Fondamenti di Basi di Dati

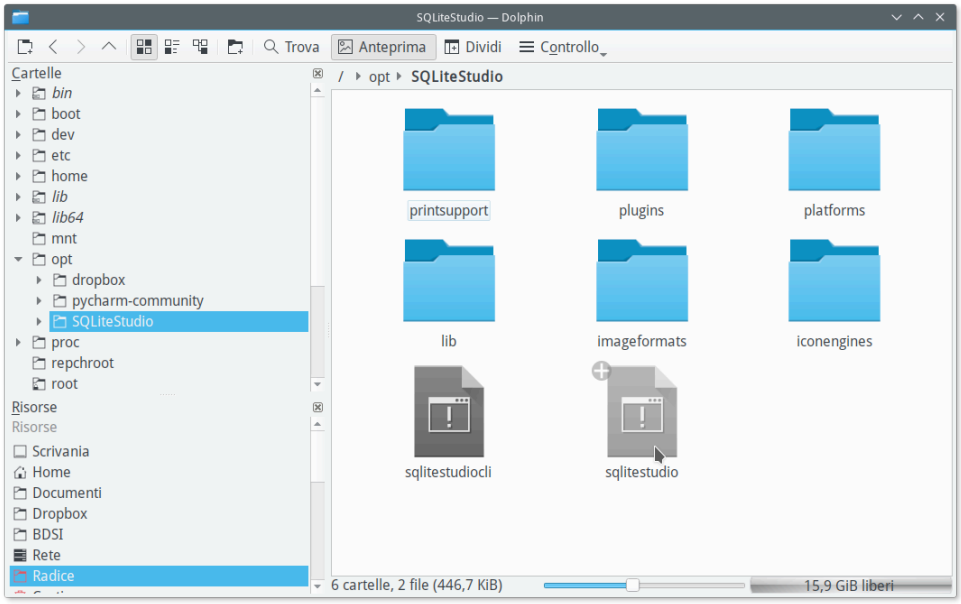
Esercitazione 1 - Creazione di un database con SQLiteStudio

Prepararsi all'esercitazione

Per poter effettuare l'esercitazione è necessario scaricare il programma SQLiteStudio (http://sqlitestudio.pl/). Il programma è gratuito e open source, ed è disponibile per le piattaforme Windows, Linux e OS X.

Inoltre occorre essere in grado di decomprimere un file compresso (.zip). Per gestire i file .zip in Windows è possibile usare il programma 7-zip (http://www.7-zip.org/).

SQLiteStudio è distribuito sotto forma di una cartella compressa in formato .zip. Non è necessaria l'installazione del programma: è sufficiente decomprimere l'intera cartella compressa in una qualsiasi cartella del computer,per esempio sul Desktop. È anche possibile decomprimere la cartella su una pen-drive. **Dopo aver decompresso i file**, aprire la cartella appena creata e fare doppio clic sul file **sqlitestudio.exe** (oppure **sqlitestudio**).



SQLite e SQLiteStudio: un'introduzione

Che cos'è SQLite

La possibilità di rappresentare i dati di interesse attraverso un database è molto utile non solo nel contesto dei sistemi informativi di organizzazioni di grandi dimensioni, ma anche per applicazioni di uso personale. Consideriamo, per esempio, un software di produttività personale quale una rubrica o un client di posta elettronica: gli sviluppatori del software potrebbero decidere di memorizzare alcuni, o tutti i dati che il programma si trova a gestire su un piccolo database. L'alternativa sarebbe rappresentare i dati di interesse direttamente su un file, gestito dal software stesso, il che comporterebbe la necessità di sviluppare tutto il codice necessario per rappresentare i dati e per effettuare le ricerche in modo efficiente. Dunque in molti casi risulta vantaggioso l'impiego di un database.

