

Trafodion Command Interface Guide

© Copyright 2015 Apache Software Foundation

[Legal Notice](#)

Licensed to the Apache Software Foundation (ASF) under one or more contributor license agreements. See the NOTICE file distributed with this work for additional information regarding copyright ownership. The ASF licenses this file to you under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Acknowledgements

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation. Java® is a registered trademark of Oracle and/or its affiliates.

Contents

About This Document	10
Intended Audience.....	10
Document Organization.....	10
New and Changed Information in This Edition	10
Notation Conventions.....	10
General Syntax Notation	10
Publishing History.....	12
We Encourage Your Comments.....	12
1 Introduction to the Trafodion Command Interface (TrafCI)	13
2 Installing and Configuring TrafCI.....	14
Installing TrafCI.....	14
Verifying and Setting the Java Path.....	14
Setting the PATH on Windows.....	14
Setting the PATH on Linux	15
Testing the Launch of TrafCI	16
3 Launching TrafCI	17
Launching TrafCI on a Windows Client Workstation.....	17
Creating a Shortcut to trafci.cmd	17
Launching TrafCI on a Linux Client Workstation.....	20
Setting the PATH of trafci.sh.....	20
Presetting the Optional Launch Parameters	21
Logging In to the Database Platform.....	21
Logging In Without Using Login Parameters.....	21
Using Login Parameters	22
Retrying the Login	22
Using Optional Launch Parameters	23
Running a Command When Launching TrafCI.....	24
Example of Running an SQL Statement With -q or -sql.....	24
Example of Running an Interface Command With -q or -sql.....	25
Running a Script File When Launching TrafCI.....	25
Example of a Script File.....	25
Example of Running a Script File With -s or -script	26
Launching TrafCI Without Connecting to the Database	26
Example of Launching TrafCI With -noconnect.....	26
Running TrafCI With -version.....	27
Example of Running TrafCI With -version.....	27
Running TrafCI With -help.....	27
Example of Running TrafCI With -help	27
Exiting TrafCI.....	27
4 Running Commands Interactively in TrafCI.....	28
User Interface	28
Product Banner	28
Interface Prompt	28
Breaking the Command Line	28
Case Sensitivity	29
Using the Interface Commands	29
Showing the Session Attributes.....	29
Setting and Showing the Idle Timeout Value for the Session	30
Customizing the Standard Prompt.....	30

SET PROMPT Command.....	30
SET TIME Command.....	30
Setting and Showing the SQL Terminator	30
Displaying the Elapsed Time.....	31
Setting and Showing the Current Schema.....	31
Limiting the Result Set of a Query.....	32
Displaying Executed Commands.....	32
Editing and Reexecuting a Command	33
Clearing the Interface Window.....	33
Obtaining Help.....	33
Running SQL Statements.....	33
Executing an SQL Statement.....	33
Repeating an SQL Statement.....	34
Preparing and Executing SQL Statements.....	34
Preparing an SQL Statement.....	34
Setting Parameters	35
Displaying the Parameters of the Session	36
Resetting the Parameters.....	36
Executing a Prepared SQL Statement	36
Logging Output.....	38
Starting the Logging Process	38
SPOOL ON or LOG ON Command.....	38
SPOOL log-file or LOG log-file Command.....	38
Using the CLEAR Option.....	38
Logging Concurrent the TrafCI Sessions.....	39
Stopping the Logging Process.....	39
Viewing the Contents of a Log File.....	39
5 Running Scripts in TrafCI	40
Creating a Script File.....	40
SQL Statements.....	40
Interface Commands.....	40
Comments.....	40
Section Headers.....	40
Example of a Script File	41
Running a Script File.....	41
Logging Output.....	42
Running Scripts in Parallel.....	42
6 Running TrafCI From Perl or Python	43
Setting the Login Environment Variables.....	43
Setting the Login Environment Variables on Windows.....	43
Setting Login Environment Variables on the Command Line.....	43
Setting Login Environment Variables in the System Properties	44
Setting the Login Environment Variables on Linux or UNIX	45
Setting Login Environment Variables on the Command Line.....	46
Setting Login Environment Variables in the User Profile.....	46
Perl and Python Wrapper Scripts.....	46
Launching TrafCI From the Perl or Python Command Line	46
Example of a Perl Program (sample.pl)	47
Example of a Python Program (sample.py).....	47
A Interface Commands.....	48
@ Command.....	51
Syntax.....	51
Considerations	51

Examples.....	51
/ Command.....	52
Syntax	52
Considerations.....	52
Example.....	52
ALIAS Command.....	53
Syntax	53
Considerations.....	53
Examples.....	53
CLEAR Command.....	54
Syntax	54
Considerations.....	54
Example.....	54
CONNECT Command.....	55
Syntax	55
Considerations.....	55
Examples.....	55
DELAY Command.....	57
Syntax	57
Considerations.....	57
Examples.....	57
DISCONNECT Command.....	58
Syntax	58
Considerations.....	58
Examples.....	58
ENV Command.....	59
Syntax	59
Considerations.....	59
Examples.....	60
EXIT Command.....	61
Syntax	61
Considerations.....	61
Examples.....	61
FC Command.....	62
Syntax	62
Considerations.....	62
Examples.....	62
GET STATISTICS Command.....	65
Syntax	65
Considerations.....	65
Examples.....	65
GOTO Command.....	67
Syntax	67
Considerations.....	67
Examples.....	67
HELP Command.....	68
Syntax	68
Considerations.....	68
Examples.....	68
HISTORY Command.....	69
Syntax	69
Considerations.....	69
Example.....	69
IF...THEN Command.....	70
Syntax	70

Condition Parameter	70
Action Parameter	70
SQL Terminator	70
Considerations	70
Examples	71
LABEL Command	72
Syntax	72
Considerations	72
Examples	72
LOCALHOST Command	73
Syntax	73
Considerations	73
Examples	73
LOG Command	74
Syntax	74
Considerations	74
Examples	74
OBEY Command	77
Syntax	77
Considerations	77
Examples	78
PRUN Command	80
Syntax	80
Considerations	80
Examples	81
QUIT Command	83
Syntax	83
Considerations	83
Examples	83
RECONNECT Command	84
Syntax	84
Considerations	84
Examples	84
REPEAT Command	85
Syntax	85
Considerations	85
Examples	85
RESET LASTERROR Command	87
Syntax	87
Considerations	87
Examples	87
RESET PARAM Command	88
Syntax	88
Considerations	88
Example	88
RUN Command	89
Syntax	89
Considerations	89
Example	89
SAVEHIST Command	90
Syntax	90
Considerations	90
Examples	90
SET COLSEP Command	91
Syntax	91

