SQuirreL, il client SQL Universale
di Gerd Wagner e Glenn Griffin

(Traduzione italiana a cura di meo)

Hai gia' utilizzato un database relazionale (RDBMS)? Se si, avrai gia' incontrato una o piu' volte una delle seguenti situazioni:
- Scrivere un lungo statement SQL per modificare un valore sul DB.
- Riscrivere lo stesso statement piu' volte, spesso con piccole variazioni.
- Lavorare con piu' database su sistemi differenti.
- Utilizzare database di diversi fornitori come Oracle, MySQL o PostgreSQL.
- Sei un insegnante, uno studente o comunque qualcuno che deve lavorare con un DB ma non sei un esperto di SQL.

Per chiunque debba lavorare con un RDBMS... SQuirreL rende tutto piu' facile!

Cos'e' SQuirreL?

Il client SQL SQuirreL fornisce una semplice interfaccia grafica verso i database relazionali. Poiche' e' realizzato in Java puo' accedere a qualsiasi database JDBC su qualsiasi sistema permettendo un accesso a piu' database contemporaneamente. Un utente di SQuirreL puo':
- visualizzare e modificare dati facilmente su un qualsiasi database JDBC
- visualizzare i metadata del database
- lavorare contemporaneamente su piu' database sia in locale che in remoto
- utilizzare un'interfaccia singola e consistente verso diversi database
- espandere lo strumento in modo da includere funzionalita' specifiche utilizzando i plugin.

L'utente puo' visualizzare e modificare i dati nelle tabelle con un click, oppure utilizzare l'SQL in tutte le sue funzionalita'. I dati possono essere visti in modalita' read-only (sola lettura) per sicurezza, oppure in modalita' modificabile in modo da consentire l'inserimento facilitato nella base dati. Tutti i metadati di un database (eg. tipi di dati, nomi delle colonne, ...) sono disponibili con SQuirreL. Nel caso in cui vengano utilizzati diversi tipi di database (eg. Oracle, MySQL, PostgreSQL, ...) non e' necessario utilizzare ambienti di gestione differenti poiche' SQuirreL-SQL permette di accedere a tutti. Nel caso in

cui un database abbia funzionalita' specifiche, l'architettura a plugin di SQuirreL consente di inserire componenti per trattarne tutti gli aspetti. I plugin permettono di sviluppare funzioni aggiuntive che gli utenti possono scegliere se utilizzare o meno.

Ecco alcuni esempi di utilizzo: la figura 1 mostra l'accesso ad una tabella singola e la figura 2 mostra la finestra SQL che e' ricca di funzioni.

Figura 1: Modifica dati su una singola tabella

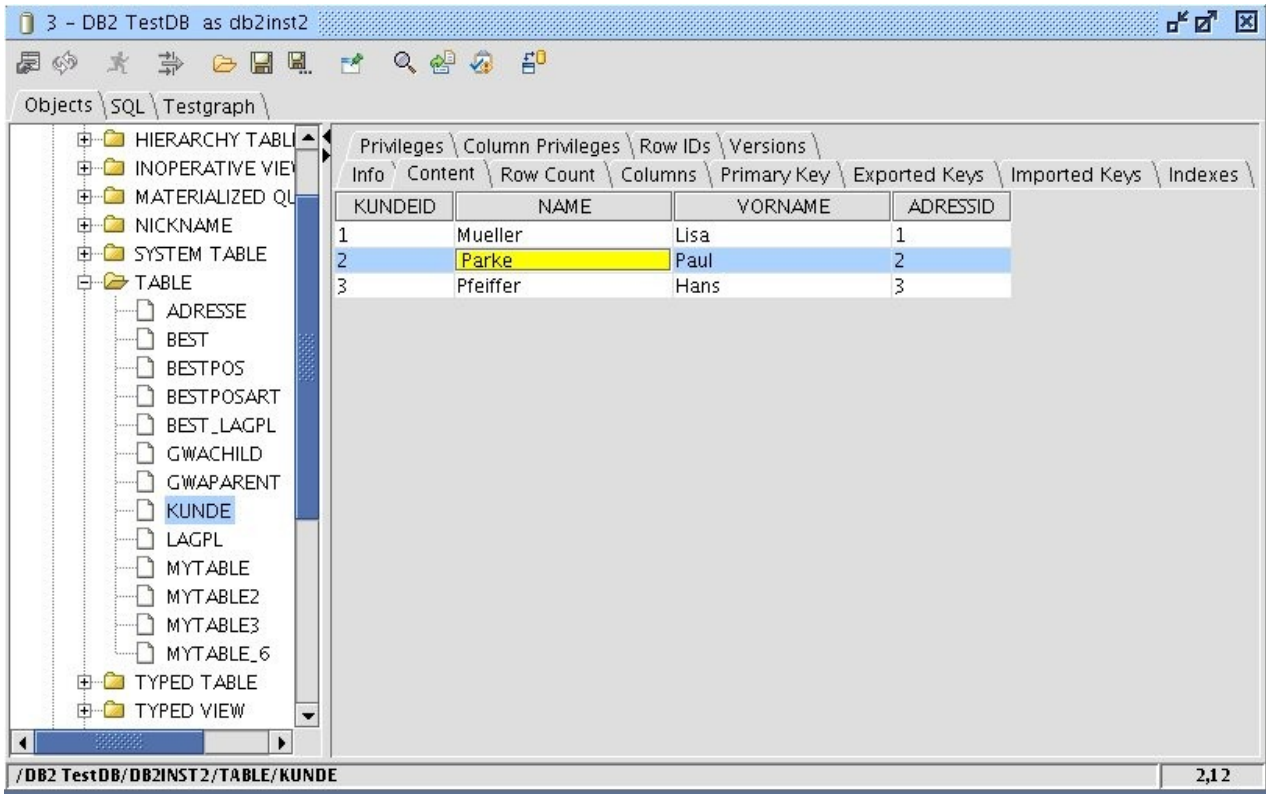
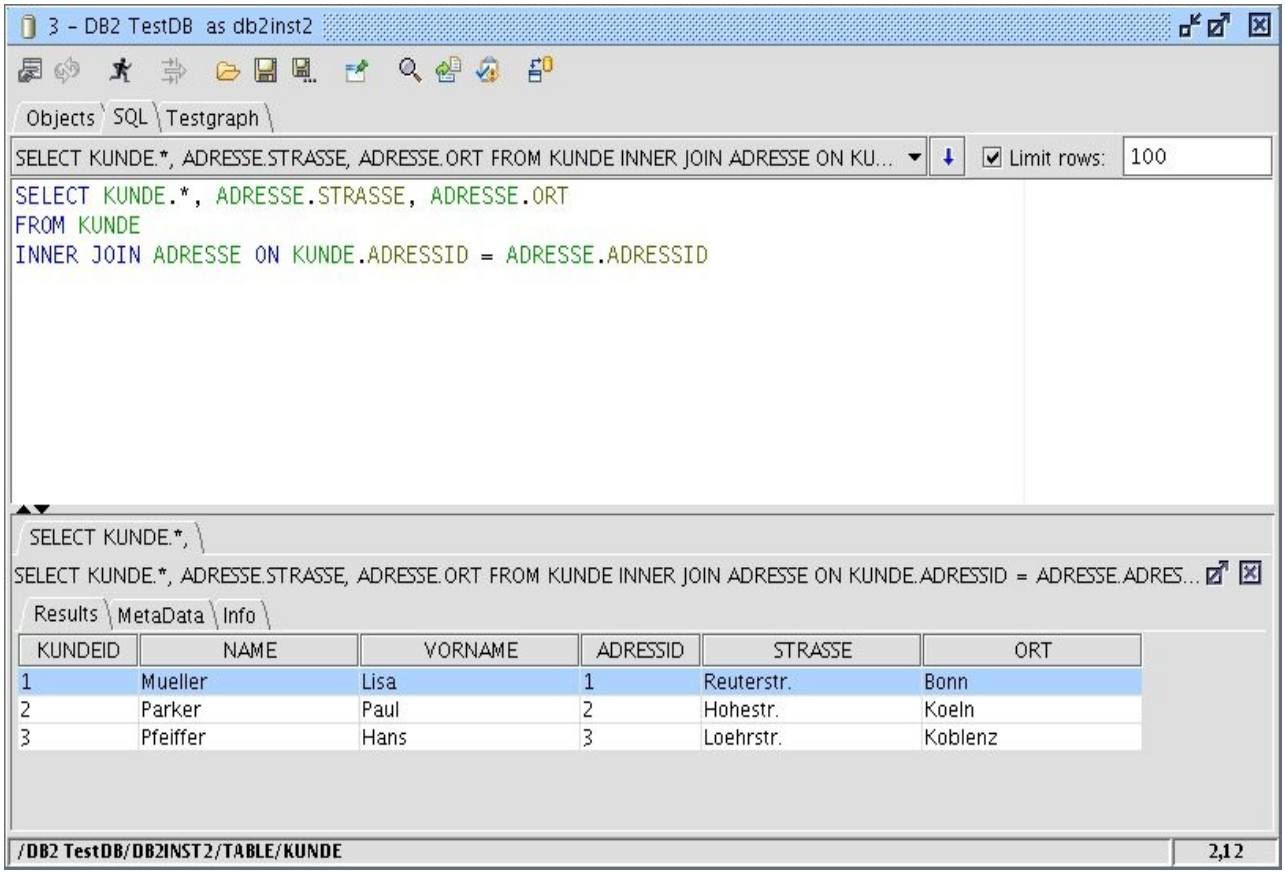


