Mini SQL Query Quick Start

This document is a quick overview of how to use **Mini SQL Query**.

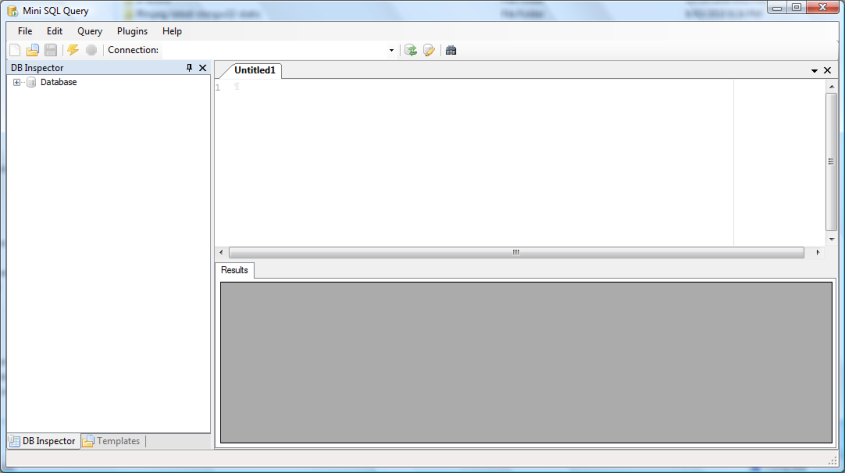
**Mini SQL Query** from **PK Software** is a minimalist SQL query tool for multiple providers (MSSQL, Oracle, OLEDB, MS Access, SQLite etc). The goal of the Mini SQL Query tool is to allow a developer or trouble-shooter to quickly diagnose issues or make changes to a database using a tool with a small footprint, is portable, fast, flexible and easy to use.

**Mini SQL Query** is "deliberately minimalist". Software too often becomes bloated and less usable while trying to satisfy the 99% of what every user wants. **Mini SQL Query** aims to satisfy the most common tasks what the average user seeks to achieve, in doing so it keeps the size and complexity of the application to a minimum.

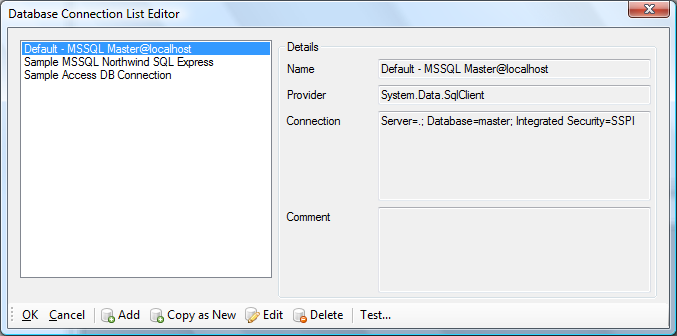
The original application was developed almost entirely using Microsoft's Visual Studio C# Express IDE.

# The First Run

The very first time you run **Mini SQL Query**, the application will look similar to the image below:



The first thing you need to do is configure some sort of connection. From the menu, select **Edit  Edit connection strings**.



There are some default connections present. Each has a Name, a Provider and a Connection with optional comments. Selecting one of the items in the left list will display the details on the right.

From this point *add*, *modify* or *copy* a connection.

## Adding a MSSQL Connection

Click **Add**.

Select a **Provider**, in this example I am leaving it as “System.Data.SqlClient”.

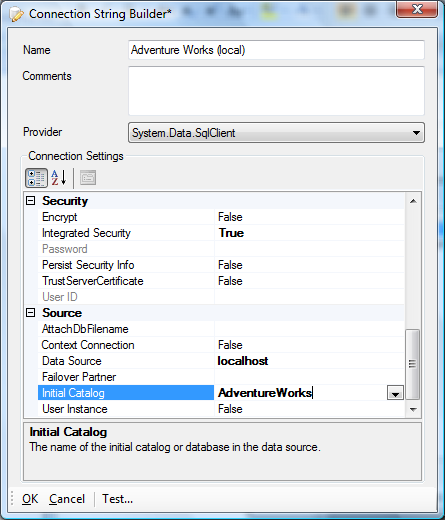
Enter a **Name** for the connection, e.g. “Adventure Works (local)”.

Set the **Connection Strings** “Integrated Security” value to “True”.

