



Table of Contents

GETTING STARTED	2
CONNECTION SETTINGS	4
<i>General Settings for MySQL</i>	6
<i>General Settings for Oracle</i>	7
Basic Connection General Settings	8
TNS Connection General Settings	9
<i>General Settings for PostgreSQL</i>	10
<i>General Settings for SQLite</i>	12
<i>General Settings for SQL Server</i>	13
<i>SSH Settings (Available only for MySQL, Oracle, PostgreSQL and SQL Server and supports SSH2 Protocol only)</i>	15
Benefit of SSH Tunneling	16
Password Authentication	17
Public Key Authentication	19
<i>HTTP Settings (Available only for MySQL, PostgreSQL and SQLite)</i>	21
<i>SSL Settings (Available only for MySQL and PostgreSQL)</i>	22
Installation of OpenSSL and MySQL/PostgreSQL	23
Setting up SSL Certificate for MySQL/PostgreSQL	24
Setting up Client Certificate for Navicat	27
<i>Advanced Settings</i>	29



Getting Started

To start viewing reports in Navicat Report Viewer, you should first establish a connection or several connections using the Connection Windows. If you are new to the MySQL/Oracle/PostgreSQL/SQLite/SQL Server or 'Net in general' and are not quite sure how things work, you may want to look at:

- [MySQL User Manual](#)
- [Oracle Database Documentation](#)
- [PostgreSQL User Manual](#)
- [SQLite User Manual](#)
- [SQL Server MSDN Library](#)

Click  or choose File ->  **New Connection** to set up the Connection Properties.

- [Connection Settings](#).

After the connections being established, you need to place your report files (.rtm) to the [Report Location](#) before viewing and printing your reports.

Navicat Report Viewer Explorer!

The Navicat Report Viewer window includes an object pane that displays the opened reports. On the left side is a navigation pane that helps you browse through the current reports. Toolbars at the top of window provide other controls that you can use to manipulate your report.

To view or hide Navigation Pane, click the **Red Indicator** below.



Navicat Report Viewer

File View Window Help

Connection Print Whole Page Page Width 100% Refresh Search

sales_statistics x OrderSummary x mailing_label x CustomerDetail x

Sales Statistics

Cur.No.	Company	Order No.	Qty	Total
1281	Sight Drive	1002	4	\$1,250.00
<hr/>				
1286	Tom Sawyer Driving Centre	1005	21	\$18,410.00
<hr/>				
1280	Blue Jack Aqua Center	1006	25	\$50,026.50
<hr/>				
1284	V101 Drivers Club	1007	20	\$6,500.00
<hr/>				
1513	Panbaltique Aquatics	1008	11	\$1,448.50
		1009	7	\$1,197.00
<hr/>				
2150	Darry Jones' Locker	1004	33	\$10,838.00
<hr/>				

11/11/2010 4:09:13 PM Page 1

Zoom Percentage 55



12 Reports Remote MySQL Connection User: root



Connection Settings

Navicat Report Viewer assembles utilitarian tools to view and print your reports. To start managing your reports in Navicat Report Viewer, the first thing you require to do is to establish your server connection.

Create Connection

Navicat Report Viewer provides three typical approaches to establish your connection, click  or choose File ->  **New Connection** to start the setup.


- [General Settings for MySQL](#)
- [General Settings for Oracle](#)
- [General Settings for PostgreSQL](#)
- [General Settings for SQLite](#)
- [General Settings for SQL Server](#)
- [SSH Settings](#) (Available only for MySQL, Oracle, PostgreSQL and SQL Server)
- [HTTP Settings](#) (Available only for MySQL, PostgreSQL and SQLite)

Note: For MySQL or PostgreSQL server, a commonly-used protocol - **Secure Sockets Layer (SSL)** is employed for managing the security of a message transmission on the Internet (see [SSL Settings](#) for details).

Note: Navicat Report Viewer authorizes you to make connection to remote server running on different platform, i.e. Windows, Mac, Linux and UNIX.

Delete Connection

To delete a connection

- Right-click the connection in the navigation pane and choose  **Delete Connection**.
- Confirm deleting in the dialog window.

Open Connection


To open a connection

- Double-click the connection to open in the navigation pane.