Qui entra in gioco SQLite, un piccolo DBMS (sistema di gestione di basi di dati) offerto come *libreria*, cioè come modulo software progettato per essere facilmente integrabile all'interno di un programma. Una libreria fornisce allo sviluppatore alcune utili funzionalità. Che cosa offre SQLite? La possibilità di creare dei database. Ogni database è memorizzato all'interno di un file. Il programma naturalmente deve "dialogare" con SQLite per manipolare il database: gli può chiedere, per esempio, di creare una tabella, inserire una nuova riga, cercare dei dati. L'interazione tra il programma e SQLite avviene tramite il linguaggio SQL, il linguaggio standard per la manipolazione di basi di dati. Il programma invia dei *comandi* a SQLite "parlando" il linguaggio SQL, e SQLite esegue in maniera servizievole.

Rispetto ad altri DBMS, SQLite ha dei limiti dovuti allo scopo per cui è stato progettato. Per esempio, non è adatto a gestire basi di dati condivise da molti utenti, proprio perché "vive" all'interno di applicazioni per uso personale. Inoltre SQLite non è pienamente conforme allo standard SQL; ne implementa comunque le funzionalità maggiormente utilizzate.

Infine è interessante sottolineare che SQLite è una libreria di *pubblico dominio*: chiunque ha la possibilità di usarla all'interno dei propri programmi, e di apportare migliorie a questo utile DBMS.

SQLite è impiegato in un gran numero di programmi, sia liberi (*open source*) che commerciali. Ne fanno largo uso i noti browser *Mozilla Firefox* e *Google Chrome*. Per esempio le seguenti informazioni sono memorizzate in un database SQLite: la cronologia delle pagine web visitate; i siti preferiti; le password salvate ecc. Ogni qualvolta l'utente scrive una parte dell'indirizzo o del titolo del sito che vuole visitare nella barra degli indirizzi, il browser chiede a SQLite di cercare, tra i siti visitati e i preferiti, tutti quelli che rispecchiano la ricerca dell'utente.

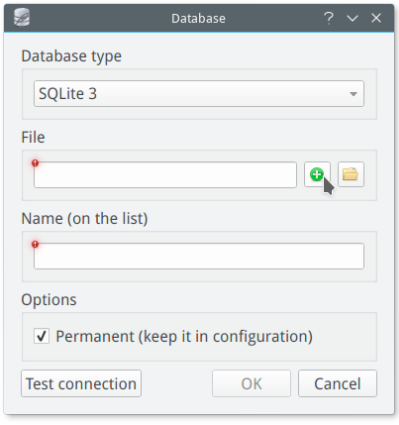
SQLiteStudio

SQLiteStudio è un'*interfaccia utente* con cui l'utente può manipolare direttamente (creare, aprire, visualizzare, modificare) i database SQLite. In altre parole, normalmente i database SQLite sono manipolati dai *programmi* per conto dell'utente, il quale non si rende nemmeno conto di questa interazione. Grazie a SQLiteStudio, anche gli *utenti* hanno la possibilità di vedere e interagire con i database SQLite.

Concretamente, come vedremo, SQLiteStudio offre una finestra che permette di aprire o creare un database, visualizzare le tabelle ivi contenute, aprire una tabella per visualizzarne i dati, manipolare le tabelle (per esempio aggiungendo una colonna o inserendo dati), effettuare ricerche, e, soprattutto, dare comandi direttamente al motore SQLite tramite il linguaggio SQL.

Creare un nuovo database

Apriamo SQLiteStudio e creiamo un nuovo database.



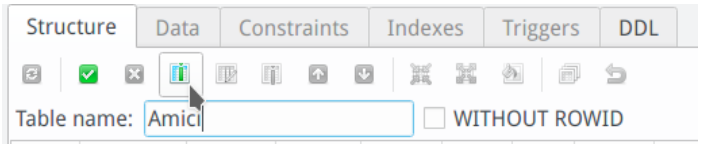
Nella finestra di SQLiteStudio, seleziona il comando **Database -> Add a database**; premi il pulsantino a destra della casella di testo **File** (con il simbolo **+**), seleziona la cartella in cui vuoi salvare il database, e assegnagli un nome, per esempio **Rubrica**. Premi **Save** e quindi **OK**.

Il database apparirà sull'area di sinistra di SQLiteStudio. Fai doppio clic sul nome (**Rubrica**) per selezionarlo ed espandere la struttura del database.