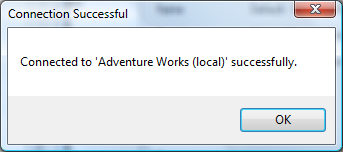
Set the **Connection Strings** “Data source” value to “localhost” (or “localhost\SQLEXPRESS” as required.)

Set the **Connection Strings** “Initial Catalog” either by using the dropdown or typing name straight in.

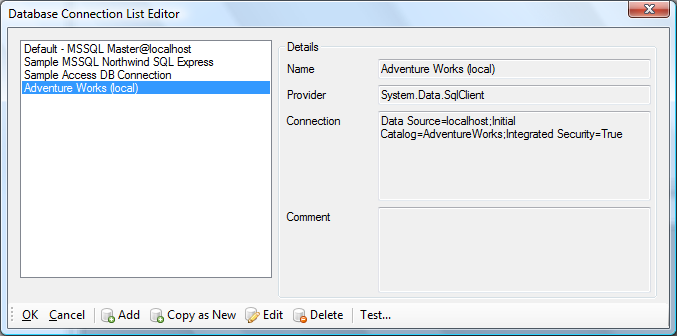
**Note** – The dropdown will only work if the connection details are sufficient to query the datasource at the time, e.g. *Integrated Security=True* will typically suffice depending on your environment.



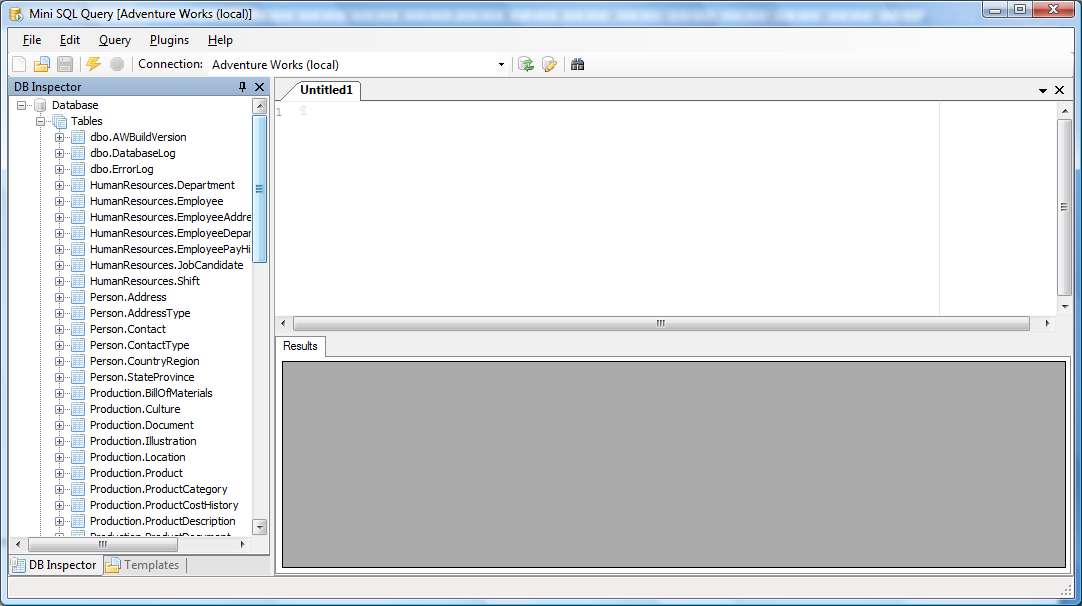
You can use the “Test...” button to confirm the connectivity.



Press **OK** and you will return to the “Database Connection List Editor” and you will see the new item in the list. Press **OK** here to return to **Mini SQL Query**.



Next select the new connection definition from the dropdown list on the toolbar. The application will pause while it loads the database schema information. Now you can explore the schema using the tree of database objects on the left. Keep in mind the “mini” in Mini SQL Query, the list of objects is by no means exhaustive! Currently the essentials such as tables, views and their associated column information are loaded. Depending on the provider foreign key information is also loaded.



**Note – Foreign Keys**The core schema engine is generic for all ADO.NET providers. The only thing not discoverable by the generic engine is the foreign key relationships. To retrieve foreign key information from a database, a provider specific implementation is required; currently there are two, Microsoft SQL Server and Microsoft’s SQL Server Compact Edition. These are typically driven by demand. Patches accepted!

**Note – Tables and Views Only**Currently only table and view information is retrieved from the databases schema. Again, this is driven by demand. The number of times access to stored procedures has been too low to warrant implementation. Patches accepted!