Close Connection

To close a connection

- Right-click the connection in the navigation pane and choose  **Close Connection**.

Edit Connection

To edit a connection information

- Close the connection if it is being opened.
- Right-click the connection and choose  **Connection Properties**.

Achieve Connection Information

To achieve a connection information

- Open the connection in the navigation pane.
- Right-click the opened connection and choose **Connection Information**.



General Settings for MySQL

The following instruction guides you through the process of creating a new connection. To successfully establish a new connection to local/remote MySQL Server - no matter via SSL, SSH or HTTP, set the connection properties in the corresponding boxes: Connection name, Host name, Port number, User name, and Password.

By default, MySQL gives "root" as username and leave the password field blank.

Connection Name

A friendly name to best describe your connection.

Host Name/IP Address

A host name where the database is situated or the IP address of the server.

Port

A TCP/IP port for connecting to the database server.

User Name

User name for connecting to the database server.

Password

Password for connecting to the server.

If your Internet Service Provider (ISP) does not provide direct access to its server, Secure Tunneling Protocol [SSH](#) / [HTTP](#) is another solution.

See also:

[Advanced Settings](#)

Related topics:

[SSL](#), [SSH](#), [HTTP](#)



General Settings for Oracle

The following instruction guides you through the process of creating a new connection for server. To successfully establish a new connection to local/remote Oracle Server - no matter via SSH, set the connection properties in the corresponding boxes: Connection name, Host name, Port number, User name, and Password.

By default, Oracle created a number of user accounts upon installation. Administrative accounts: SYS, SYSTEM, SYSMAN, and DBSNMP. Sample schema accounts: SCOTT, HR, OE, OC, PM, IX and SH.

Navicat supports 2 types of Oracle Server connection:

- [Basic Connection](#)
- [TNS Connection](#)

If your Internet Service Provider (ISP) does not provide direct access to its server, Secure Tunneling Protocol [SSH](#) is another solution.

Related topics:

[SSH](#)



Basic Connection General Settings

Connection Name

A friendly name to best describe your connection.

Connection Type

Connection type for connecting to the server: **Basic** or [TNS](#).

Basic

In Basic mode, Navicat Report Viewer connects to Oracle through the Oracle Call Interface (OCI). OCI is an application programming interface that allows an application developer to use a third-generation language's native procedure or function calls to access the Oracle database server and control all phases of SQL statement execution. OCI is a library of standard database access and retrieval functions in the form of a dynamic-link library.

Host Name/IP Address

A host name where the database is situated or the IP address of the server.

Port

A TCP/IP port for connecting to the database server.

Service Name/SID

Set the Service Name/SID which the user connects when making connection. Select the corresponding radio button.

User Name

User name for connecting to the database server.

Password

Password for connecting to the server.

See also:

[Advanced Settings](#)

Related topics:

[TNS](#), [SSH](#)



TNS Connection General Settings

Connection Name

A friendly name to best describe your connection.

Connection Type

Connection type for connecting to the server: [Basic](#) or **TNS**.

TNS

In TNS mode, Navicat Report Viewer connects to Oracle server using an alias entry from a tnsnames.ora file through the Oracle Call Interface (OCI). OCI is an application programming interface that allows an application developer to use a third-generation language's native procedure or function calls to access the Oracle database server and control all phases of SQL statement execution. OCI is a library of standard database access and retrieval functions in the form of a dynamic-link library.

Net Service Name

The net service name.

User Name

User name for connecting to the database server.

Password

Password for connecting to the server.

See also:

[Advanced Settings](#)

Related topics:

[Basic](#), [SSH](#)



General Settings for PostgreSQL

The following instruction guides you through the process of creating a new connection. To successfully establish a new connection to local/remote PostgreSQL Server - no matter via SSH, HTTP or SSL, set the connection properties in the corresponding boxes: Connection name, Host name, Port number, Initial Database, User name, and Password.

By default, PostgreSQL gives "postgres" as username and leave the password field blank.

Connection Name

A friendly name to best describe your connection.

Host Name/IP Address

A host name where the database is situated or the IP address of the server.

Port

A TCP/IP port for connecting to the database server.

Initial Database

The initial database to which user connects when making connection.

