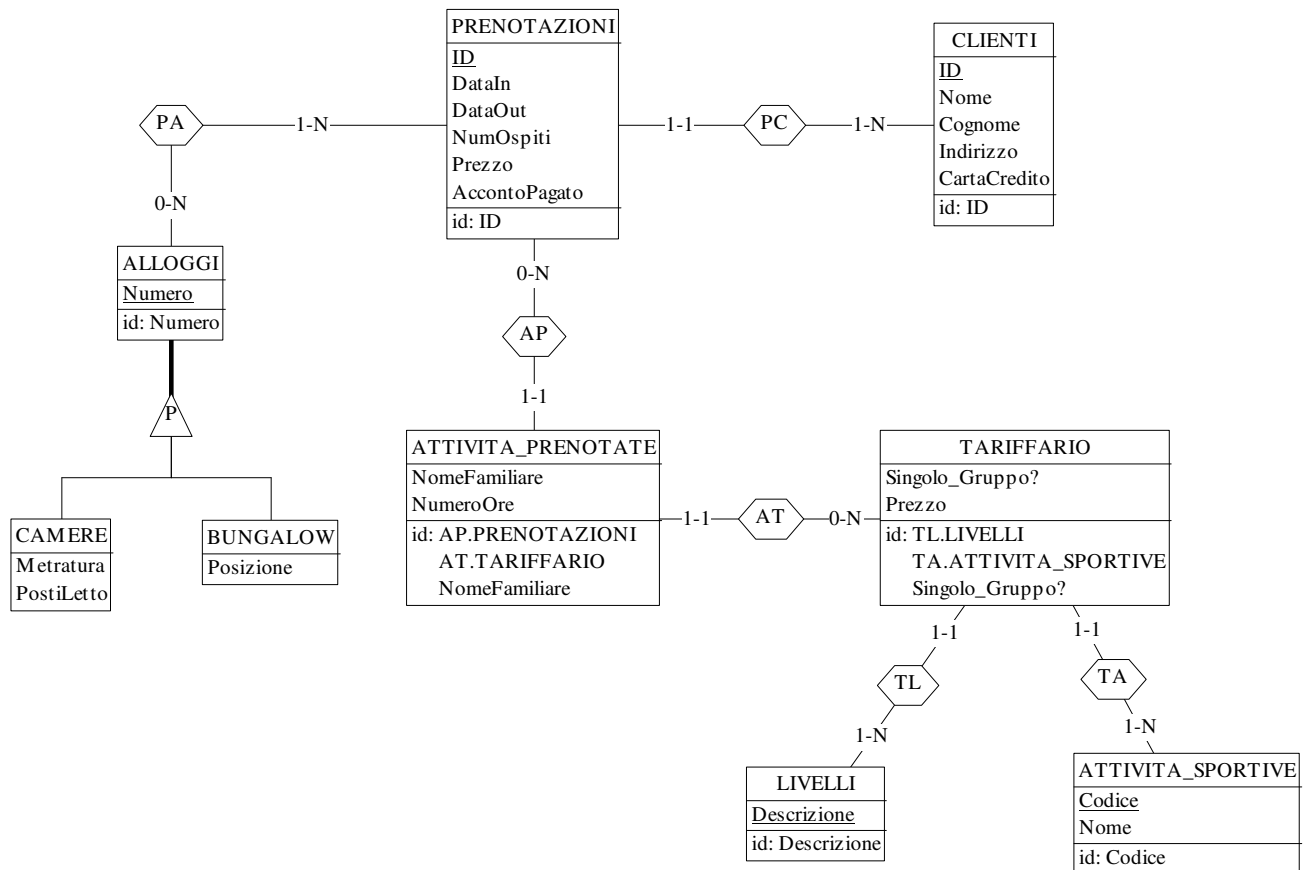
```
WITH
  NUMTERR (CF, Comune, NumTerreni) AS (
    SELECT  P.CF, T.Comune, COUNT(*)
    FROM    TERRENI T, PROPRIETA P
    WHERE   T.IDT = P.IDT
    GROUP BY P.CF, T.Comune
  )

SELECT  N.CF, N.Comune
FROM    NUMTERR N
WHERE   N.NumTerreni = ( SELECT MAX(N1.NumTerreni)
                       FROM    NUMTERR N1
                       WHERE   N1.CF = N.CF )
```

-- La c.t.e., per ogni comune e ogni proprietario, conta il numero di  
-- terreni che quel proprietario possiede in quel comune

## Risoluzione

All'atto della prenotazione i clienti di SUNB devono fornire i propri dati anagrafici e un numero di carta di credito, che viene usato per pagare un acconto pari almeno al 30% dell'importo totale (che il sistema calcola includendo sia l'alloggio che le attività prenotate).