Figura 2: Esecuzione di comandi SQL e DDL



Introduzione

Grazie allo standard JDBC ed alla attuale grande diffusione dei driver, la piattaforma Java e' in grado di accedere praticamente a tutti i database relazionali. Lo standard JDBC ha permesso un livello di uniformita' e di semplicita' prima sconosciuti. Percio' le JDBC API e la piattaforma Java offrono ai programmatori un accesso uniforme e semplice a tutti i database relazionali. Il client Open Source SQuirreL vuole portare questi stessi vantaggi anche agli utenti e non solo ai programmatori.

Nella prossima sezione si vedra' come configurare ed utilizzare SQuirreL per avere un semplice ed uniforme accesso ai database. Quindi si parlera' dei diversi tipi di utenti e di come questi possono sfruttare SQuirreL e le sue funzionalita'. Al termine verranno presentati un programma di esempio ed i passi necessari per creare un nuovo plugin.

Download ed installazione

SQuirreL puo' essere utilizzato e scaricato gratuitamente (con la licenza LGPL) da www.squirrelsql.org. Su sito si trovano:

- il programma di installazione SQuirreL (squirrel-sql-<versione>-install.jar),
- il programma di installazione SQuirreL per MacOS X (squirrel-sql-<versione>-MacOSX-install.jar

I plugin erano in precedenza distribuiti separatamente come file .zip. Tuttavia questo generava qualche problema di versione per gli utenti e gli sviluppatori, cosi' ora tutti i plugin disponibili su www.squirrelsql.org sono inclusi nel programma di installazione. La procedura di installazione consente di scegliere tra l'installazione base, un insieme “standard” di plugin utili alla maggior parte degli utenti e plugin opzionali che arricchiscono SQuirreL con funzionalita' ulteriori.

Prima di installare SQuirreL e' necessario disporre dell'ambiente Java runtime (JRE) nella versione 1.5.x o superiore.

Il file jar di installazione utilizza IzPack ed e' direttamente eseguibile. Dopo le videate per accettare la licenza e selezionare la directory di installazione, la procedura d'installazione chiede se si vuole procedere con l'installazione “basic” o “standard”. L'installazione “basic” contiene tutte le funzioni necessarie per visualizzare e modificare i dati ed i metadati sui DB. L'installazione “standard” contiene una serie di plugin che sono ritenuti utili e che non fanno riferimento a nessuna funzionalita' specifica di un database. Questi plugin sono:

- Code Completion – La funzione di completamento del codice tipica di molti IDE
- Syntax – Evidenziatori sintattici ed abbreviazioni
- Edit Extras – Funzioni ausiliarie per operare in SQL e per la formattazione
- Graph – Crea un grafico con le tabelle e delle relazioni foreign-key tra esse
- SQL Script – Genera script SQL e DDL
- SQL Bookmarks – Gestisce i template SQL
- Look and Feel – Consente di cambiare l'interfaccia grafica

Scegliendo l'installazione basic e' comunque possibile aggiungere uno o piu' plugin successivamente rilanciando l'installazione.

Oltre alla directory di installazione vi sono altre due directory che SQuirreL utilizza:

1. Nella directory di installazione vi e' una sotto-directory chiamata “plugins” dove sono posti i plugin.

2. L'altra directory e' creata quando SQuirreL viene eseguito per la prima volta. La directory contiene gli alias, le definizioni dei driver ed i vari file di history e di personalizzazione. La directory e' posta in C:\Documents and Settings\<username>\.squirrel-sql su Windows, $HOME/.squirrel-sql su Linux and /Users/<username> su Mac OS X.

Vengono prodotte diverse release durante l'anno. Oltre ai regolari rilasci vengono prodotti “snapshot” settimanali realizzati con IzPack ed i sorgenti che contengono le modifiche occorse al codice. L'obiettivo degli snapshot e' quello di fornire una risposta immediata per le correzioni e le nuove funzionalita' al di fuori del normale ciclo delle release.

Ora siete pronti ad utilizzare SQuirreL!

Connessione al database

Connettersi ad un database puo' essere un poco complesso. Con SQuirreL si fa in due passi:

- Ottenere il file JDBC .jar che contiene il driver corretto e dire a SQuirreL dov'e' tale driver
- Definire un collegamento ad uno specifico database su una specifica macchina utilizzando il driver impostato nel passo precedente.

Questi due passi vengono chiamati definire il “Driver” e creare un “Alias”, dove “Alias” puo' essere pensato come una specifica istanza di una configurazione piu' generale di un “Driver”.

Ogni volta che SQuirreL e' avviato apre le finestre Driver e Alias sul desktop come mostrato nella figura 3. Nella finestra Driver si vede un segno blu che indica quali sono i driver disponibili in SQuirreL in questo momento. Se il driver per il database cui si vuole accedere ha una croce rossa e' necessario configurare il driver prima di proseguire. Per farlo e' sufficiente scaricare il driver, salvarlo sul sistema ed infine dire a SQuirreL dove si trova (impostando i dati nella finestra Driver). Possono anche essere aggiunti Driver per database non elencati da SQuirreL.