User Name

User name for connecting to the database server.

Password

Password for connecting to the server.

If your Internet Service Provider (ISP) does not provide direct access to its server, Secure Tunneling Protocol [SSH](#) / [HTTP](#) is another solution.

Note: For security reasons native remote direct connections to the PostgreSQL server are disabled. Therefore, you may not be able to use Navicat Premium or other similar PostgreSQL admin applications running on your computer to connect to the remote server. For more details, refer to next paragraph on Server Administration.

For Server Administration:

By default, PostgreSQL only allows connections from the local machine using TCP/IP connections. Other machines will not be able to connect unless you modify *listen_addresses* in the *postgresql.conf* file, enable host-based authentication by modifying the *\$PGDATA/pg_hba.conf* file, and restart the server. For more information: [Client Authentication](#)



See also:

[Advanced Settings](#)

Related topics:

[SSL](#), [SSH](#), [HTTP](#)



General Settings for SQLite

The following instruction guides you through the process of creating a new connection. To successfully establish a new connection to local/remote SQLite Server - no matter via HTTP, set the connection properties in the corresponding boxes: Connection name, Type and Database Name.

Connection Name

A friendly name to best describe your connection.

Database File

Specify the initial database file. If the [HTTP Tunnel](#) is enabled, you need to enter an absolute file path of the database file in your webserver.

See also:

[Advanced Settings](#)

Related topics:

[HTTP](#)



General Settings for SQL Server

The following instruction guides you through the process of creating a new connection. To successfully establish a new connection to local/remote SQL Server - no matter via SSH, set the connection properties in the corresponding boxes: Connection name, Host name, and Authentication Type.

Connection Name

A friendly name to best describe your connection.

Host Name/IP Address

A host name where the database is situated or the IP address of the server.

Authentication

SQL Server uses two ways to validate connections to SQL Server databases: SQL Server Authentication and Windows Authentication.

SQL Server Authentication

SQL Server Authentication uses login records to validate the connection. Users must provide their login username and password every time that they connect.

User Name

User name for connecting to the database server.

Password

Password for connecting to the server.

Windows Authentication

When a user connects through a Windows user account, SQL Server validates the account name and password using the Windows principal token in the operating system. This means that the user identity is confirmed by Windows. SQL Server does not ask for the password, and does not perform the identity validation.

Initial Database

The initial database to which user connects when making connection.

If your Internet Service Provider (ISP) does not provide direct access to its server, Secure Tunneling Protocol [SSH](#) is another solution.

See also:

[Advanced Settings](#)



Related topics:

[SSH](#)



SSH Settings (Available only for MySQL, Oracle, PostgreSQL and SQL Server and supports SSH2 Protocol only)

Secure SHell (SSH) is a program to log in into another computer over a network, execute commands on a remote server, and move files from one machine to another. It provides strong authentication and secure encrypted communications between two hosts, known as **SSH Port Forwarding (Tunneling)**, over an insecure network. Typically, it is employed as an encrypted version of Telnet.

In a Telnet session, all communications, including username and password, are transmitted in plain-text, allowing anyone to listen-in on your session and steal passwords and other information. Such sessions are also susceptible to session hijacking, where a malicious user takes over your session once you have authenticated. SSH serves to prevent such vulnerabilities and allows you to access a remote server's shell without compromising security.

- [Benefit of SSH Tunneling.](#)

To ensure that the incoming connection request is from you, SSH can use a password, or public/private key pair (also called public key) authentication mechanism.

- [Password Authentication.](#)
- [Public Key Authentication.](#)

Note: Please make sure that the parameter - "AllowTcpForwarding" in the Linux Server must be set to value "yes", otherwise, the SSH port forwarding will be disabled. To look for the path: `/etc/ssh/sshd_config` .By default, the SSH port forwarding should be enabled. Please double check the value settings.

****** Even the server support SSH tunnel, however, if the port forwarding being disabled, Navicat cannot connect via SSH Port 22.

See also:

[Advanced Settings](#)

Related topic:

[SSL](#)



Benefit of SSH Tunneling

SSH has a wonderful feature called SSH Port Forwarding, sometimes called SSH Tunneling, which allows you to establish a secure SSH session and then tunnel arbitrary TCP connections through it. Tunnels can be created at any time, with almost no effort and no programming, which makes them very appealing. SSH Port Forwarding can be used for secure communications in a myriad of different ways.