- Le entità **TARIFFARIO** e **ATTIVITA\_PRENOTATE** sono ottenute per reificazione, al fine di rispettare compiutamente le specifiche del testo.
- I vincoli sulla compatibilità degli alloggi e sull'importo dell'acconto non sono rappresentabili in E/R.
- Le entità **LIVELLI** e **ATTIVITA\_SPORTIVE** non sono strettamente necessarie, potendosi i relativi attributi inserire in **TARIFFARIO**. La scelta operata permette di rappresentare meglio che tali entità sono predefinite rispetto al tariffario vero e proprio.
- Si è previsto che una prenotazione possa includere più alloggi, ma questo non è un aspetto critico dell'esercizio.

**4) Progettazione logica (6 punti totali)**

Dato lo schema concettuale in figura e considerando che:

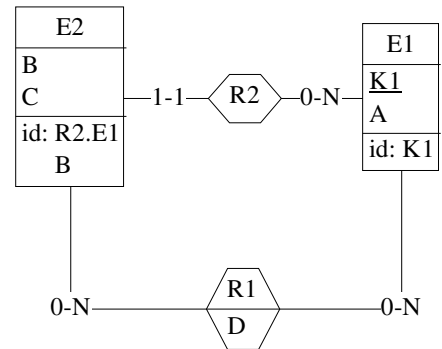
- tutti gli attributi sono di tipo INT;
- in ogni istanza di R1 il valore di D è pari alla somma dei valori di A e C delle istanze di E1 ed E2 associate;

**4.1) [3 p.]** Si progettino gli opportuni schemi relazionali e si definiscano tali schemi in DB2 (sul database SIT\_STUD) mediante un file di script denominato **SCHEMI.txt**

```
CREATE TABLE E1 (
  K1 INT NOT NULL PRIMARY KEY,
  A INT NOT NULL
);
```

```
CREATE TABLE E2 (
  K1 INT NOT NULL REFERENCES E1,
  B INT NOT NULL,
  C INT NOT NULL,
  PRIMARY KEY (K1,B)
);
```

```
CREATE TABLE R1 (
  K1E1 INT NOT NULL REFERENCES E1,
  K1E2 INT NOT NULL,
  B INT NOT NULL,
  D INT NOT NULL,
  CONSTRAINT FK_E2 FOREIGN KEY (K1E2,B) REFERENCES E2,
  PRIMARY KEY (K1E1,K1E2,B)
);
```



**4.2) [3 p.]** Per i vincoli non esprimibili a livello di schema si predispongano opportuni **trigger che evitino inserimenti di tuple non corrette**, definiti in un file **TRIGGER.txt** e usando se necessario il simbolo '@' per terminare gli statement SQL (altrimenti ';')

-- La soluzione migliore consiste evidentemente nel definire un trigger che calcoli automaticamente il valore di D:

```
CREATE TRIGGER R1_D
BEFORE INSERT ON R1
REFERENCING NEW AS N
FOR EACH ROW
SET N.D = ( SELECT A FROM E1
            WHERE N.K1E1 = E1.K1 )
+
( SELECT C FROM E2
  WHERE (N.K1E2,N.B) = (E2.K1,E2.B) );
```

-- Un'alternativa, evidentemente meno precisa, ma comunque corretta considerando le specifiche dell'esercizio, è quella di limitarsi a evitare valori di D non corretti:

```
CREATE TRIGGER R1_D_MENO_PRECISO
BEFORE INSERT ON R1
REFERENCING NEW AS N
FOR EACH ROW
WHEN N.D <> (( SELECT A FROM E1
               WHERE N.K1E1 = E1.K1 )
+
( SELECT C FROM E2
  WHERE (N.K1E2,N.B) = (E2.K1,E2.B) ) )
SIGNAL SQLSTATE '70001' ('Il valore di D deve essere pari a A+C!');
```