Il passo successivo e' creare un Alias (figura 4), che descrive una connessione ad uno specifico database su uno specifico sistema. Creando un Alias e' necessario inserire l'URL del database. Questa e' spesso la cosa piu' difficile della configurazione JDBC perche' ogni database utilizza un formato diverso ed alcune opzioni (e.g. i numeri di porta) possono essere specifici di ogni installazione. SQuirreL cerca di aiutare riportando un “suggerimento” che indica la sintassi attesa dal driver. Spesso servono alcune altre informazioni, che l'amministratore della base dati puo' fornire (eg. nome sistema, numero di porta, username, password), necessarie a creare un URL corretto e stabilire una connessione al DB.

Quando e' stato creato un Alias per il database, con un doppio click sul nome si apre una connessione.

Figura 3 SQuirreL desktop

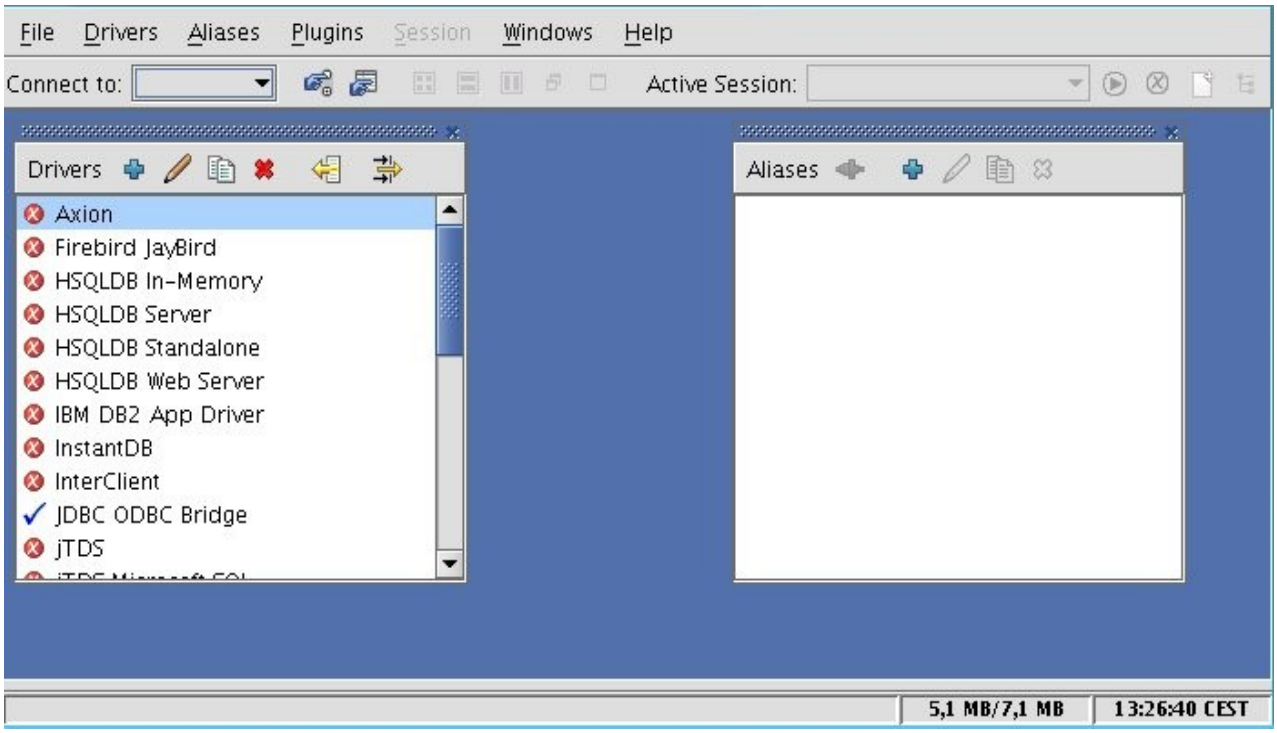
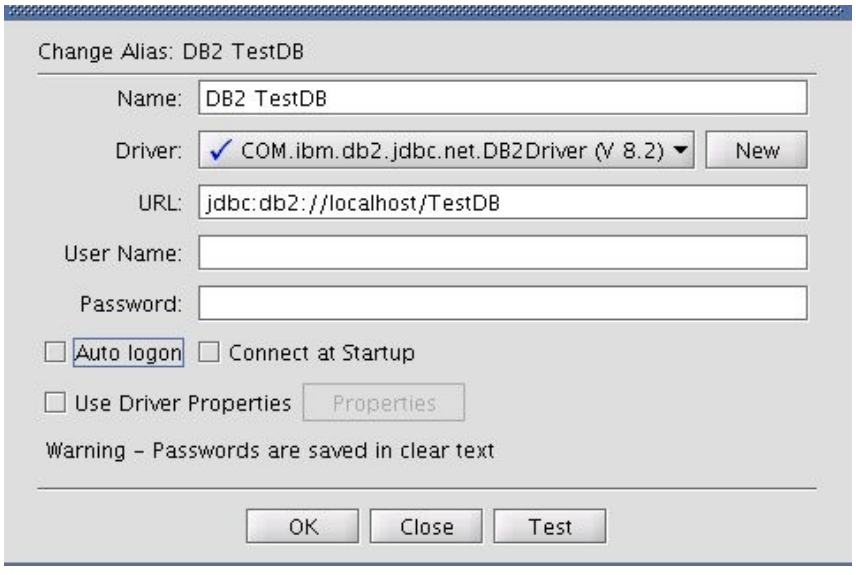


Figura 4 Alias window



Lavorare con un database - una sessione

Quando si apre una connessione ad un database si apre una finestra Session. Una sessione corrisponde ad una connessione con un singolo database. Si possono avere sessioni multiple con piu' database e ciascuna avra' una propria finestra Session.

La filosofia di SQuirreL e' che le cose semplici siano semplici da fare e che le cose complesse siano il piu' facili possibile. Pertanto ogni finestra Session ha due modi per lavorare con un database che corrispondono a due tab nella finestra.

Il **tab Object** fornisce una vista a tabella del database. Tutti i metadati (il tipo dei dati, la dimensione del database, ...) sono riportati in formato tabellare con un click sul database nella vista ad albero del pannello di sinistra e quindi selezionando un tab nel pannello a destra. Con un click su una tabella vengono visualizzati il contenuto della tabella cosi' come i metadati quali la descrizione delle colonne, il numero di righe, ... La tabella puo' essere visualizzata in forma testuale in sola lettura o come tabella modificabile. Quando si visualizza come tabella modificabile ogni cambiamento viene riportato sulla base dati (semplicissimo!). I dati possono anche essere esportati ed importati da file e tutti i tipi di dati, compresi i BLOB ed i CLOB, sono supportati. Si possono inserire nuove righe nella tabella e si possono cancellare righe in modo semplice. Le modifiche possono essere eseguite immediatamente oppure possono essere eseguite nel contesto di una transazione controllata dall'utente.