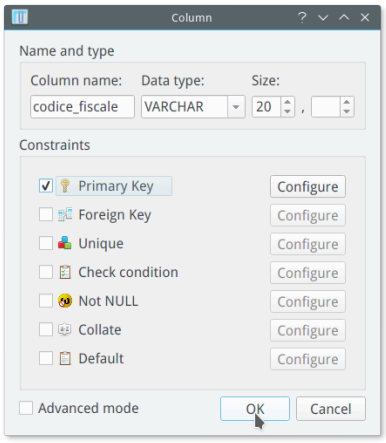
Nota: con una procedura analoga è possibile aprire un database esistente, tramite il comando **Database -> Add a database** e premendo il pulsantino che raffigura una cartella gialla.

Creiamo ora due tabelle: Amici e Telefoni. Useremo la creazione guidata di SQLiteStudio per la prima, mentre per creare la seconda tabella invieremo un comando SQL direttamente a SQLite.

Iniziamo da Amici. Dai il comando **Structure -> Create a table**. Dentro la casella Table Name inserisci **Amici**, e poi aggiungi le colonne una a una. Per aggiungere una colonna, premi il quarto pulsantino (**Add column**).

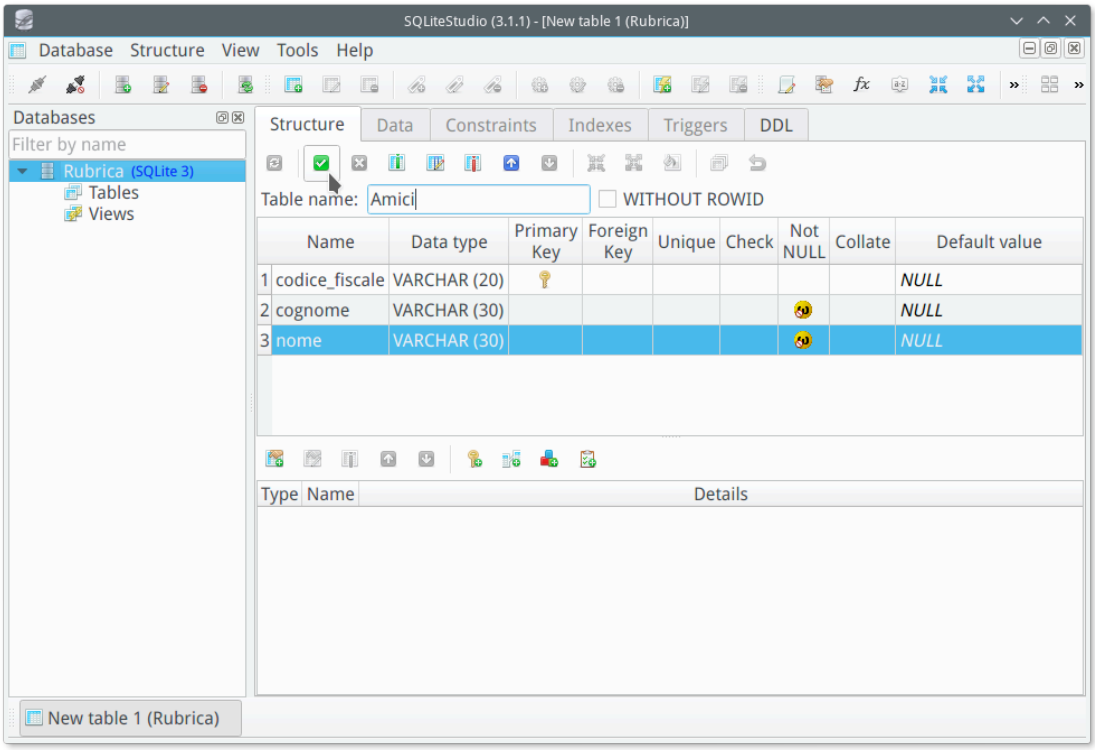


Aggiungi dunque le seguenti colonne, confermando ogni volta con **OK**. Il numero che appare tra parentesi andrà specificato nella casella **size**:



1. codice_fiscale - VARCHAR(20) – Spunta la casella **Primary key.**
2. cognome – VARCHAR(30)
3. nome – VARCHAR(30)

Prestare particolare attenzione agli spazi: i nomi degli attributi non devono contenere spazi, neanche prima o dopo il nome stesso. SQLiteStudio non elimina automaticamente eventuali spazi di troppo, e manipolare nomi di attributi contenenti spazi in SQL è molto macchinoso.Infine, per confermare e creare la tabella, premi il bottone con un segno di spunta verde (**Commit structure change**).