## Basic Usage

Below is a list of typical usage scenarios with **Mini SQL Query**.

One of the core features **Mini SQL Query** tries to achieve is *intelligent SQL code generation* where primary key, read-only or timestamp type columns are observed and not just sprayed into generated code making the output not so useful.

### Some Shortcuts

**Mini SQL Query** uses the typical editor key shortcuts for cut, copy, paste etc.

|  |  |
| --- | --- |
| **Action** | **Key Combination** |
| New window | Control+N |
| New File (then select type, txt, sql etc) | Control+Alt+N |
| Open | Control+O |
| Save | Control+S |
| Go to line | Control+G |
| Execute | F5 |
| View table | Control+T |
| Cut | Control+X |
| Copy | Control+C |
| Paste | Control+V |
| Comment block (language specific) | Control+/ |

### Context Menu Commands

Following is a brief explanation of the commands available on the context menu of the DB Inspector.

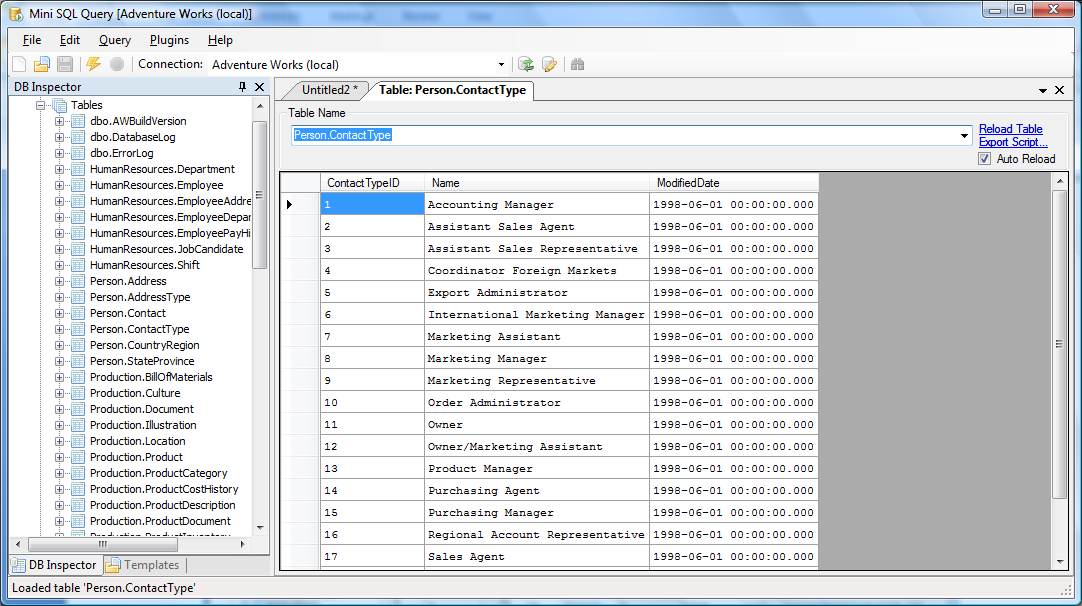
#### View Table Data

To view the contents of a table, either:

* Right-click the name of the table in the **DB Inspector** and select **View table data**
* From the **Plugins** menu, select **View table...** (Ctrl+T) and then choose a table name from the dropdown list or type it in.

Use **F5** to refresh or click the “Reload Table” link button.

Below is an example of data loaded by the tool. The date format can be modified via the **Edit  Options menu** item.



**NOTE – Exporting Data**An interesting addition here is the “Export Script...” link. It will convert the contents of the window into insert statements. By default it will observe identity or timestamp columns for example and not add those columns to the insert statements. If you want these values output the option can be changed. This can be very useful with setting up test data for example. With respect to MSSQL the “SET IDENTITY\_INSERT <tablename> ON” command can be used to insert the data.

#### Generate Select Statement

Make sure you have an active edit window in focus (Control+N) and right click a table and select “Generate Select Statement”. SQL code similar to below will be generated.

SELECT

JobCandidateID,

EmployeeID,

Resume,

ModifiedDate

FROM HumanResources.JobCandidate

##### Generate Select COUNT(\*) Statement

No prize for guessing what this does. Make sure you have an active edit window in focus (Control+N) and right click a table and select “Generate Select COUNT(\*) Statement”. SQL code similar to below will be generated.