Il **tab SQL** (figura 2) supporta i comandi SQL. Il tab Object e' molto semplice da utilizzare ma non puo' eseguire operazioni complesse come lavorare su piu' tabelle contemporaneamente con un join, modificare le strutture dei dati come con i comandi “alter column” o “add table” oppure effettuare operazioni specifiche di un database come visualizzare una stored procedure. Il tab SQL consente di introdurre un qualsiasi comando SQL che viene passato al database per l'esecuzione. I risultati sono riportati come tabelle che vengono presentati come testo in sola lettura o, per le selezioni su una singola tabella, come tabelle modificabili. I risultati sono riportati su un pannello con piu' tab che include i metatadi associati alla risposta ottenuta. Il tab SQL ha un combo box con la storia dei comandi lanciati che consente di selezionare un comando precedentemente inserito per una nuova esecuzione o per essere modificato.

Inoltre i plugin aggiungono nuove funzioni all'Editor SQL del tab. La sequenza di tasti Ctrl+T puo' essere utilizzata per ottenere la lista di tutte le funzioni richiamabili (figura 5).

Il popup degli strumenti mostra tutte le funzioni legate all'Editor con un nome per selezionarle, una breve descrizione e, se presente, la sequenza di tasti da utilizzare. Il contenuto del popup puo' essere filtrato con l'iniziale della selezione. In questo modo tutte le funzioni di editing sono disponibili da tastiera e l'unica sequenza di tasti che l'utente deve ricordare e' Ctrl+T.

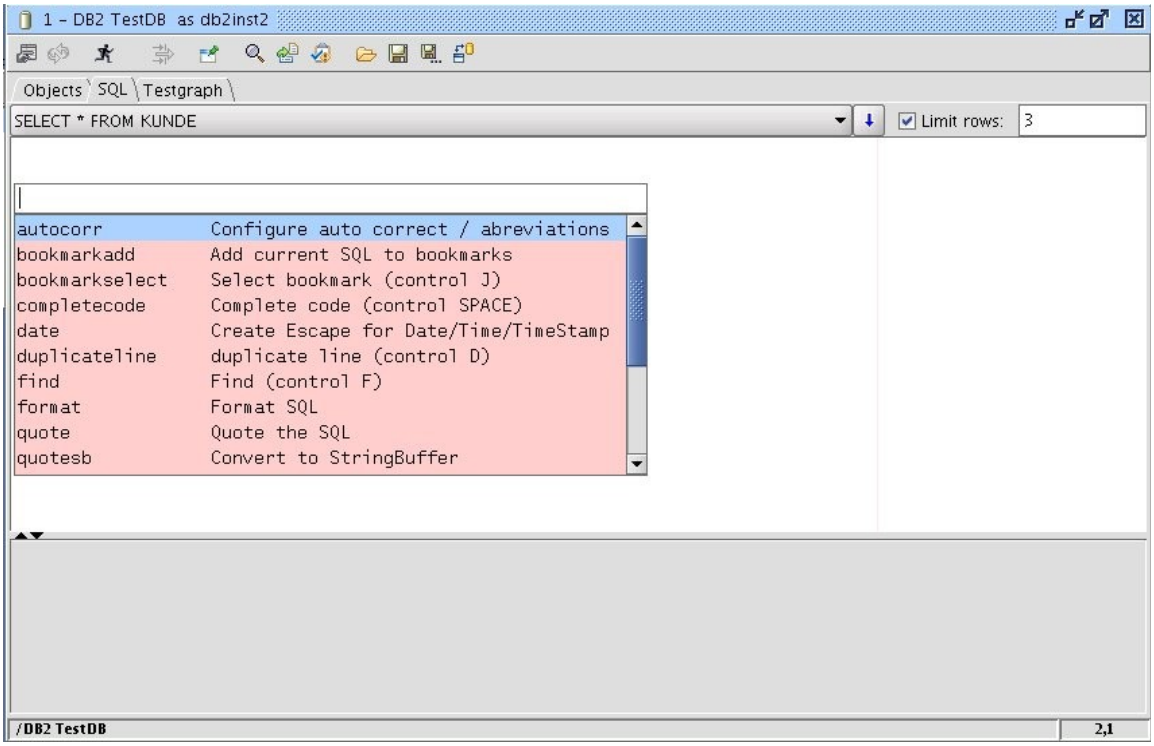
Come in tutti i migliori programmi general purpose SQuirreL permette agli utenti una serie di personalizzazioni. Quando vi sono due modi possibili per fare qualcosa SQuirreL li implementa entrambe e fornisce un parametro che consente all'utente di scegliere quale dei due utilizzare. I parametri si dividono in tre gruppi:

- Le preferenze globali sono impostazioni che sono configurate una sola volta e valgono per tutte le sessioni. Queste opzioni includono quali toolbar/status bar mostrare, se mostrare i suggerimenti, i timeout del JDBC, le impostazioni di debug, la configurazione del proxy e come visualizzare alcuni tipi di dati come i BLOB/CLOB e data/ora.
- Le preferenze di sessione si riferiscono alle singole sessioni. E' possibile impostare il default utilizzato per tutte le nuove sessioni e quindi personalizzarlo per una sessione specifica. Le preferenze di sessione includono dove porre i tab dei pannelli, che tipo di output utilizzare, i limiti su quanti dati estrarre e visualizzare ed i controlli utilizzati nel tab SQL come il carattere di separazione dei comandi.
- Le preferenze specifiche dei plugin permettono all'utente di impostare come opera il plugin estendendo le funzionalita' di SQuirreL. Poiche' alcuni plugin introducono elementi che sono adatti sono per un particolare database (Oracle, DB2, SQL-Server, Sybase, Derby, H2, HSQL, ...)

queste impostazioni si applicano a tutte le sessioni su quel database. Per esempio il carattere di separazione per tutte le sessioni Oracle e' “;”. Lo stesso avviene per le sessioni Sybase con “GO”. Poiche' SQuirreL puo' connettersi a database differenti nello stesso momento questo consente di utilizzare separatori specifici per ogni database in ogni sessione.

Ci sono diverse dozzine di parametri che si possono configurare. I valori di default sono generalmente validi per iniziare ma, una volta acquisita padronanza dello strumento, si possono modificare a piacere.

Figura 5 Tools Popup



Plugin

Le tabelle 1-3 riportano i plugin disponibili su www.squirrelsql.org con una breve descrizione. Questa panoramica iniziale e' seguita da una descrizione dettagliata dei cinque plugin piu' utilizzati.

Tabella 1 Plugin standard (parte dell'installazione standard di SQuirreL)

Nome	***Descrizione***
Look and Feel	Consente di scegliere tra differenti look-and-feels.
SQL Bookmarks	Definizione ed utilizzo di template SQL.
SQL Scripts	Caricamento e salvataggio di script e funzionalita' collegate.
Graph Plugin	Visualizzazione di tabelle e delle relazioni tra esse presenti.
Edit Extras	Diverse funzioni di editor aggiuntive.
Code Completion	Completamento automatico del codice SQL e DDL.
Syntax	Editor sintattico, abbreviazioni ed integrazione dell'editor Netbeans.

Tabella 2 Plugin ufficiali (non contenuti nell'installazione standard)

Name	***Descrizione***
MySQL Plugin	Funzioni specifiche per il database MySQL.
Oracle Plugin	Funzioni specifiche per il database Oracle.
SQL Validator	Controllo della conformita' dell'SQLs agli standards SQL. Il plugin utilizza il Web Service Mimer SQL.

