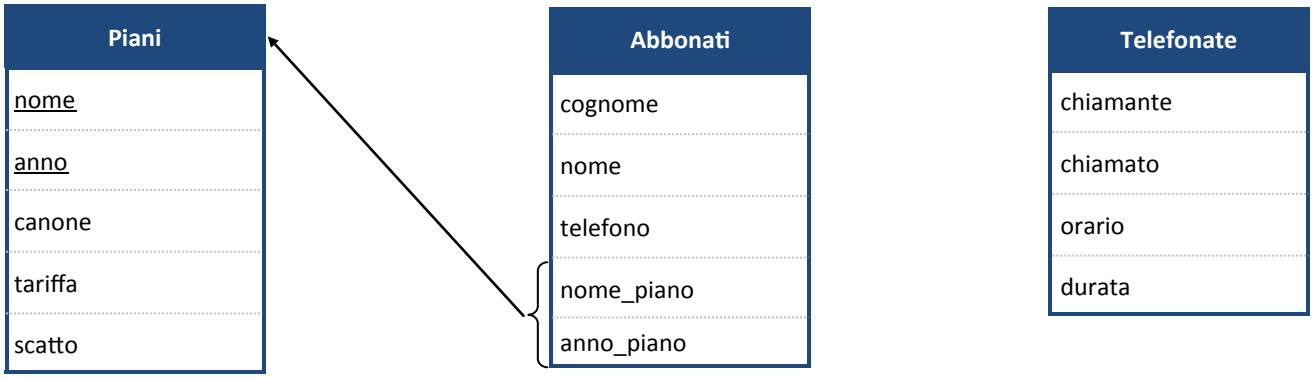
# Fondamenti di Basi di Dati

Esercitazione 3a - Interrogazioni con il join

## Analizzare il database

Apriamo il file **Telefonia.sqlite** con SQLiteStudio, e analizziamo il database, lo stesso usato per la seconda esercitazione. Completare lo schema sotto riportato, rappresentando tutti i vincoli di chiave primaria e i vincoli di integrità referenziale. Saranno particolarmente utili per guidare le operazioni di join.




Verificare che esistono *due* vincoli di integrità referenziale fra gli attributi della tabella **Telefonate** e la tabella **Abbonati**.

## Durata delle telefonate di Mario Rossi

Cerchiamo la durata delle telefonate fatte da Mario Rossi, in ordine decrescente di durata.

La durata delle telefonate si trova nella tabella **Telefonate** (attributo **durata**); il nome e il cognome del chiamante, però, si trovano in **Abbonati**. Occorre effettuare un join tra le due tabelle. La condizione di join è che l'attributo **chiamante** di **Telefonate** sia uguale all'attributo **telefono** di **Abbonati**, in quanto il numero di telefono di colui che effettua le telefonate di cui vogliamo conoscere la durata deve essere uguale al numero di telefono dell'abbonato che ci interessa (cioè Mario Rossi).

Come sempre, prima di eseguire la query è buona prassi analizzare visivamente i dati delle tabelle, per determinare il risultato che ci attendiamo. In questo caso il risultato atteso è costituito dalle durate $\{2, 1, 0\}$.

Scriviamo ora la query:

```
select durata
from Telefonate, Abbonati
where chiamante = telefono
  and cognome = 'Rossi'
  and nome = 'Mario'
order by durata desc
```

La prima delle condizioni della clausola **where** è il predicato di join; le altre sono predicati di selezione, per selezionare solo le telefonate relative a Mario Rossi. Eseguiamo la query e verifichiamone la correttezza.

Proviamo a riscrivere la query esplicitando il join tra le due tabelle:

```
select durata
from Telefonate
  join Abbonati on chiamante = telefono
where cognome = 'Rossi'
  and nome = 'Mario'
order by durata desc
```

## Persone che non hanno risposto a una telefonata

Cercare nome e cognome delle persone che non hanno risposto a una telefonata. Si tratta delle persone che sono state chiamate in una telefonata la cui durata è 0.

Analizzare i dati delle tabelle per constatare che il risultato della query dovrà essere costituito da una sola tupla: (Carla, Rossi). Scrivere la query ed eseguirla.

Osserviamo che la tupla è ripetuta due volte nel risultato: perché? Come è possibile evitarlo?

Come prima, provare a eseguire il join sia implicitamente (condizione di join nella clausola where) sia esplicitamente.

## Prodotto cartesiano di due relazioni

Vediamo che cosa succede se si dimentica la condizione di join. Eseguiamo la seguente query:

```
select *
from Abbonati, Telefonate
```

Stiamo tentando di eseguire un join tra Abbonati e Telefonate senza specificare la condizione di join. Ne consegue il prodotto cartesiano tra le due tabelle: ciascuna tupla della prima tabella è associata in tutti i modi possibili alle tuple della seconda. Verificare che il risultato ha molte tuple (quante? perché?) – e che, inoltre, non ha molto senso!