Considerations.....	91
Examples.....	91
SET FETCHSIZE Command.....	92
Syntax	92
Considerations.....	92
Examples.....	92
SET HISTOPT Command	93
Syntax	93
Considerations.....	93
Examples.....	93
SET IDLETIMEOUT Command	95
Syntax	95
Considerations.....	95
Examples.....	95
SET LIST_COUNT Command.....	96
Syntax	96
Considerations.....	96
Examples.....	96
SET MARKUP Command.....	98
Syntax	98
Considerations.....	98
Examples.....	98
SET PARAM Command.....	102
Syntax	102
Considerations.....	102
Examples.....	103
SET PROMPT Command	104
Syntax	104
Considerations.....	104
Examples.....	104
SET SQLPROMPT Command	106
Syntax	106
Considerations.....	106
Examples.....	106
SET SQLTERMINATOR Command.....	108
Syntax	108
Considerations.....	108
Examples.....	108
SET STATISTICS Command.....	109
Syntax	109
Considerations.....	109
Examples.....	109
SET TIME Command	110
Syntax	110
Considerations.....	110
Examples.....	110
SET TIMING Command	111
Syntax	111
Considerations.....	111
Examples.....	111
SHOW ACTIVITYCOUNT Command.....	112
Syntax	112
Examples.....	112
SHOW ALIAS Command.....	113
Syntax	113

Considerations	113
Examples	113
SHOW ALIASES Command	114
Syntax	114
Considerations	114
Examples	114
SHOW CATALOG Command	115
Syntax	115
Considerations	115
Example	115
SHOW COLSEP Command	116
Syntax	116
Considerations	116
Examples	116
SHOW ERRORCODE Command	117
Syntax	117
Examples	117
SHOW FETCHSIZE Command	118
Syntax	118
Considerations	118
Examples	118
SHOW HISTOPT Command	119
Syntax	119
Considerations	119
Examples	119
SHOW IDLETIMEOUT Command	120
Syntax	120
Considerations	120
Examples	120
SHOW LASTERROR Command	121
Syntax	121
Considerations	121
Examples	121
SHOW LIST_COUNT Command	122
Syntax	122
Considerations	122
Examples	122
SHOW MARKUP Command	123
Syntax	123
Considerations	123
Examples	123
SHOW PARAM Command	124
Syntax	124
Considerations	124
Example	124
SHOW PREPARED Command	125
Syntax	125
Considerations	125
Examples	125
SHOW RECCOUNT Command	126
Syntax	126
Considerations	126
Examples	126
SHOW REMOTEPROCESS Command	127
Syntax	127

Considerations	127
Example	127
SHOW SCHEMA Command	128
Syntax	128
Considerations	128
Example	128
SHOW SESSION Command	129
Syntax	129
Considerations	129
Examples	130
SHOW SQLPROMPT Command	131
Syntax	131
Considerations	131
Example	131
SHOW SQLTERMINATOR Command	132
Syntax	132
Considerations	132
Example	132
SHOW STATISTICS Command	133
Syntax	133
Considerations	133
Example	133
SHOW TIME Command	134
Syntax	134
Considerations	134
Example	134
SHOW TIMING Command	135
Syntax	135
Considerations	135
Example	135
SPOOL Command	136
Syntax	136
Considerations	136
Examples	136
VERSION Command	139
Syntax	139
Considerations	139
Example	139

Index	140
-------------	-----

About This Document

This guide describes how to use the Trafodion Command Interface (TrafCI) on a client workstation to connect to and query a Trafodion database. The TrafCI enables you to run SQL statements interactively or from script files.

Intended Audience

This guide is intended for database administrators and support personnel who are maintaining and monitoring a Trafodion database.

Document Organization

Chapter 1: Introduction to the Trafodion Command Interface (TrafCI)	Introduces the Trafodion Command Interface (TrafCI) and describes its capabilities.
Chapter 2: Installing and Configuring TrafCI	Describes how to configure TrafCI on the client workstation.
Chapter 3: Launching TrafCI	Describes how to launch, log in to, and exit TrafCI on a client workstation.
Chapter 4: Running Commands Interactively in TrafCI	Describes how to run commands interactively in TrafCI.
Chapter 5: Running Scripts in TrafCI	Describes how to run script files in TrafCI.
Chapter 6: Running TrafCI From Perl or Python	Describes how to run TrafCI from Perl or Python.
Appendix A: Interface Commands	Provides syntax, considerations, and examples for the interface commands.

New and Changed Information in This Edition

This manual is new.

Notation Conventions

General Syntax Notation

This list summarizes the notation conventions for syntax presentation in this manual.

UPPERCASE LETTERS

Uppercase letters indicate keywords and reserved words. Type these items exactly as shown. Items not enclosed in brackets are required. For example:

SELECT

Italic Letters

Italic letters, regardless of font, indicate variable items that you supply. Items not enclosed in brackets are required. For example:

file-name

Computer Type

Computer type letters within text indicate case-sensitive keywords and reserved words. Type these items exactly as shown. Items not enclosed in brackets are required. For example:

`myfile.sh`

[] Brackets

Brackets enclose optional syntax items. For example:

```
DATETIME [start-field TO] end-field
```

A group of items enclosed in brackets is a list from which you can choose one item or none. The items in the list can be arranged either vertically, with aligned brackets on each side of the list, or horizontally, enclosed in a pair of brackets and separated by vertical lines. For example:

```
DROP SCHEMA schema [CASCADE]
                        [RESTRICT]
```

```
DROP SCHEMA schema [ CASCADE | RESTRICT ]
```

{ } Braces

Braces enclose required syntax items. For example:

```
FROM { grantee [, grantee] ... }
```

