```
Utente(username, nome, cognome, dataNascita, email)
AK: email

Evento(codEvento,tipoEvento,nomeEvento,dataEvento,citta)

Prenotazione(codEvento, username, dataPrenotazione)
FK: codEvento References Evento
FK: username References Utente

Partecipazione(codEvento, username, OrarioIngressoEvento)

FK: codEvento References Evento
FK: username References Utente
```

> Selezionare gli utenti che hanno sempre prenotato a tutti gli eventi a cui hanno partecipato

$$S1 = \pi_{codEvento,username}(Partecipazione) - \pi_{codEvento,username}(Prenotazone)$$

Questi sono gli utenti che sono andati ad almeno un evento senza prenotare. Pertanto mi basta togliere dagli utenti il nostro insieme S1.

$$S1 = \pi_{codEvento,username}(Partecipazione) - \pi_{codEvento,username}(Prenotazone)$$
$$\pi_{username}(Utente) - \pi_{username}(S1)$$

Ed eventualmente proiettare

$$S1 = \pi_{codEvento,username}(Partecipazione) - \pi_{codEvento,username}(Prenotazone)$$
$$Utente \ltimes \left(\pi_{username}(Utente) - \pi_{username}(S1)\right)$$

Normalmente la doppia differenza agisce su una **coppia di chiavi** e in generale è indicata dalla parola *"sempre"*.

```
SELECT * FROM Utenti WHERE NOT EXISTS (
      SELECT * FROM Partecipazione WHERE Partecipazione.username =
      Utenti.username
      AND NOT EXISTS (
            SELECT * FROM Prenotazione PR
            WHERE PR.codEvento = P.codEvento
            AND PR.username = P.username
      )
)
```

oppure

```
SELECT * FROM Utenti WHERE
      Utenti.username NOT IN (
            SELECT U.username FROM Utenti U
```

```
                FULL JOIN Prenotazione ON Prenotazione.username =
U.username WHERE Prenotazione.username = NULL
      )
)
```

Supponiamo il seguente schema

```
CREATE TABLE dipendente(
     id INT PRIMARY KEY,
     nome VARCHAR(255) NOT NULL,
     capo INT NULL REFERENCES dipendente(id)
)
```

E la seguente gerarchia

```
INSERT INTO dipendente VALUES
(1, 'Mario Rossi', NULL),
(2, 'Giuseppe Verdi', 1),
(3, 'Maria Bianchi', 1),
(4, 'Luigi Garibaldi', 2),
(5, 'Gianni Leonardi', 2),
(6, 'Lucia Quaranta', 5);
```

Viene richiesto di selezionare, per ogni dipendente, il suo capo e il suo “livello”. La query è ricorsiva

```
WITH Gerarchia (id_dipendente, nome_dipendente, nome_capo,
livello_gerarchico) AS (SELECT id, nome, capo, 1 FROM dipendente
WHERE capo IS NULL)
UNION ALL (
      SELECT D.id, D.nome, D.capo. D.livello + 1
      FROM  Dipendente D WHERE D.capo = Gerarchia.id
)
SELECT * FROM Gerarchia
```

1: è la subquery di base. Sostanzialmente è il primo ramo della gerarchia.
2: è la subquery ricorsiva, che “lavora” su Gerarchia e costruisce, step per step, le varie tabelle.
3: Dà il via all’esplorazione della Gerarchia.