Tabella 3 Plugin Beta

Name	***Descrizione***
Firebird Plugin	Funzioni specifiche per il database Firebird.
Microsoft MSSQL Plugin	Funzioni specifiche per il database Microsoft SQL Server.
Session Scripts	Consente di definire gli script da lanciare all'avvio di una sessione.

Code completion plugin

Il completamente automatico del codice e' una delle funzionalita' piu' utilizzate negli IDE piu' moderni. Il plugin di Code completion (figura 7) fornisce il completamento automatico per quasi tutti i costrutti SQL e DDL:

- Parole chiave, incluse le parole chiave SQL come quelle del driver JDBC
- Tabelle
- Colonne
- Viste
- Stored procedures. Il pluging genera la sintassi completa di chiamata JDBC ed il template per i parametri
- Cataloghi
- Schemi

Inoltre il plugin fornisce la funzione di completamento che puo' essere utilizzata per generare i join SQL. Come esempio si utilizzeranno le tabelle BEST, BEST_LAGPL e LAGPL come in figura 6. Per generare il Join da BEST a LAGPL si scrive la seguente espressione:

#i,BEST,BEST_LAGPL,LAGPL,

Se il cursore e' posizionato alla fine di questa espressione e si preme Ctrl + Space il plugin genera il seguente codice:

```
INNER JOIN BEST_LAGPL ON BEST.BESTID = BEST_LAGPL.BESTID
INNER JOIN LAGPL ON LAGPL.LAGPLID = BEST_LAGPL.LAGPLID
```

Graph plugin

Con il Graph plugin si possono visualizzare gruppi di tabelle con le loro relazioni di foreign key. Inizialmente il plugin inserisce l'item di menu 'Add to graph' al menu delle tabelle. Quando l'utente seziona questo item viene creato un nuovo tab nella finestra principale. Questo tab mostra il grafo delle tabelle selezionate (cfr. figura 6).

La finestra principale Session puo' avere un numero grande a piacere di tab Graph. L'utente puo' rinominare i tab e salvarli. Quando la finestra Session viene riaperta la volta successiva anche i tab Graph vengono riaperti. In questo modo l'utente puo' avere a disposizione le tabelle piu' importanti e relazioni con un singolo click del mouse.

Tutte le colonne foreign key sono indicate con un '(FK)' al termine del loro nome (cfr. figura 6). Con un doppio click su una colonna foreign key la tabella a cui punta la chiave viene aggiunta al Graph. Utilizzando il menu di contesto di una tabella tutti le tabelle "genitori", tutte le tabelle "figlie" o entrambe possono essere aggiunte al Graph, questo fornisce un semplice modo di navigare lungo la struttura della base dati.

Le API dei Plugin di SQuirreL consentono ad ogni plugin di comunicare con gli altri. Grazie a questo il plugin Graph utilizza il plugin SQL Scripts. In questo modo tutti gli elementi visualizzati in un Graph possono essere trasformati in statement DDL con un click del mouse.

I grafici creati dal plugin possono essere semplicemente allargati o ristretti per essere stampati su una o su piu' pagine.

Syntax plugin
Il plugin Syntax mostra la potenza delle plugin API di SQuirreL perche' sostituisce una delle parti centrali di SQuirreL, l'editor SQL, con un nuovo componente: l'editor Netbeans (http://editor.netbeans.org). Il plugin Syntax si occupa di colorare il codice SQL. L'utente, nelle proprieta' della sessione, puo' scegliere il colore utilizzato per le parole chiave, per i nomi di tabella, per le colonne, ... Il plugin consente all'utente di definire le correzioni automatiche e le abbreviazioni. Le correzioni automatiche e le abbreviazioni si comportano nello stesso modo dei comuni prodotti di Office. Possono essere configurati oppure essere completamente disattivati.

Script plugin
Il plugin Script crea script SQL basati sulle tabelle visualizzate nel tab Object o in un Graph. Script di Insert possono essere generati basandosi sull'intero contenuto di una tabella o dei soli dati selezionati da una clausola WHERE. Possono anche essere generati script per memorizzare i risultati di una query SQL in una tabella temporanea. In questo caso l'utente puo' anche scegliere se lanciare immediatamente lo script o salvarlo per un uso successivo.

SQLBookmarks plugin
Con il plugin Bookmarks l'utente puo' definire dei template per il codice SQL e DDL. Per inserire velocemente un template nell'editor SQL si utilizza Ctrl + J. Questo apre un popup simile al popup dei tools. Il popup permette di scegliere uno tra i diversi template.

Il plugin include template per i piu' comuni statement SQL e DDL. Questo rende molto piu' semplice imparare l'SQL ed il DDL.