A group of items enclosed in braces is a list from which you are required to choose one item. The items in the list can be arranged either vertically, with aligned braces on each side of the list, or horizontally, enclosed in a pair of braces and separated by vertical lines. For example:

```
INTERVAL { start-field TO end-field }
          { single-field }
```

```
INTERVAL { start-field TO end-field | single-field }
```

| Vertical Line

A vertical line separates alternatives in a horizontal list that is enclosed in brackets or braces. For example:

```
{expression | NULL}
```

... Ellipsis

An ellipsis immediately following a pair of brackets or braces indicates that you can repeat the enclosed sequence of syntax items any number of times. For example:

```
ATTRIBUTE[S] attribute [, attribute] ...
```

```
{ [, sql-expression } ...
```

An ellipsis immediately following a single syntax item indicates that you can repeat that syntax item any number of times. For example:

```
expression-n...
```

Punctuation

Parentheses, commas, semicolons, and other symbols not previously described must be typed as shown. For example:

```
DAY (datetime-expression)
```

```
@script-file
```

Quotation marks around a symbol such as a bracket or brace indicate the symbol is a required character that you must type as shown. For example:

```
"{ " module-name [, module-name] ... " }
```

Item Spacing

Spaces shown between items are required unless one of the items is a punctuation symbol such as a parenthesis or a comma. For example:

```
DAY (datetime-expression)
```

```
DAY(datetime-expression)
```

If there is no space between two items, spaces are not permitted. In this example, no spaces are permitted between the period and any other items:

```
myfile.sh
```

Line Spacing

If the syntax of a command is too long to fit on a single line, each continuation line is indented three spaces and is separated from the preceding line by a blank line. This spacing distinguishes items in a continuation line from items in a vertical list of selections. For example:

```
match-value [NOT] LIKE pattern  
  
[ESCAPE esc-char-expression]
```

Publishing History

Product Version	Publication Date	
Trafodion Release 1.3.0	November 2015	

We Encourage Your Comments

The Trafodion community encourages your comments concerning this document. We are committed to providing documentation that meets your needs. Send any errors found, suggestions for improvement, or compliments to:

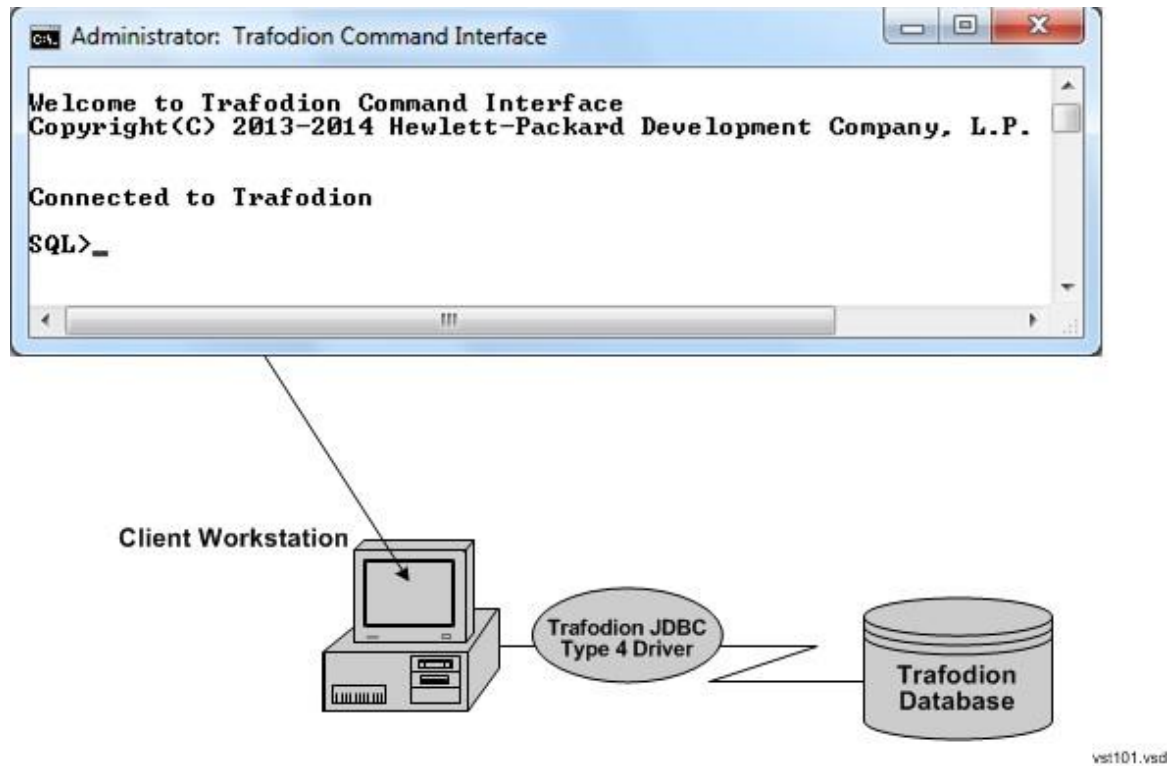
issues@trafodion.incubator.apache.org

Include the document title and any comment, error found, or suggestion for improvement you have concerning this document.

1 Introduction to the Trafodion Command Interface (TrafCI)

The Trafodion Command Interface (TrafCI) is a command-line interface that you download and install on a client workstation that has the Trafodion JDBC Type 4 Driver installed. Operating systems that support the JDBC driver include Windows and Linux. The JDBC driver connects TrafCI on a client workstation to a Trafodion database.

Figure 1 TrafCI Connected to a Trafodion Database



TrafCI enables you to perform daily administrative and database management tasks by running SQL statements or other commands interactively or from script files. You can also run TrafCI from a Perl or Python command line or from Perl or Python programs.

2 Installing and Configuring TrafCI

- “Installing TrafCI” (page 14)
- “Verifying and Setting the Java Path” (page 14)
- “Testing the Launch of TrafCI” (page 16)

Installing TrafCI

To install TrafCI on a client workstation, follow the procedures in the *Trafodion Client Installation Guide*.