Many Hosting Companies, that provide database server hosting, will block access to the server from outside the hosting company's network, and only grant access to users connecting from localhost.

There are several benefits to using SSH:

- Connection to a server from behind a firewall when the server port is blocked.
- Automatic authentication of users, no passwords sent in plain text to prevent the stealing of passwords.
- Multiple strong authentication methods that prevent such security threats as spoofing identity.
- Encryption and compression of data for security and speed.
- Secure file transfer.

Related topics:



[Password Authentication](#), [Public Key Authentication](#)



Password Authentication

Using this mode, SSH is almost identical to the program telnet. When you make a connection, you are asked for your password. You type it in and you are either logged in or denied. Your password is first encrypted and then sent over the network and then decrypted at the remote host. This is the mode that most users will be encouraged to use, as it requires no additional setup or configuration.

The following instruction guides you through the process of configuring a SSH connection using Password Authentication. To successfully establish a SSH connection, set the SSH connection properties in the corresponding boxes: Host name/IP address, Port number, User name, Authentication Method and Password.

1. Click  or choose File ->  **New Connection** to set up the Connection Properties.
2. Select the SSH tab and enable **Use SSH Tunnel**.
3. Fill in the required information:

Host Name/IP Address

A host where SSH server is activated.

Port

A port where SSH server is activated, by default it is 22.

User Name

A user on Linux machine. (It is a Linux user. It is not a user of database server.)

Authentication Method

Choose between Password Authentication and [Public Key Authentication](#)

Password

It is a Linux user password.



General | Advanced | SSL | **SSH** | HTTP

Use SSH Tunnel

Host Name/IP Address: SSH_server_IP_address

Port: 22

User Name: SSH_login_name

Authentication Method: Password

Password: ●●●●●●●●

Save Password

4. Navicat host name at the General Settings page ([MySQL](#), [Oracle](#), [PostgreSQL](#) or [SQL Server](#)) should be set relatively to the SSH server which provided by your database hosting company.

See also:

[Advanced Settings](#)

Related topics:

[Public Key Authentication](#)



Public Key Authentication

Public-key Authentication is based on the use of digital signatures and provides the best authentication security.

For Public Key Authentication to work four things are needed:

- the remote server(s) you are connecting must have your public key.
- the local client you are connecting from must have your private key.
- the remote server must be configured to allow you to login using your public key.
- the local client must be configured to use your private key while logging into remote server.

The following instruction guides you through the process of configuring a SSH connection using Public Key Authentication. To successfully establish a SSH connection , set the SSH connection properties in the corresponding boxes: Host name/IP address, Port number, User name, Authentication Method, Private Key and Passphrase.

1. Click  or choose File ->  **New Connection** to set up the Connection Properties.
2. Select the SSH tab and enable **Use SSH Tunnel**.
3. Fill in the required information:

Host Name/IP Address

A host where SSH server is activated.

Port

A port where SSH server is activated, by default it is 22.

User Name

A user on Linux machine. (It is a Linux user. It is not a user of database server.)

Authentication Method

Choose between [Password Authentication](#) and Public Key Authentication

Private Key

It is used together with your public key. The private key should be readable only by you.



Passphrase

A passphrase is exactly like a password, except that it applies to the keys you are generating and not an account. The passphrase be any length under 1024 characters.

The screenshot shows the SSH settings tab in a configuration window. The window has tabs for General, Advanced, SSL, SSH, and HTTP. The SSH tab is selected. The settings are as follows:

- Use SSH Tunnel
- Host Name/IP Address: SSH_server_IP_address
- Port: 22
- User Name: SSH_login_name
- Authentication Method: Public Key (dropdown menu)
- Private Key: c:\ssh\private_key\identify (with a browse button)
- Passphrase: [masked with dots]
- Save Passphrase

4. Navicat host name at the General Settings page ([MySQL](#), [Oracle](#), [PostgreSQL](#) or [SQL Server](#)) should be set relatively to the SSH server which provided by your database hosting company.