Il sistema compone una query SQL in grado di creare la tabella con le colonne richieste, e la mostra:

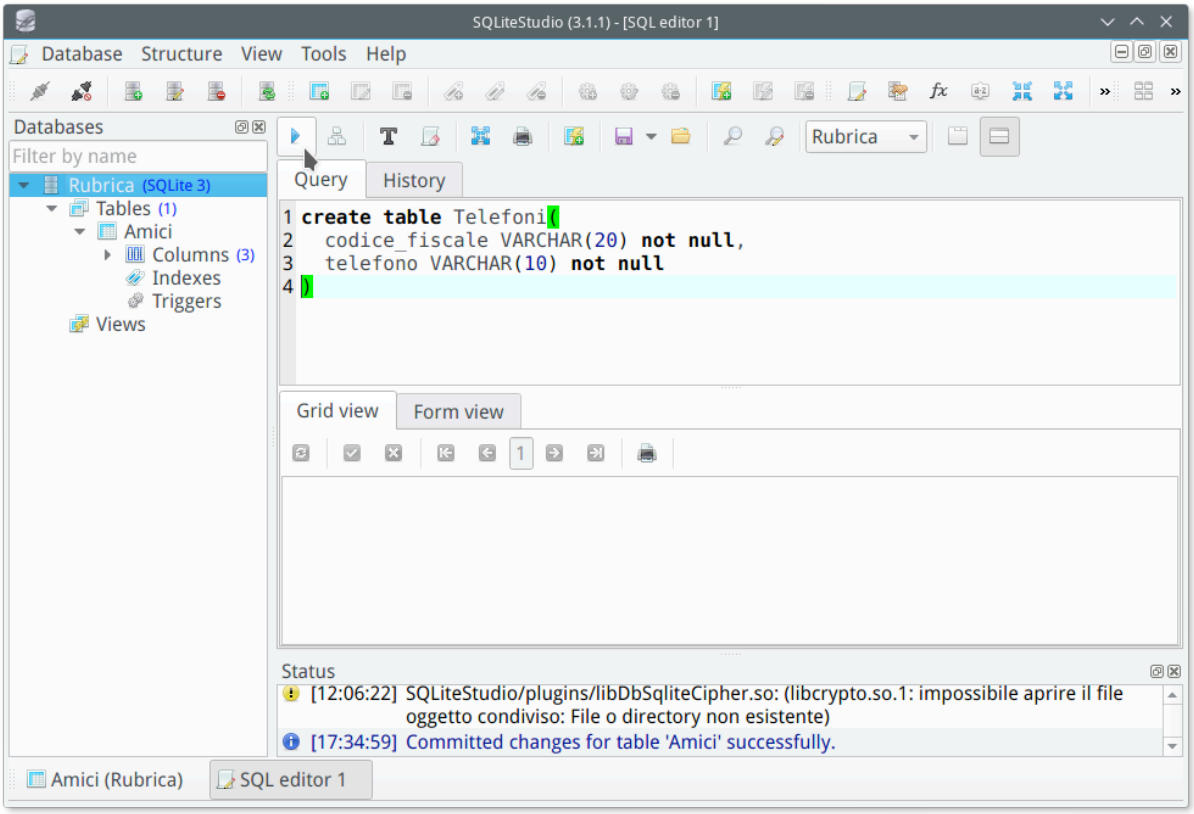
```
CREATE TABLE "Amici " (codice_fiscale VARCHAR (20) PRIMARY KEY, cognome
VARCHAR (30), nome VARCHAR (30)
```

Confermiamo, e la tabella apparirà nell'elenco a sinistra (**Databases**), nella voce **Tables.**

Nota: il pannellino a sinistra (**Databases**) mostra i database aperti, ciascuno con gli elementi che lo compongono (tabelle, colonne ecc.). Qualora per errore dovessi chiudere il pannello, puoi aprirlo nuovamente tramite il comando **View → Databases**.