Verifying and Setting the Java Path

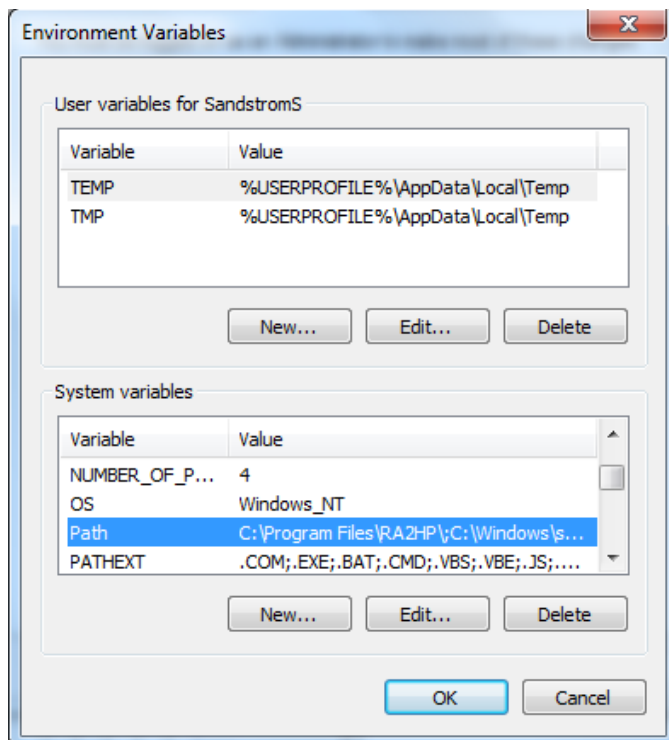
To be able to launch and run TrafCI, you must have the Java path set to the correct location. Follow these instructions:

- “Setting the PATH on Windows” (page 14)
- “Setting the PATH on Linux” (page 15)

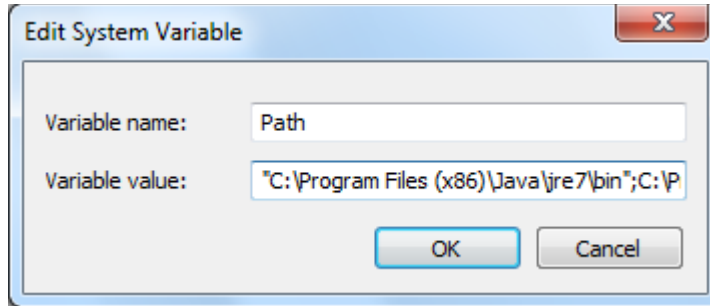
NOTE: To install the Java Runtime Environment (JRE), see the TrafCI installation instructions in the *Trafodion Client Installation Guide*.

Setting the PATH on Windows

1. Right-click the Computer icon on your desktop, and then select Properties.
The Control Panel > System and Security > System window appears.
2. In the left navigation bar, click the Advanced system settings link.
3. In the System Properties dialog box, click the Environment Variables button.
4. Under System variables, select the variable named Path, and then click Edit:



5. Place the cursor at the beginning of the Variable value field and enter the path of the Java `bin` directory, ending with a semicolon (;):



For example:

```
"C:\Program Files (x86)\Java\jre7\bin";
```

NOTE: Check that no space exists after the semicolon (;) in the path. If there are spaces in the directory name, delimit the entire directory path in double quotes (") before the semicolon.

6. Click OK.
7. Verify that the updated Path appears under System variables, and click OK.
8. In the System Properties dialog box, click OK to accept the changes.

Setting the PATH on Linux

1. Open the user profile (`.profile` or `.bash_profile` for the Bash shell) in the `/home` directory. For example:
2. In the user profile, set the PATH environment variable to include the path of the Java `bin` directory. For example:

```
vi .profile
```

```
export PATH=/opt/java1.7/jre/bin:$PATH
```

NOTE: Place the path of the Java `bin` directory before `$PATH`, and check that no space exists after the colon (:) in the path. In the C shell, use the `setenv` command instead of `export`.

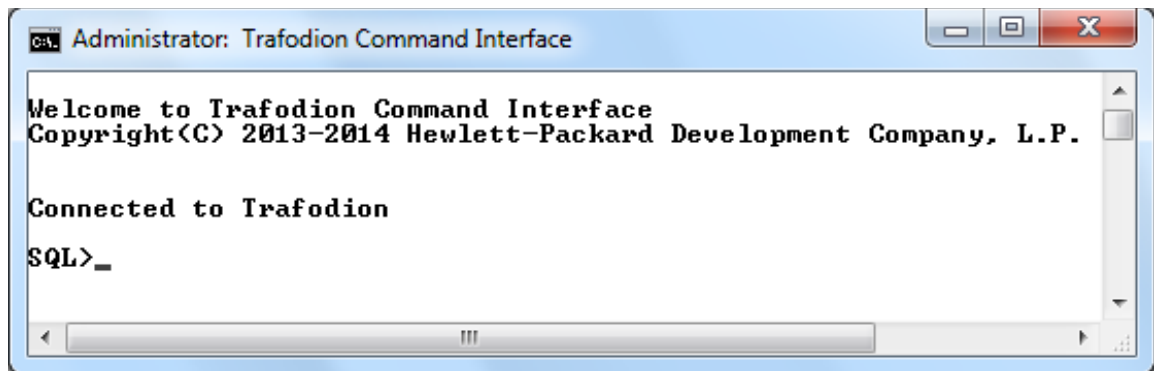
3. To activate the changes, either log out and log in again or execute the user profile. For example:

```
. .profile
```

Testing the Launch of TrafCI

1. Launch TrafCI and verify that you can connect to the database. For instructions, see [Chapter 3 \(page 17\)](#).

This window should appear:



2. If you cannot launch TrafCI or connect to the database, verify that:
 - The database platform is available and running, and the port number is correct for the database platform.
 - The Java path is set to the correct location. See [“Verifying and Setting the Java Path” \(page 14\)](#).
 - You installed the TrafCI and JDBC software files correctly. See the *Trafodion Client Installation Guide*.