See also:

[Advanced Settings](#)

Related topics:

[Password Authentication](#)



HTTP Settings (Available only for MySQL, PostgreSQL and SQLite)

HTTP Tunneling is a method for connecting to a server that uses the same protocol (http://) and the same port (port 80) as a webserver does. It is used while your ISPs do not allow direct connections, but allows establishing HTTP connections.

Steps of setting up HTTP Connection:



1. Uploading the Tunneling Script

To use this connection method, first thing you need to do is to upload the tunneling script to the webserver where your server is located.

Note: `ntunnel_mysql.php`, `ntunnel_pgsql.php` or `ntunnel_sqlite.php` is available in the Navicat installation folder.

2. Setting up HTTP Tunnel

The following instruction guides you through the process of configuring a HTTP connection.

1. Click  or choose File ->  **New Connection** to set up the Connection Properties.
2. Select the HTTP tab and enable **Use HTTP Tunnel**.
3. Enter URL of the tunneling script, e.g.
`http://www.navicat.com/ntunnel_mysql.php` .
4. If your server installed ModSecurity, you can check the **Encode outgoing query with base64** option.
5. If the tunneling script is hosted in a password protected server or you have to access internet over a proxy server, you can provide the required authentication details in **Authentication** or **Proxy** tab..
6. Navicat host name at the General Settings page should be set relatively to the HTTP server which provided by your database hosting company.

Note: HTTP Tunnel and SSH Tunnel cannot function simultaneously. The SSH Tunnel is disabled when you select the HTTP Tunnel and vice versa.



SSL Settings (Available only for MySQL and PostgreSQL)

Secure Sockets Layer(SSL) is a protocol for transmitting private documents via the Internet. To get a secure connection to work with MySQL/PostgreSQL Server, the first thing you need to do is to install OpenSSL Library and download MySQL/PostgreSQL Database Source.

Steps of setting up SSL Connection for MySQL/PostgreSQL Server and Navicat:

1. [Installation of OpenSSL and MySQL/PostgreSQL.](#)
2. [Setting up SSL Certificate for MySQL/PostgreSQL.](#)
3. [Setting up Client Certificate for Navicat.](#)

Note: Support from PostgreSQL 8.4 or later.

See also:

[Advanced Settings](#)

Related topics:

[General Settings for MySQL](#), [General Settings for PostgreSQL](#), [SSH](#)



Installation of OpenSSL and MySQL/PostgreSQL

Installing OpenSSL

1. Download OpenSSL - <http://www.openssl.org>
2. Linux command : [`zcat 0.96l.tar.gz | tar xvf -`]
3. Linux command : [`./config`]
4. Linux command : [`make`]
5. Linux command : [`make install`]

Installing MySQL

1. Download MySQL - <http://www.mysql.com>
2. Linux command : [`./configure --with -vio --with -openssl`]
3. Linux command : [`make`]
4. Linux command : [`make install`]

Note: Please ensure if MySQL Server supports OpenSSL using query statement:
[`SHOW VARIABLES LIKE 'have_openssl';`] - Returns value = YES

Installing PostgreSQL

1. Download PostgreSQL - <http://www.postgresql.org>
2. Linux command : [`./configure --with-openssl`]
3. Linux command : [`gmake`]
4. Linux command : [`gmake install`]

Note: Please ensure if PostgreSQL Server supports OpenSSL using query statement:
[`SHOW ssl;`] - Returns value = ON

See also:

Step 2: [Setting up SSL Certificate for MySQL/PostgreSQL](#)



Setting up SSL Certificate for MySQL/PostgreSQL

To create server/client side Certificate, login to the Linux Server as root and employ the Shell Command below:

MySQL

1. `DIR=`pwd`/openssl`
2. `PRIV=$DIR/private`
3. `mkdir $DIR $PRIV $DIR/newcerts`
4. `cp /usr/share/ssl/openssl.cnf $DIR`
5. `replace ./demoCA $DIR -- $DIR/openssl.cnf`
6. Generation of Certificate Authority(CA)

```
/usr/local/ssl/bin/openssl req -new -x509 -keyout $PRIV/cakey.pem -out  
$DIR/cacert.pem -config $DIR/openssl.cnf
```

Note: If "PEM" is required, please enter different "PEM pass" via steps below.