Ora creiamo "a mano" la tabella Telefoni. Selezioniamo il comando **Tools -> Open SQL Editor**, che ci permette di inviare un comando SQL direttamente al DBMS, e inseriamo il seguente comando nella casella **Query**:

```
create table Telefoni(
  codice_fiscale VARCHAR(20) not null,
  telefono VARCHAR(10) not null
)
```



Premiamo il bottone "Play" (**Execute query**) e, se non abbiamo commesso errori, la tabella Telefoni apparirà nel riquadro a sinistra.

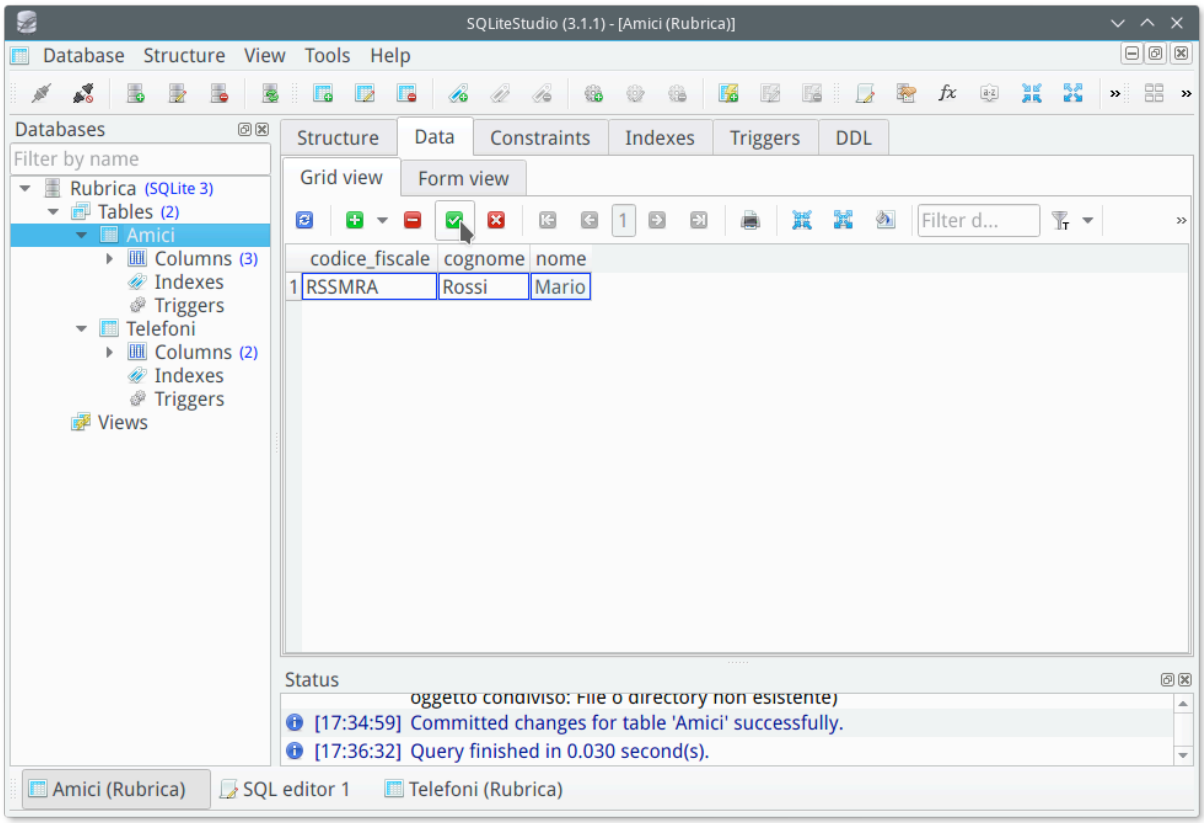
Inserire tuple (righe) nel database

Inseriamo ora alcuni dati nelle tabelle. Come in precedenza, per inserire una parte dei dati useremo l'interfaccia di SQLiteStudio; per inserire gli altri dati costruiremo dei comandi SQL.