3 Launching TrafCI

This chapter describes how to launch TrafCI from the Window or Linux environment of a client workstation:

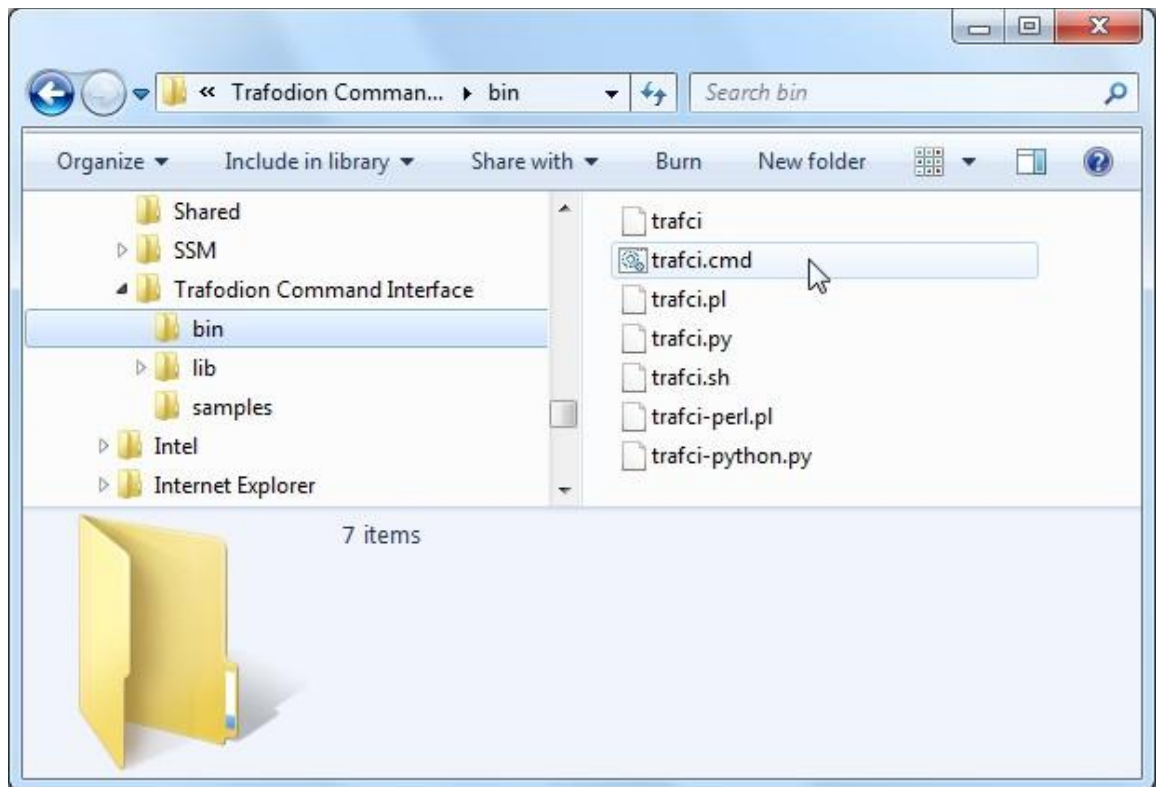
- “Launching TrafCI on a Windows Client Workstation” (page 17)
- “Launching TrafCI on a Linux Client Workstation” (page 20)
- “Logging In to the Database Platform” (page 21)
- “Using Optional Launch Parameters” (page 23)
- “Exiting TrafCI” (page 27)

For information about launching TrafCI from Perl or Python, see [Chapter 6 \(page 43\)](#).

- ❗ **IMPORTANT:** Before launching TrafCI, make sure that you have set the Java path to the correct location. See “[Verifying and Setting the Java Path](#)” (page 14).

Launching TrafCI on a Windows Client Workstation

1. Find the Windows launch file, `trafci.cmd`, in the `bin` folder:



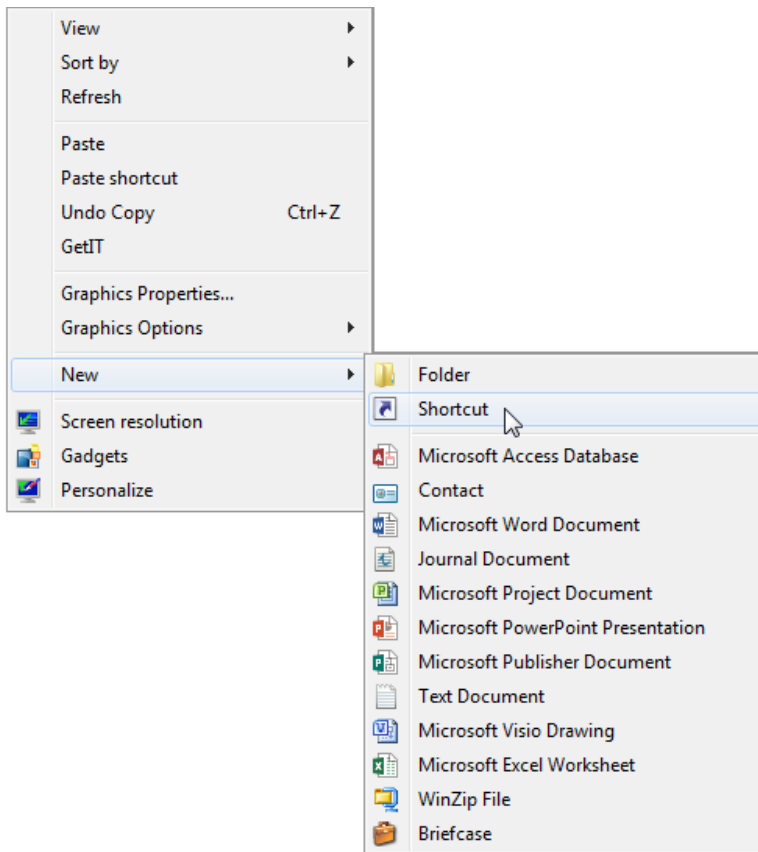
2. Double-click the `trafci.cmd` file.

TrafCI appears, prompting you to enter the host name or IP address of the database platform, your user name, and password. See “[Logging In to the Database Platform](#)” (page 21).