7. Create server request and key

```
/usr/local/ssl/bin/openssl req -new -keyout $DIR/server-key.pem -out  
$DIR/server-req.pem -days 3600 -config $DIR/openssl.cnf
```

8. Remove the passphrase from the key (optional)

```
/usr/local/ssl/bin/openssl rsa -in $DIR/server-key.pem -out $DIR/server-key.pem
```

9. Sign server cert

```
/usr/local/ssl/bin/openssl ca -policy policy_anything -out $DIR/server-cert.pem  
-config $DIR/openssl.cnf -infiles $DIR/server-req.pem
```

10. Create client request and key

```
/usr/local/ssl/bin/openssl req -new -keyout $DIR/client-key.pem -out  
$DIR/client-req.pem -days 3600 -config $DIR/openssl.cnf
```



11. Remove a passphrase from the key (optional)

```
/usr/local/ssl/bin/openssl rsa -in $DIR/client-key.pem -out $DIR/client-key.pem
```

12. Sign client cert

```
/usr/local/ssl/bin/openssl ca -policy policy_anything -out $DIR/client-cert.pem  
-config $DIR/openssl.cnf -infiles $DIR/client-req.pem
```

13. Create a **my.cnf** file for testing the Certificates. Store it either in `/etc` or MySQL data directory (typically `/usr/local/var` for source installation)

my.cnf example content:

```
[client]  
ssl-ca=$DIR/cacert.pem  
ssl-cert=$DIR/client-cert.pem  
ssl-key=$DIR/client-key.pem  
[mysqld]  
ssl-ca=$DIR/cacert.pem  
ssl-cert=$DIR/server-cert.pem  
ssl-key=$DIR/server-key.pem
```

14. To start MySQL daemon:

```
/usr/local/libexec/mysqld -u mysql &
```

or

```
/usr/local/sbin/mysqld -u &
```

PostgreSQL

1. To create a quick self-signed certificate for the server, use the following OpenSSL command:

```
openssl req -new -text -out server.reqm
```

2. Fill out the information that openssl asks for. Make sure you enter the local host name as "Common Name"; the challenge password can be left blank. The program will generate a key that is passphrase protected; it will not accept a passphrase that is less



than four characters long. To remove the passphrase (as you must if you want automatic start-up of the server), run the commands:

```
openssl rsa -in privkey.pem -out server.key  
rm privkey.pem
```

3. Enter the old passphrase to unlock the existing key. Now do:

```
openssl req -x509 -in server.req -text -key server.key -out server.crt
```

4. to turn the certificate into a self-signed certificate and to copy the key and certificate to where the server will look for them. Finally do:

```
chmod og-rwx server.key
```



See also:

Step 3: [Setting up Client Certificate for Navicat](#)


Setting up Client Certificate for Navicat

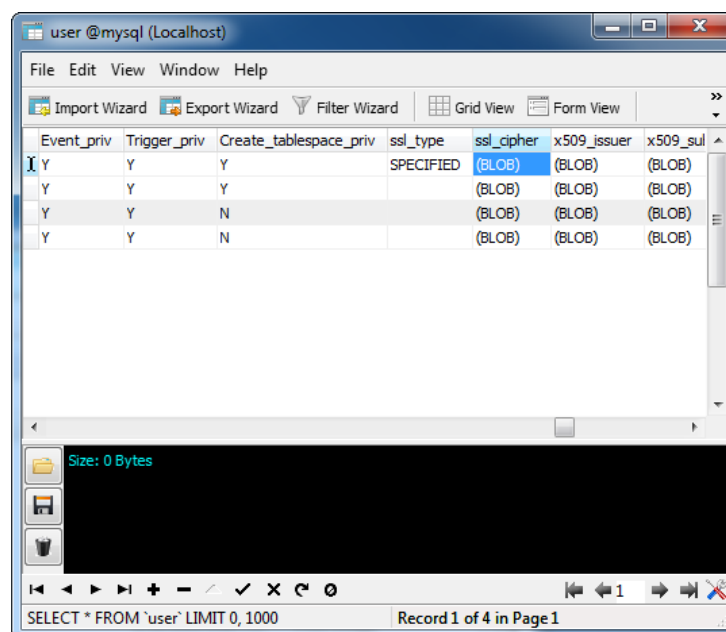
The following instruction guides you through the process of configuring a connection between Navicat and MySQL/PostgreSQL Server using SSL. To successfully establish a SSL connection, please complete [Step 1: Installation of OpenSSL and MySQL/PostgreSQL](#) and [Step 2: Setting up SSL Certificate for MySQL/PostgreSQL](#), and set the connection properties in the corresponding boxes.