Figura 6 Graph plugin

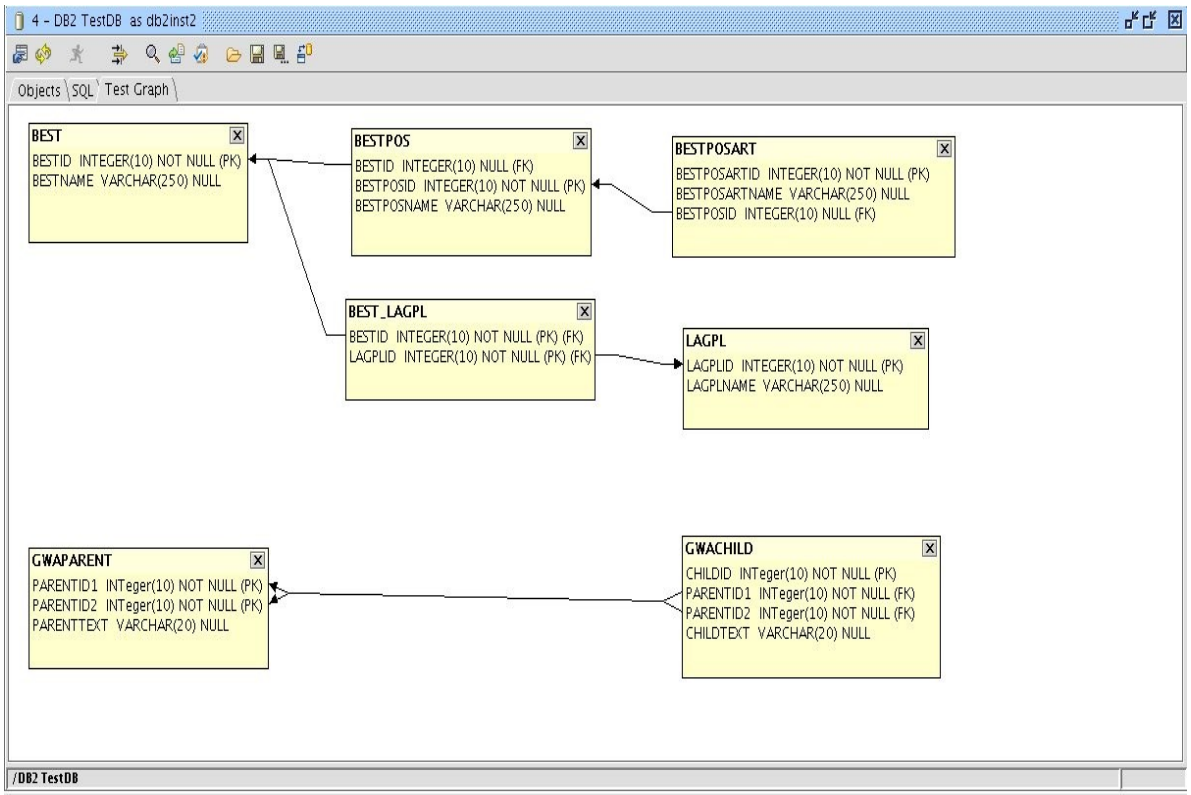
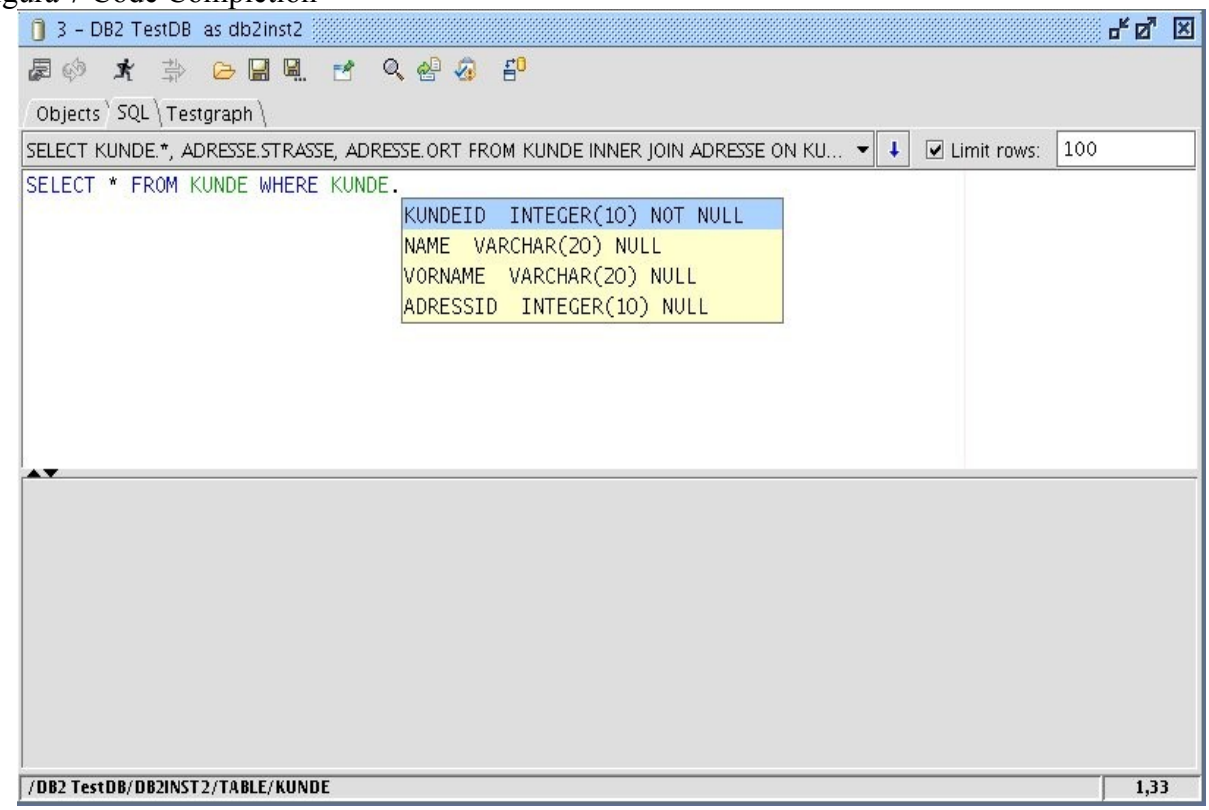


Figura 7 Code Completion



A cosa serve SQuirreL

E' stato detto che SQuirreL rende piu' semplice il lavoro di chi utilizza i database ma utenti differenti hanno necessita' differenti a seconda dei propri compiti. In questa sezione si vedra' come SQuirreL puo' aiutare tipi di utenti differenti. Verranno anche descritte alcune funzioni di SQuirreL in maggior dettaglio.

- Amministratori di applicazioni
  Il compito dell'amministratore di un applicazione e' quello di risolvere i problemi di un'applicazione, compresi quelli relativi ai dati nel database. In un tipico scenario un utente segnala un problema sui dati che non puo' essere risolto utilizzando l'applicazione. Si puo' trattare di un valore non corretto in un campo, di un riferimento mancante su una tabella di collegamento, una riga in piu' in una tabella o qualsiasi altra anomalia sui dati. Utilizzando il tab Object di SQuirreL l'amministratore puo' velocemente trovare il problema visualizzando il contenuto delle tabelle sullo schermo. I valori errati possono essere corretti semplicemente digitando i valori corretti. Righe possono essere inserite o cancellate con alcuni click del mouse. Per le tabelle di maggiori dimensioni SQuirreL fornisce una finestra di popup che consente di limitare il numero di righe ricercate aggiungendo condizioni alla clausola WHERE.

  Quando il problema riguarda relazioni complesse tra i dati il tab SQL puo' essere utilizzato per inserire selezioni SQL. Tali selezioni possono utilizzare l'SQL standard e tutte le estensioni che il motore relazionale supporta. Se e' necessario ripetere una selezione o eseguire una selezione simile alle precedenti e' possibile prendere una o piu' linee dall'history SQL e modificarle come necessario prima di eseguirle. Il risultato e' visualizzato come una singola tabella ed i dati possono essere modificati sullo schermo e la tabella modificata direttamente. Per fare semplici correzioni si puo' utilizzare l'Object tab altrimenti si possono fare le modifiche piu' complesse sfruttando a pieno l'SQL.

  Compiti complessi che debbono essere eseguiti in modo ripetitivo (pulizia notturna o mensile, caricamento periodico di tabelle, ...) possono essere automatizzati con il plugin SQL Script.

  Tutte le tabelle in SQuirreL possono essere copiate selettivamente come liste o in formato HTML per essere incollate su documenti o pagine web.

- Programmatori e collaudatori di programmi
  Quando si scrive un programma o si testa un'applicazione debbono essere provate funzionalita' specifiche e provare percorsi differenti. SQuirreL aiuta i programmatori in diverse cose:
  1. Definire i dati prima di eseguire un test, controllare i dati dopo il test, correggere i dati corrotti dal test.
  2. Controllare e correggere velocemente i dati durante un System Test o su un ambiente di produzione.
  3. Fornire un accesso remoto semplice e consistente tra macchine differenti ad esempio: unit test, system test e production.
  4. Fornire un simpatico ambiente per generare uno statement SQL, per modificare l'SQL gia' scritto in modo semplice, per gestire i dati usati dallo statement, per eseguire i comandi e per analizzare i risultati.

Molte di queste attivita' sono simili a quelle di un amministatore applicativo, perche' richiedono di controllare e modificare i dati nella base dati, pertanto quanto riportato nelle sezioni precedenti vale anche in questo caso.
La differenza principale e' che i programmatori e chi esegue il collaudo deve spesso ripartire con un certo insieme di dati. Il plugin di SQL Script menzionato nella sezione precedente puo' aiutare su questa attivita'.