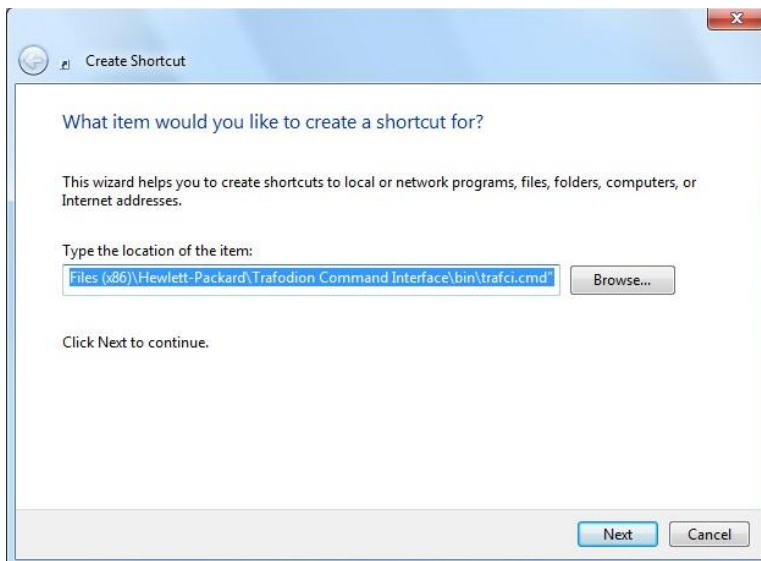
Creating a Shortcut to `trafci.cmd`

To enable a user to launch TrafCI from a shortcut icon on the desktop:

1. Right-click the desktop and select New > Shortcut:



2. Type the location of `trafci.cmd` within double quotes (") or click Browse to locate that file, and then click Next:

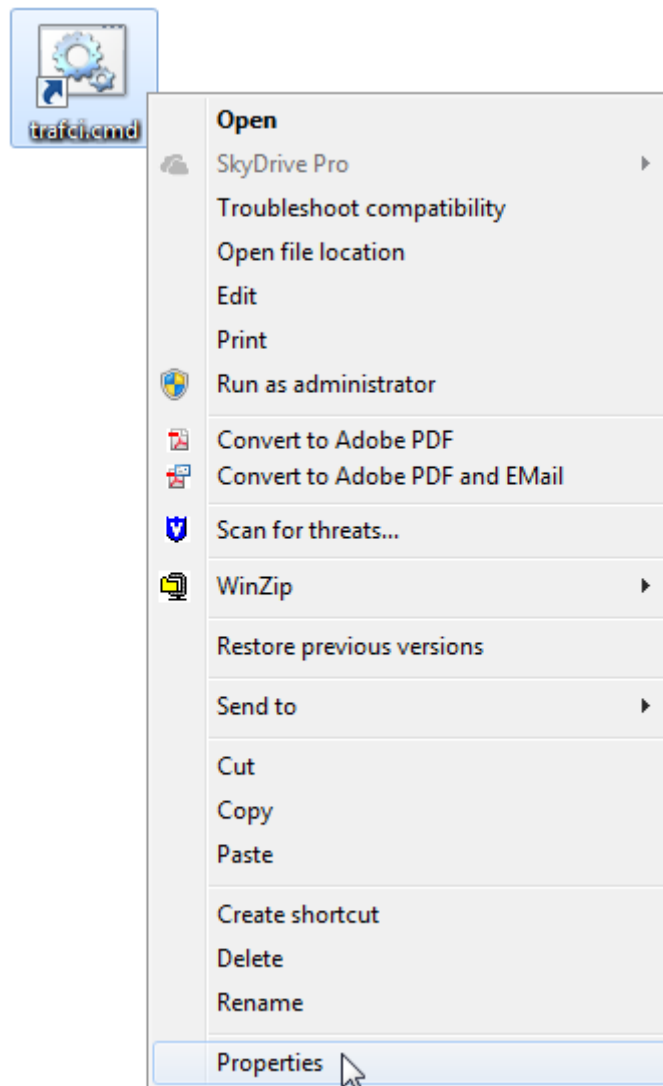


For the locations of the installed TrafCI software files, see the *Trafodion Client Installation Guide*.

3. Type a name for the shortcut and click Finish:

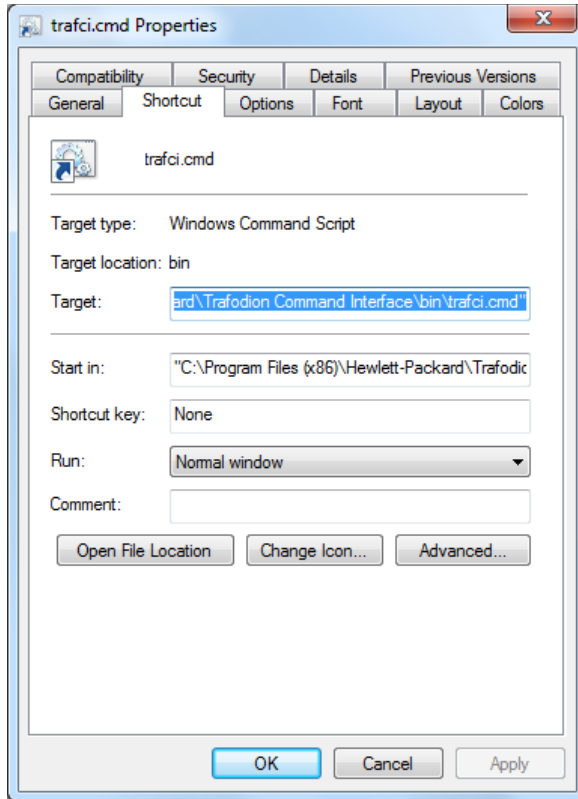


4. If desired, specify optional launch parameters for the shortcut:
 - a. Right-click the shortcut icon and select Properties:



- b. Select the Shortcut tab.

- c. In the Target box, insert a space after "...\\Trafodion Command Interface\\bin\\trafci.cmd" and add the optional launch parameters:



For more information, see [“Using Optional Launch Parameters”](#) (page 23).

- d. Click OK.
5. To launch TrafCI, double-click the shortcut icon.
- TrafCI appears. If you did not set the optional launch parameters, TrafCI prompts you to enter the host name or IP address of the database platform, your user name, and password. See [“Logging In to the Database Platform”](#) (page 21).

Launching TrafCI on a Linux Client Workstation

In the terminal window, enter:

```
./trafci-installation-directory/trafci/bin/trafci.sh
```

trafci-installation-directory is the directory where you installed the TrafCI software files. For more information, see the *Trafodion Client Installation Guide*.

Setting the PATH of trafci.sh

To enable a user to launch TrafCI anywhere on the client workstation:

1. Open the user profile (`.profile` or `.bash_profile` for the Bash shell) in the `/home` directory. For example:

```
vi .profile
```
2. In the user profile, set the PATH environment variable to include the path of the `trafci.sh` file. For example:

```
export PATH=/trafci-installation-directory/trafci/bin/:...
```

trafci-installation-directory is the directory where you installed the TrafCI software files. For more information, see the *Trafodion Client Installation Guide*. Check that no space is after the colon (:) in the path.