MySQL

1. Click  or choose File ->  **New Connection** to set up the Connection Properties.
2. Select the SSL tab and enable **Use SSL**.
3. To provide authentication details, fill in the required information:



Client Key, Client Certificate and **CA Certificate** are usually stored in your Server - `/usr/local/openssl`. Please copy them from the remote server to local computer. **Specified Cipher** (optional) is only required while **ssl_type** field has been set to "**SPECIFIED**" - [ssl_type can be found in a system database called "mysql" -> table called "user"]. Example of Specified Cipher is "EDH-RSA-DES-CBC3-SHA" which can be filled in either through the Connection Properties shown above or the "mysql" database -> "user" table -> "ssl_cipher" blob field shown below.

Note: You are allowed to store your Specified Cipher into a text file in order to load  into the "ssl_cipher" blob field.





PostgreSQL

1. Click  or choose File ->  **New Connection** to set up the Connection Properties.
2. Select the SSL tab and enable **Use SSL**.
3. Select the **SSL Mode**.
require - only try an SSL connection.
verify-ca - only try an SSL connection, and verify that the server certificate is issued by a trusted CA.
verify-full - only try an SSL connection, verify that the server certificate is issued by a trusted CA and that the server hostname matches that in the certificate.
4. To provide authentication details, enable **Use Authentication** and fill in the required information:

Client Key, **Client Certificate** and **CA Certificate** are usually stored in your Server - `/usr/local/openssl`. Please copy them from the remote server to local computer.

Certificate Revocation List specifies the file path of the SSL certificate revocation list (CRL).

For PostgreSQL server, OpenSSL supports a wide range of ciphers and authentication algorithms, of varying strength. While a list of ciphers can be specified in the OpenSSL configuration file, you can specify ciphers specifically for use by the database server by modifying `ssl_ciphers` in `postgresql.conf`.

See also:

[Advanced Settings](#)



Advanced Settings

Customize connection options according to your needs. The detailed description is given below:

Report Location

When a new connection being established, Navicat Report Viewer will create a subfolder under the Report Location. You should place all your report files (.rtm) within this subfolder.

Hint: You are allowed to change any locations as you like.

MySQL

Encoding

Choose a codepage to communicate with MySQL Server while MySQL character set not being employed.

Keepalive Interval (sec)

This option allows you to keep the connection with the server alive by pinging it. You can set the period between pings in the edit field.

Use MySQL character set

This option should be enabled if employing MySQL 4.1 or above.

Use Compression

This option allows you to use compression protocol. It is used if both client and server support zlib compression, and the client requests compression.

Auto Connect

With this option on, Navicat Report Viewer automatically open connection with the registered database at application startup.

Use Named Pipe, Socket

With this option on, Navicat Report Viewer uses socket file for localhost connection.



Oracle

Role

Indicate that the database user is connecting with either the Default, SYSOPER or SYSDBA system privilege.

OS Authentication

With this option on, Oracle Database uses Windows user login credentials to authenticate database users.

Auto Connect

With this option on, Navicat Report Viewer automatically opens connection with the registered database at application startup.

PostgreSQL

Keepalive Interval (sec)

This option allows you to keep the connection with the server alive by pinging it. You can set the period between pings in the edit field

Auto Connect

With this option on, Navicat Report Viewer automatically opens connection with the registered database at application startup.

SQLite

Auto Connect

With this option on, Navicat Report Viewer automatically opens connection with the registered database at application startup.

Encrypted

Enable this option and provide **Password** when connecting to an encrypted SQLite database.

Attached Database

To attach or detach databases in the connection.



SQL Server

Use Encryption

This option allows you to use encryption.

Auto Connect

With this option on, Navicat Report Viewer automatically opens connection with the registered database at application startup.