## Abbonati a un piano tariffario che prevede un canone mensile

Cercare nome e cognome degli abbonati il cui piano tariffario preveda un canone mensile, ossia, in cui l'importo del canone mensile è maggiore di zero. Il risultato dovrà essere ordinato in ordine alfabetico per cognome e nome.

Osserviamo che le informazioni sui piani tariffari si trovano nella tabella **Piani**. È dunque necessario un join fra le tabelle, basato sul confronto di *entrambi* gli attributi che costituiscono la chiave primaria di Piani: il nome e l'anno del piano.

Analizzare i dati delle tabelle per constatare che il risultato della query dovrà essere costituito da tre tuple: {(Pina, Fantozzi) , (Ugo, Fantozzi), (Giovanna, Franchi)}.

Scrivere la query ed eseguirla. Attenzione al fatto che le tabelle coinvolte nel join hanno alcuni attributi con lo stesso nome: occorre disambiguarli. Si consiglia di definire degli alias.

# Costo delle telefonate

Determinare orario e costo di ciascuna telefonata andata a buon fine (cioè con durata maggiore di zero).

Procediamo per passi. Le informazioni che ci occorrono sono suddivise su due tabelle: **Telefonate**, che contiene orario e durata delle telefonate; e **Piani**, che ci occorre per calcolare il costo perché contiene l'importo dello scatto alla risposta e la tariffa al minuto.

Iniziamo a raccogliere tutte le informazioni che ci occorrono in un'unica tabella. Non è possibile effettuare direttamente il join tra **Telefonate** e **Piani**, in quanto non hanno attributi in comune. Occorre "transitare" per la tabella **Abbonati**: a partire da ogni tupla di **Telefonate** che rappresenta una telefonata, dobbiamo prima di tutto "estendere" la tupla con le informazioni sull'abbonato che ha effettuato la telefonata, il che si ottiene con un join tra **Telefonate** e **Abbonati**. Iniziamo a scrivere la query parziale:

```
select *
from
  Telefonate T,
  Abbonati A
where
  T.chiamante = A.telefono
  and T.durata > 0
```

Eseguiamo la query.

Ora che abbiamo eseguito il primo join, conosciamo tutte le informazioni sull'abbonato che ha effettuato la telefonata; in particolare, nome e anno del piano tariffario. Possiamo dunque effettuare un join con la tabella **Piani** per "estendere" ulteriormente la tupla con le informazioni che ci occorrono sul piano tariffario: scatto alla risposta e tariffa. Modifichiamo la query introducendo questo secondo join:

```
select *
from
  Telefonate T,
  Abbonati A,
  Piani P
where
  T.chiamante = A.telefono
  and A.nome_piano = P.nome
  and A.anno_piano = P.anno
  and T.durata > 0
```

Eseguiamo la query.

Ora che abbiamo raccolto tutte le informazioni che ci occorrono, modifichiamo la clausola select per calcolare quanto richiesto: orario e costo delle telefonate selezionate. L'orario è già presente tra gli attributi; il costo è il risultato dell'espressione **scatto + tariffa * durata**. Pertanto la query diventa:

```
select
  T.orario,
  P.scatto + P.tariffa * T.durata as costo
from
```

```
    Telefonate T,
    Abbonati A,
    Piani P
  where
    T.chiamante = A.telefono
    and A.nome_piano = P.nome
    and A.anno_piano = P.anno
    and T.durata > 0
```

Eseguire la query.

Provare a riscrivere la query esplicitando le condizioni di join.

## Da un abbonato a un altro

Determinare cognome e nome degli abbonati che hanno ricevuto una telefonata da Mario Rossi.

L'interrogazione dovrà usare due volte la tabella Abbonati (e non solo): una per cercare l'abbonato Mario Rossi, una per cercare i suoi "amici", cioè gli abbonati da cui Mario Rossi ha ricevuto una telefonata.

Il cammino della query è:

**Abbonati** (1° copia) di nome Mario Rossi

→ **Telefonate** effettute da Mario Rossi (che è il chiamante di tali telefonate)

→ **Abbonati** (2° copia) che hanno un numero di telefono chiamato da tali telefonate

→ Nome e cognome di tali abbonati

## Abbonati con il nome o il cognome che inizia per C e relative telefonate

Per ogni abbonato il cui nome o il cognome inizia con la lettera C, determinare:

- il nome;
- il cognome;
- eventuali numeri chiamati.

In altre parole, se un abbonato il cui nome o cognome inizia per C ha fatto qualche telefonata, vogliamo che nel risultato appaiano tante tuple quante sono i numeri da lui chiamati. Ogni tupla avrà la forma (Cognome, Nome, NumeroChiamato).

Se invece l'abbonato non ha fatto alcuna chiamata, allora il risultato dovrà contenere la tupla (Cognome, Nome, NULL).

Quale tipo di join occorre usare?