NOTE: In the C shell, use the `setenv` command instead of `export`.

3. To activate the changes, either log out and log in again or execute the user profile. For example:

```
. .profile
```
4. On the command line, execute the `trafci.sh` file to launch TrafCI:

```
trafci.sh
```

TrafCI appears, prompting you to enter the host name or IP address of the database platform, your user name, and password. See [“Logging In to the Database Platform” \(page 21\)](#).

NOTE: To enable all users to launch TrafCI anywhere on the system, create a symbolic link to the `trafci.sh` file in the `/usr/bin` or `/usr/local/bin` directory:

```
ln -s ./trafci-installation-directory/trafci/bin/trafci.sh /usr/bin/trafci.sh
```

Presetting the Optional Launch Parameters

To preset the optional launch parameters for each session, use an alias in the shell command. For example:

```
alias trafci='trafci.sh -h 16.123.456.78:37800 -u user1 -p xxxxxx'
```

You can add the alias, `trafci`, to the user profile, or you can enter it at a command prompt. For more information about the optional launch parameters, see [“Using Optional Launch Parameters” \(page 23\)](#).

Logging In to the Database Platform

- [“Logging In Without Using Login Parameters” \(page 21\)](#)
- [“Using Login Parameters” \(page 22\)](#)
- [“Retrying the Login” \(page 22\)](#)

Logging In Without Using Login Parameters

If you launch TrafCI and do not specify login parameters on the command line, follow these steps:

1. After you launch TrafCI, TrafCI shows the welcome banner and prompts you to enter the host name or IP address of the database platform:

```
Host Name/IP Address: _
```

Enter a host name:

```
host-name[.domain-name][:port-number]
```

- If you do not specify the domain name, TrafCI uses the domain of the client workstation.
- If you do not specify a port number, TrafCI uses the default port number, which is 37800.

Or enter an IP address:

```
IP-address[:port-number]
```

2. Enter your directory-service (or LDAP) username.
User names are case-insensitive.
3. Enter your password.
Passwords are case-sensitive.

4. After you finish logging in to the database platform, the SQL prompt appears:

```
Connected to Trafodion
```

```
SQL>
```

At the prompt, you can enter an SQL statement or an interface command. For more information, see [Chapter 4 \(page 28\)](#).

NOTE: TrafCI allows you to reenter the login values, with a maximum of three retries, before it closes the session. For more information, see [“Retrying the Login” \(page 22\)](#).

Using Login Parameters

To avoid entering a host name, user name, or password each time you launch TrafCI, use these login parameters:

- `-h` or `-host`
- `-u` or `-user`
- `-p` or `-password`

For example, on Windows, in the Command Prompt window, enter:

```
cd trafci-installation-directory\Trafodion Command Interface\bin
```

```
trafci.cmd -h 16.123.456.78:37800 -u user1 -p xxxxxx
```

For example, on Linux or UNIX, in the terminal window, enter:

```
cd trafci-installation-directory/trafci/bin
```

```
./trafci.sh -h 16.123.456.78:37800 -u user1 -p xxxxxx
```

TrafCI launches and prompts you to enter an SQL statement or an interface command:

```
Welcome to Trafodion Command Interface
Copyright(C) 2013-2105 Apache Software Foundation
```

```
Connected to Trafodion
```

```
SQL>
```

For more information about the login parameters, see [Table 1 \(page 24\)](#).



TIP: You can include these parameters in a shortcut to the `trafci.cmd` file or in a launch file for the `trafci.sh` file. For more information, see [“Creating a Shortcut to trafci.cmd” \(page 17\)](#) or [“Presetting the Optional Launch Parameters” \(page 21\)](#), respectively.

Retrying the Login

TrafCI allows you to reenter the login values, with a maximum of three retries, before it closes the session.

TrafCI applies the retry logic as follows:

- If you specify an invalid host name, TrafCI prompts you to reenter the host name. For example:

```
trafci -h dd
```

```
Welcome to Trafodion Command Interface
Copyright(C) 2013-2105 Apache Software
```

```
Unknown Host: dd
```

```
Host Name/IP Address: 172.16.1.1
```

```
User Name: user1
```


Password:

Connected to Trafodion

SQL>

- If you specify an invalid user name or password, TrafCI prompts you to reenter the user name and password. For example, if you specify an invalid password, TrafCI prompts only for your user name and password. After three unsuccessful retries, the session is terminated:

```
trafci -h 172.16.1.1 -u user1 -p x
```

```
Welcome to Trafodion Command Interface  
Copyright(C) 2013-2105 Apache Software
```

```
*** ERROR[8837] CLI Authentication : User: user1 : invalid username or password [2105-03-12 16:23:44]
```

```
User Name: user1  
Password:
```

```
*** ERROR[8837] CLI Authentication : User: user1 : invalid username or password [2105-03-12 16:25:28]
```

```
User Name: user1  
Password:
```

```
*** ERROR[8837] CLI Authentication : User: user1 : invalid username or password [2105-03-12 16:26:36]
```

Press any key to close this session

- If all the login parameters that you specify are invalid, TrafCI prompts you to enter the host name. When you enter a valid host name or IP address, TrafCI prompts you to enter a user name and password.
- The retry logic applies to the CONNECT and RECONNECT commands. For the RECONNECT command, the retry logic applies only when no prior connection has been established (`-noconnect`). This example shows the CONNECT command with a valid user name and host name. TrafCI prompts only for the user name and password:

```
SQL>connect user1/xxx@172.16.1.1
```

```
com.hp.jdbc.HPT4Exception: *** ERROR[8837] CLI Authentication : User: user1 : invalid username or password  
[2105-03-12 16:35:15]
```

```
User Name: user1  
Password: abc
```

Connected to Trafodion

SQL>

- TrafCI does not prompt you to reenter the login values in these cases:
 - When you include the `-q` or `-version` parameter on the command line. (The `-s` parameter permits login retries.)
 - For a session started using redirected or piped input.In these cases, TrafCI returns an error message and closes the session. You must re-launch the TrafCI session to connect to the Trafodion database.