Facciamo doppio clic sulla tabella Amici a sinistra; nella finestra di destra, selezioniamo la scheda **Data**, che mostra i dati attualmente presenti nella tabella (nessun dato, a parte i nomi delle colonne). A questo punto premiamo il tasto **+** (**Insert rows**) e inseriamo i seguenti dati:

- ✓ codice_fiscale: RSSMRA;
- ✓ cognome: Rossi;
- ✓ nome: Mario.

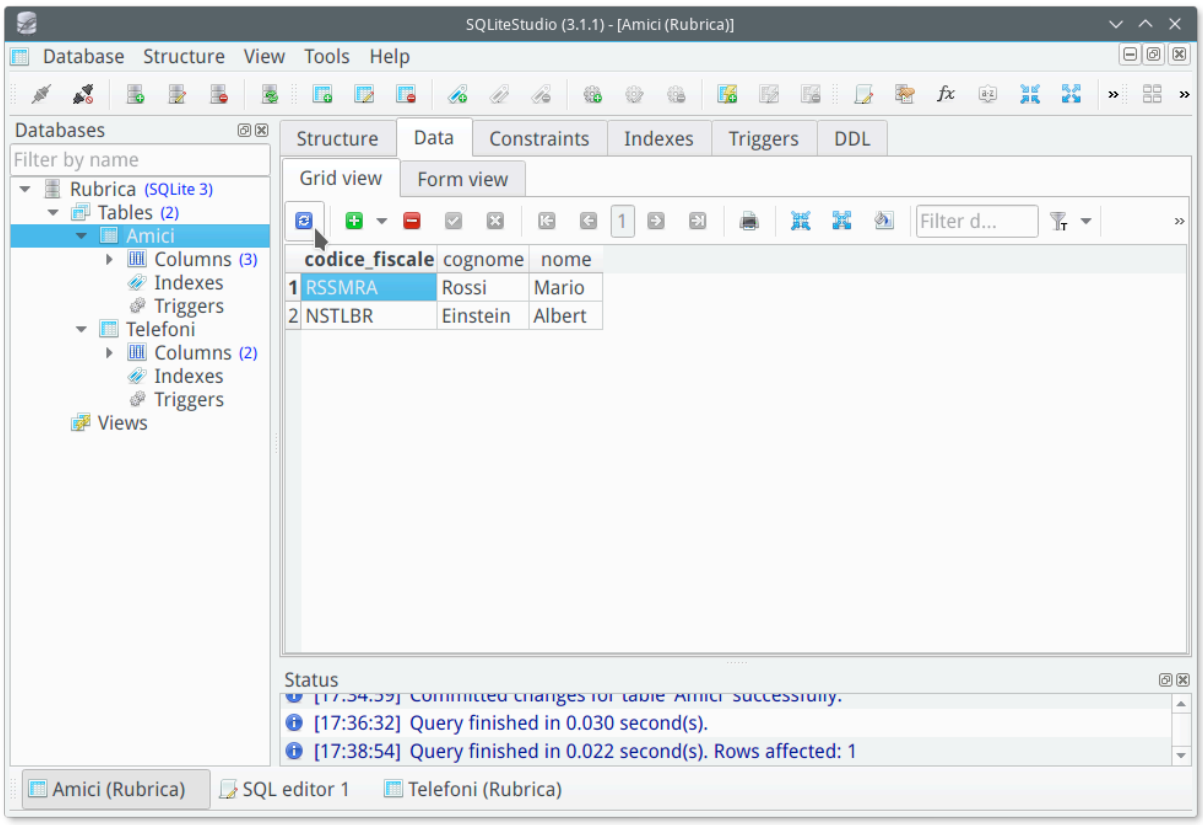
Infine premiamo il simbolo di spunta verde (**Commit**) per rendere definitivo l'inserimento.