Il plugin Code Completion fornisce il completamento dell'SQL e del DDL e genera la corretta sintassi per i join.

Il plugin Syntax controlla la sintassi degli statement e consente all'utente di definire abbreviazioni e l'auto correzione. Assieme ai Code template introdotti dal Bookmarks plugin, SQuirreL e' un potente generatore di codice.

Molto spesso gli sviluppatori debbono provare stringhe SQL di un programma oppure statement SQL scritti e provati su un client SQL debbono essere inseriti nel codice di un programma. In questi casi i delimitatori di stringa debbono essere inseriti o rimossi dallo statement. Con l'aiuto del plugin Edit Extras questo si puo' fare con un click.

Un altro problema comune per i programmatori e' quello di analizzare stringhe SQL da file di trace o debug. Molto spesso queste stringhe non hanno alcuna formattazione: sono lunghe stringhe su una sola linea. Il plugin Edit Extras fornisce un modo semplice di formattare gli statement SQL. Gli statement di Insert sono formattati in modo che i nomi delle colonne si trovino sopra i relativi valori. Il plugin formatta anche i valori Timestamp/Date/Time nella sintassi standard JDBC con un singolo click.

Molte applicazioni utilizzano viste e Stored Procedures. Le viste e le Stored Procedures possono essere eseguite in SQuirreL, ma non c'e' uno standard SQL per memorizzarle. Poiche' il metodo per leggerle e modificarle dipende dal prodotto e' necessario un plugin specifico del prodotto per avere questa funzionalita'. Molti gia' esistono, ma potrebbe essere necessario crearne uno per un nuovo tipo di database. Si vedra' quanto e' facile nella sezione ‘Programming plugins’.

Come nota finale, gli sviluppatori di driver JDBC troveranno che SQuirreL e' un ottimo ambiente di test. Non solo e' facile aggiungere e sostituire i driver, ma SQuirreL sfrutta tutti i componenti di un driver JDBC e fornisce i risultati su tabelle di semplice lettura che possono essere formattate in HTML e copiate su documenti e pagine web.

- Database administrator
  Per gli amministratori di database (DBA) SQuirreL fornisce un accesso semplice e comune per tutti i sistemi locali e remoti di cui sono responsabili. Un DBA piu' utilizzare SQuirreL per controllare i parametri del database o controllare la dimensione delle tabelle.

  Rispetto agli altri gruppi di utenti i DBA si occupano di aspetti piu' specifici dei database. Poiche' si tratta di elementi non standard queste peculiarita' non sono supportate dal codice base di SQuirreL. Ma le API dei plugin sono sufficentemente sofisticate per supportate anche le funzioni piu' critiche di un database.
  Poiche' il plugin e' un codice Java puo' attivare un processo separato cosi', anche quando una funzione non e' richiamabile con un API, il plugin puo' comunque eseguirla richiamando un tool fornito dal produttore della base dati. In questo caso il vantaggio e' che l'amministratore non deve necessariamente conoscere i dettagli dei tool richiamati.

Sono gia' stati realizzati diversi plugin specifici per prodotto. Nonostante questo suggeriamo ai DBA interessati alla programmazione Java di dare un'occhiata alla sezione "Programmare i plugin" e di scrivere plugin SQuirreL specifici per prodotto. Se si debbono amministrare differenti prodotti di database e' molto utile un client che renda univoche le funzionalita' standard e che sia in grado di gestire le variazioni specifiche dei prodotti.

- Utenti che imparano o insegnano l'SQL
  Per chi sta iniziando ad imparare l'SQL, il plugin Bookmark fornisce i template per molti comuni statement SQL e DDL. Il plugin consente anche di definire propri template.

  Il plugin Syntax controlla immediatamente la sintassi SQL ed evidenzia gli errori con un testo colorato. Se il puntatore del mouse e' posto su un errore viene mostrato un tool tip con un messaggio.
  Il plugin Graph (figura 6) genera una vista grafica delle tabelle e delle foreign key del database e questo da' agli studenti una semplice visione d'insieme delle strutture relazionali. Le funzioni di scripting del Graph permettono all'utente di tradurre la rappresentazione visuale nuovamente in codice DDL e quindi chiarire le relazioni tra entrambe.

  Per gli studenti e' molto importante capire che tutti i database relazionali seguono principi comuni che sono indipendenti dal fornitore di database. Poiche' SQuirreL consente un accesso unificato a praticamente tutti i database relazionali, questi principi vengono evidenziati in modo particolare.

- Per chi debba familiarizzare con un database gia' esistente
  Quando si entra a far parte di un progetto gia' avviato, comprendere come e' organizzato il database richiede tempo. Il plugin Graph e' di aiuto mostrando le tabelle e le loro relazioni organizzate in uno o piu' grafi. I grafi possono essere salvati con nomi definiti dall'utente e vengono riaperti quando si apre novamente una sessione. I grafi possono essere messi in scala e stampati su piu' pagine.

  Scrivere statement SQL puo' essere difficile quando non si ha familiarita' con nomi di tabelle e di colonne. Il plugin Code Completion puo' completare in automatico il codice SQL e DDL come un IDE. La finestra di pop up di Completion (figura 7) mostra i nomi delle colonne e le loro principali proprieta'. Nel caso in cui questo non sia sufficiente si puo' porre il cursore sul nome della tabella o della vista e controllare la sua definizione nell'albero degli oggetti.

Programmare i plugin

Noi incoraggiamo la scrittura di plugin per SQuirreL. Poiche' i plugin sono indipendenti dal codice centrale di SQuirreL, lasciamo molta liberta' nella loro scrittura. D'altra parte, se si vuole costruire un plugin, e' una buona idea contattarci prima. Per prima cosa potrebbe gia' esserci qualcuno che sta lavorando sulla stessa funzionalita'. Un'altra ragione puo' essere la necessita' di arricchire le API dei plugin. Se si sta programmando una funzionalita' probabilmente questa sara' utile anche ad altri. La condizione che preferiamo e' che i nuovi plugin siano rilasciati sotto la stessa licenza LGPL di SQuirreL, ma se chi li realizza non e' di questa idea puo' comunque far conto del nostro supporto. Per ogni informazione siamo raggiungibili tramite la Mailing List squirrel-sql-develop riportata nella home page del sito www.squirrelsql.org

Nel seguito descriveremo la programmazione dei plugin creando un semplice esempio che visualizza la definizione di una vista o di una Stored Procedure. Questo viene fatto copiando la definizione nell'editor SQL, in questo modo qualsiasi cambiamento effettuato non causa nessuna modifica nella base dati. Per modificare le viste o le Stored Procedure l'utente deve eseguire i normali comandi DDL. SQuirreL richiede un plugin per accedere a queste definizioni poiche' ogni produttore di database memorizza ed accede a queste definizioni in modo proprietario.

Anche se il codice dell'esempio e' relativo al dababase DB2 di IBM puo' essere utilizzato come template per creare plugin che operano con prodotti di altri fornitori.

Il codice sorgente completo e' disponibile nel modulo sql12 del repository CVS di SQuirreL. Sul sito www.squirrelsql.org si trovano tutte le informazioni necessarie per effettuare il check out del modulo. Il codice sorgente verra' scaricato nella directory di checkout plugins/example/.