SELECT COUNT(\*) FROM HumanResources.EmployeeAddress

##### Generate Insert Statement

Make sure you have an active edit window in focus (Control+N) and right click a table and select “Generate Insert Statement”. SQL code similar to below will be generated.

When an insert statement is generated the tables’ schema is used to ignore columns that are “read-only”. Examples are identity or timestamp columns. Also note the default values and comments to assist filling out the statement. The default for a GUID column is an empty GUID, if you need a new one generated use the “Insert GUID” menu item from the Plugins menu. Dates are a bit of a can of worms so I opted for a question mark (sorry!)

INSERT INTO HumanResources.Employee

(NationalIDNumber,

ContactID,

LoginID,

ManagerID,

Title,

BirthDate,

MaritalStatus,

Gender,

HireDate,

SalariedFlag,

VacationHours,

SickLeaveHours,

CurrentFlag,

rowguid,

ModifiedDate)

VALUES

(N'' /\*NationalIDNumber,nvarchar(15)\*/,

0 /\*ContactID,int\*/,

N'' /\*LoginID,nvarchar(256)\*/,

null /\*ManagerID,int\*/,

N'' /\*Title,nvarchar(50)\*/,

'?' /\*BirthDate,datetime\*/,

N'' /\*MaritalStatus,nchar(1)\*/,

N'' /\*Gender,nchar(1)\*/,

'?' /\*HireDate,datetime\*/,

0 /\*SalariedFlag,bit\*/,

0 /\*VacationHours,smallint\*/,

0 /\*SickLeaveHours,smallint\*/,

0 /\*CurrentFlag,bit\*/,

'00000000-0000-0000-0000-000000000000' /\*rowguid,uniqueidentifier\*/,

'?' /\*ModifiedDate,datetime\*/)

##### Generate Update Statement

Make sure you have an active edit window in focus (Control+N) and right click a table and select “Generate Update Statement”. SQL code similar to below will be generated.

As with the insert statement generation, the tables’ schema is used to ignore columns that are “read-only”. Also, the primary key columns go into the where clause with comments to assist.

UPDATE HumanResources.EmployeeDepartmentHistory

SET

EndDate = null,

ModifiedDate = '?'

WHERE

EmployeeID = /\*value:EmployeeID,int\*/ AND

DepartmentID = /\*value:DepartmentID,smallint\*/ AND

ShiftID = /\*value:ShiftID,tinyint\*/ AND

StartDate = /\*value:StartDate,datetime\*/

##### Generate Delete Statement

Make sure you have an active edit window in focus (Control+N) and right click a table and select “Generate Delete Statement”. SQL code similar to below will be generated.

As with the update statement generation, the tables’ schema is used to create a where clause with the primary key columns.

DELETE FROM

HumanResources.Department

WHERE

DepartmentID = /\*value:DepartmentID\*/

##### Copy Table Name

This command copies the fully qualified table name to the windows clipboard.

##### Truncate Table

This command deletes all the rows from the selected table. The truncate statement is not actually used currently due to the requirements (no foreign keys) and cross database support so in essence the resulting command is just “DELETE FROM <table>”. Patches welcome!

The truncate table command also sends an application level message so that if you are viewing a table with the view data command, the contents of that window is refreshed.

## TODO...

This is a starer document but there is much I could add. One of the bigger areas that is very useful is the template engine that uses NVelocity.

# Feedback

Feedback is always welcome. Suggestions, code snippets, patches and vague ideas are also welcome.

# Thank You

**Mini SQL Query** is written by Paul Kohler (<http://pksoftware.net/>; “[paul (at) pksoftware (dot) net](mailto:paul@pksoftware.net)”)

Also, a special thanks to **Kent Carlsle** for his contributions “[kdcarlisle (at) gmail (dot) com](mailto:kdcarlisle@gmail.com).”

**Mini SQL Query** relies on several other 'free/open source' products to pull together a simple but useful SQL working environment. In no particular order, thank you...

* **Mark James**, for the **Silk icon set** (<http://www.famfamfam.com/>)
* **ic#code** for the **SharpDevelopSharpDevelop** project where I get the ICSharpCode.TextEditor from that removes the need for using an plain old textbox (<http://www.icsharpcode.net/OpenSource/SD/>
* **Weifen Luo** for his extremely easy to use **WinForms** docking suite (<http://sourceforge.net/projects/dockpanelsuite/>)
* To the **Castle Project** (<http://www.castleproject.org/>) for the **NVelocity** text template engine port.
* To the **Ninject** (http://www.ninject.org/) for the dependency injection container