Ora aggiungiamo un'altra riga tramite un comando SQL. Torniamo alla finestra **SQL editor** e scriviamo il seguente comando:

```
insert into Amici(codice_fiscale, cognome, nome)
values ('NSTLBR', 'Einstein', 'Albert')
```

Dopo aver eseguito il comando, torniamo nella finestra **Amici**, scheda **Data**, e verifichiamo che sia comparsa la nuova riga. In caso negativo, premiamo il bottone di aggiornamento blu (**Refresh table data**).



Esercizi

Inserimento righe

Aggiungere alla tabella Telefoni alcune righe. Aggiungere la prima usando la finestra di inserimento righe, le altre usando comandi SQL. Le righe da aggiungere sono:

1. codice_fiscale: RSSMRA, telefono: 0637892765
2. codice_fiscale: RSSMRA, telefono: 3389103776
3. codice_fiscale: NSTLBR, telefono: 00101010101

Progettazione e creazione di un database

Ideare un piccolo database per la gestione di una **biblioteca**. Si vogliono rappresentare le seguenti informazioni:

- ✓ Gli **autori**, con un nome, un cognome, una nazionalità (e un codice autore, che sarà utile per la rappresentazione dei libri).
- ✓ I **libri**, con un titolo, un prezzo di copertina, una casa editrice.
- ✓ Un libro può avere più autori. Un autore può scrivere più libri.

Creare il database con SQLiteStudio, e inserire qualche elemento:

- ✓ Il Deserto dei Tartari, autore Dino Buzzati.
- ✓ Bàrnabo delle Montagne, autore Dino Buzzati.
- ✓ Basi di Dati, autori Paolo Atzeni, Stefano Ceri, Piero Fraternali, Stefano Paraboschi, Riccardo Torlone.
- ✓ Progetto di Programmi in Pascal, autori Paolo Atzeni e Maurizio Lenzerini.

Curiosità: esplorare un database di Google Chrome con SQLiteStudio

Abbiamo visto che molti programmi, tra cui Google Chrome, usano SQLite (o altri DBMS) per memorizzare alcuni dati dell'utente. Curiosiamo dunque in uno dei database gestiti da Chrome. Essi sono memorizzati in un'area nascosta del sistema, che contiene le impostazioni dell'utente.

1. Assicurati che Chrome non sia in esecuzione sul PC.
2. In SQLiteStudio, esegui il comando **Database -> Add a database** e premi il pulsantino con la cartella gialla.
3. Nella finestra, in **Nome file** inserisci il percorso riportato in seguito, a seconda del sistema operativo che utilizzi.
4. Nella directory troverai i database di Chrome. Aprine uno, per esempio **History**, il database che raccoglie informazioni sui siti visitati dall'utente.

Nel riquadro a sinistra possiamo vedere le tabelle che costituiscono il database. Facciamo clic, per esempio, sulla tabella **urls**. Nel riquadro a destra selezioniamo la scheda **Data**: potremo visionare il contenuto della tabella. Facendo clic sull'intestazione della colonna **visit_count** ordineremo la tabella per numero di visite, potendo così determinare quale sito abbiamo visionato più volte.

Percorso dei database di Google Chrome

Windows XP

- **Google Chrome:** `C:\Documents and Settings\%USERNAME%\Local Settings\Application Data\Google\Chrome\User Data\Default`
- **Chromium:** `C:\Documents and Settings\%USERNAME%\Local Settings\Application Data\Chromium\User Data\Default`

Windows 10 / 8 / 7 / Vista

- **Google Chrome:** `C:\Users\%USERNAME%\AppData\Local\Google\Chrome\User Data\Default`
- **Chromium:** `C:\Users\%USERNAME%\AppData\Local\Chromium\User Data\Default`

Mac OS

- **Google Chrome:** `~/Library/Application Support/Google/Chrome/Default`
- **Chromium:** `~/Library/Application Support/Chromium/Default`

Linux

- **Google Chrome:** `~/.config/google-chrome/Default`
- **Chromium:** `~/.config/chromium/Default`