Per fare in modo che SQuirrel utilizzi un plugin questo va compilato come file Jar. Il file Jar va copiato nella directory di plugin dell'installazione di SQuirreL. All'avvio SQuirreL carica tutti i file Jar dei plugin e cerca in ogni Jar l'implementazione dell'interfaccia IPlugin. Questa implementazione e' la classe centrale di un plugin. Nell'esempio questa classe e' chiamata ExamplePlugin. ExamplePlugin e' derivata da DefaultSessionPlugin che a sua volta implementa IPlugin. I metodi principali di ExamplePlugin sono initialize() e sessionStarted(), come riportato nel Listato 1. Per controllare se SQuirreL ha caricato correttamente un plugin basta richiamare il menu "Plugins" --> "Summary".

Il metodo initialize() e' richiamato una volta alla partenza di SQuirreL. Nell'implementazione di initialize() vengono caricate le risorse, nell'esempio il file example.properties. Questo file contiene le label degli elementi del menu che verranno aggiunti all'albero degli oggetti. Le label sono '(DB2) Script

View' e '(DB2) Script Procedure'.

Il metodo sessionStarted() viene richiamato quando viene aperta una sessione. Nell'esempio il primo controllo e' se la sessione aperta e' per un database DB2. Se non lo e' il plugin resta inattivo. Se invece si tratta di un database DB2 vengono aggiunti al menu degli oggetti i nodi per le viste e per le Stored Procedure. Se l'utente seleziona '(DB2) Script View' viene richiamato il metodo actionPerformed() della classe ScriptDB2ViewAction (cfr. Listato 2).

Nel metodo actionPerformed() di ScriptDB2ViewAction vengono prima raccolti gli oggetti selezionati nell'albero. Quindi e' necessario raccogliere l'attributo 'simple name' di tali oggetti. Questo non corrisponde ad altro se non al nome della vista. Quindi segue un codice specifico: si accede alla tabella di sistema 'SYSIBM.SYSVIEWS' di DB2. La restante parte del codice e' necessaria per visualizzare la definizione della vista nell'editor SQL.

Il codice per le Stored Procedure e' simile e non e' riportato.

Listato 1 ExamplePlugin:

```
public synchronized void initialize() throws PluginException
{
   // Initializing the resources in example.properties.
   // example.properties contains the labels for the menu items.
   _resources = new PluginResources
      ("net.sourceforge.squirrel_sql.plugins.example.example", this);
}


/**
 * Called when a session is opened.
 * The menu items are added are added here.
 *
 * @param   session  The started Session.
 */
public PluginSessionCallback sessionStarted(ISession session)
{
   try
   {
      Connection con = session.getSQLConnection().getConnection();
      String driverName =
         con.getMetaData().getDriverName().toUpperCase();

      if(false == driverName.startsWith("IBM DB2 JDBC"))
      {
         // The plugin only knows how to script views and stored
         // procedures for the IBM DB2 database. So if this is
         // not an DB2 Session we tell SQuirreL not
         // to use the plugin.
         return null;
      }

      // Add entries to the view and stored procedure
      // nodes in the Object tree.
      IObjectTreeAPI otApi =
         session.getSessionInternalFrame().getObjectTreeAPI();
```

```
      ScriptDB2ViewAction viewAct =
         new ScriptDB2ViewAction(getApplication(),
                                 _resources,
                                 session);

      otApi.addToPopup(DatabaseObjectType.VIEW, viewAct);


      ScriptDB2ProcedureAction procAct =
         new ScriptDB2ProcedureAction(getApplication(),
                                      _resources,
                                      session);

      otApi.addToPopup(DatabaseObjectType.PROCEDURE, procAct);

      // ...
```

Listato 2 ScriptDB2ViewAction:

```
public class ScriptDB2ViewAction extends SQuirreLAction
{
   private ISession _session;


   public ScriptDB2ViewAction(IApplication app,
                              Resources rsrc,
                              ISession session)
   {
      super(app, rsrc);
      _session = session;
   }

   public void actionPerformed(ActionEvent evt)
   {
      try
      {
         Statement stat =
            _session.getSQLConnection().createStatement();

         SessionInternalFrame sessMainFrm =
            _session.getSessionInternalFrame();

         IDatabaseObjectInfo[] dbObjs =
            sessMainFrm.getObjectTreeAPI().
               getSelectedDatabaseObjects();


         StringBuffer script = new StringBuffer();
         for (int i = 0; i < dbObjs.length; i++)
         {
            ITableInfo ti = (ITableInfo) dbObjs[i];
```

```
            ////////////////////////////////////////////////////
            // IBM DB 2 specific code to read view definitions.
            String sql =
               "SELECT TEXT " +
               "FROM SYSIBM.SYSVIEWS " +
               "WHERE NAME = '" + ti.getSimpleName() + "'";

            ResultSet res = stat.executeQuery(sql);
            res.next();

            script.append(res.getString("TEXT"));
            script.append(getStatementSeparator());
            res.close();
            //
            ////////////////////////////////////////////////////
         }

         stat.close();

         sessMainFrm.getSQLPanelAPI().
            appendSQLScript(script.toString());

         sessMainFrm.getSessionPanel().
            selectMainTab(ISession.IMainPanelTabIndexes.SQL_TAB);
      }
      catch (Exception e)
      {
         throw new RuntimeException(e);
      }
   }

   /**
    * Returns the user defined statement separator with
    * suitable line feeds.
    */
   private String getStatementSeparator()
   {
      String statementSeparator =
        _session.getQueryTokenizer().getSQLStatementSeparator();

      if (1 < statementSeparator.length())
         statementSeparator = "\n" + statementSeparator + "\n";
      else
         statementSeparator += "\n";

      return statementSeparator;
   }
}
```

Conclusioni

SQuirreL fornisce un'unica interfaccia semplice ed uniforme che opera verso tutti i database relazionali. L'interfaccia fornisce funzionalita' analoghe ai piu' moderni IDE e rende semplice accedere e modificare dati, comprendere la struttura delle basi dati, lavorare ed imparare meglio l'SQL. SQuirreL sfrutta le caratteristiche che tutti i database hanno in comune gestendo le differenze con un'architettura

a plugin. Creando plugin possono essere inserite anche nuove funzionalita'.

SQuirreL aiuta chi deve operare tutti i giorni in ambienti differenti ad essere piu' produttivo. Gli utenti finali di SQuirreL sono la miglior fonte di idee per nuovi sviluppi. Chiunque abbia un suggerimento puo' contattarci all'indirizzo
squirrel-sql-users@lists.sourceforge.net.

Infine una parola per i programmatori. I programmatori di SQuirreL sono spinti sia dal piacere di programmare che dal desiderio di aggiungere funzioni a loro utili. E' bello esplorare il mondo dei database relazionali utilizzando Java ed e' una bella soddisfazione vedere la propria idea realizzata e poi utilizzata da utenti in tutto il mondo. Lavorare sul codice di SQuirreL da' una grande opportunita' per imparare a come utilizzare Java per costruire applicazioni enormemente flessibili. Unisciti a noi: ci puoi trovare all'indirizzo
squirrel-sql-develop@lists.sourceforge.net.

Traduzione italiana a cura di mail@meo.bogliolo.name