Using Optional Launch Parameters

To customize how you launch and log in to TrafCI, use the optional parameters described in [Table 1](#) on the commandline:

```
trafci{.sh | .cmd} [optional-parameter]...
```

optional-parameter

is one of the launch or login parameters. For details, see [Table 1](#).

Table 1 Launch and Login Parameters

Launch or Login Parameter	Description
<pre>{-h -host} host-name[:port-number]</pre> <pre>{-h -host} IP-address[:port-number]</pre>	<p>Specifies the host name or IP address of the database platform to which you want the client to connect. The <i>host-name</i> should include the domain name of the database platform if it is different from the domain of the client workstation. If you do not specify a port number, TrafCI uses the default port number, which is 37800. For an example, see “Using Login Parameters” (page 22).</p>
<pre>{-u -user} username</pre>	<p>Specifies the user name for logging in to the database platform. The <i>username</i> is case-insensitive. For an example, see “Using Login Parameters” (page 22).</p>
<pre>{-r -role} role-name</pre>	<p>Reserved for future use.</p>
<pre>{-p -password} password</pre>	<p>Specifies the password of the user for logging in to the database platform. <i>password</i> is case-sensitive.</p> <p>For an example, see “Using Login Parameters” (page 22).</p>
<pre>{-q -sql} "command"</pre>	<p>Specifies that an SQL statement or an interface command be run when launching TrafCI. You cannot specify this parameter at the same time as the <code>-s</code> or <code>-script</code> parameter. For more information, see “Running a Command When Launching TrafCI” (page 24).</p>
<pre>{-s -script} script-file-name</pre>	<p>Specifies that a script file be run when launching TrafCI in interactive mode. You cannot specify this parameter at the same time as the <code>-q</code> or <code>-sql</code> parameter. For more information, see “Running a Script File When Launching TrafCI” (page 25).</p>
<pre>-noconnect</pre>	<p>Launches an TrafCI session without connecting to the database. For more information, see “Launching TrafCI Without Connecting to the Database” (page 26).</p>
<pre>-version</pre>	<p>Displays the build version of TrafCI and the Trafodion JDBC Type 4 Driver. Upon completion of the display, the client exits. If any other parameters are included with the <code>-version</code> parameter, they are ignored. For more information, see “Running TrafCI With -version” (page 27).</p>
<pre>-help</pre>	<p>Displays a list of accepted arguments with descriptions and then exits. For more information, see “Running TrafCI With -version” (page 27).</p>

Running a Command When Launching TrafCI

To execute an SQL statement or an interface command when launching TrafCI, use the `-q` or `-sql` command-line parameter. This parameter enables you to run a single command on the command line without having to enter commands in TrafCI.

NOTE: You cannot specify this parameter at the same time as the `-s` or `-script` parameter.

When using `-q` or `-sql`, you must enclose the command in double quotes. The SQL terminator is not required at the end of an SQL statement and is disallowed after an interface command.

Although you can run any of the interface commands with `-q` or `-sql`, the `@`, `OBEY`, and `PRUN` commands are the most useful.

Example of Running an SQL Statement With `-q` or `-sql`

Use `-q` or `-sql` with the `CREATE SCHEMA` statement to create a schema when launching TrafCI:

- On Windows, in the Command Prompt window, enter:

```
cd trafci-installation-directory\Trafodion Command Interface\bin
trafci.cmd -q "create schema persnl"
```

- On Linux or UNIX, in the terminal window, enter:

```
cd trafci-installation-directory/trafci/bin
./trafci.sh -q "create schema persnl"
```

After you enter the SQL statement, TrafCI launches and prompts you to log in by default (if you did not specify `-h`, `-u`, and `-p` on the command line), runs the SQL statement, and then returns to the command prompt:

```
Host Name/IP Address: 16.123.456.78:37800
User Name: user1
Password:
```

```
--- SQL operation complete.
```

```
C:\Program Files (x86)\Apache Software Foundation\Trafodion Command
Interface\bin>_
```

Example of Running an Interface Command With `-q` or `-sql`

Use `-q` or `-sql` with the `PRUN` command to run multiple script files simultaneously from the command line:

- On Windows, in the Command Prompt window, enter:

```
cd trafci-installation-directory\Trafodion Command Interface\bin
trafci.cmd -q "prun"
```

- On Linux, in the terminal window, enter:

```
cd trafci-installation-directory/trafci/bin
./trafci.sh -q "prun"
```

After you enter the interface command, TrafCI launches and prompts you to log in by default (if you did not specify `-h`, `-u`, and `-p` on the command line), and runs the command. The parallel run (`PRUN`) operation prompts you to enter settings and then executes the script files. At the end of the `PRUN` operation, TrafCI returns to the command prompt. For more information about the `PRUN` operation, see [“PRUN Command” \(page 80\)](#).

Running a Script File When Launching TrafCI

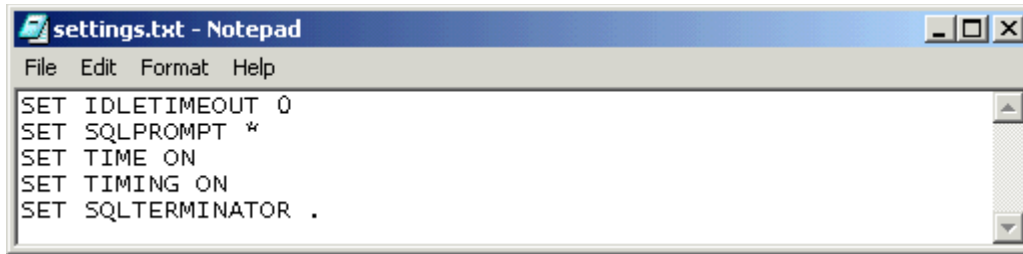
To run a script file when launching TrafCI, use the `-s` or `-script` command-line parameter.

NOTE: You cannot specify this parameter at the same time as the `-q` or `-sql` parameter.

After you launch TrafCI with `-s` or `-script`, TrafCI executes the script file in interactive mode. TrafCI remains open until you enter the `EXIT`, `QUIT`, or `DISCONNECT` command. To quit the interface immediately after executing a script file, include the `EXIT`, `QUIT`, or `DISCONNECT` command at the end of the script file.

Example of a Script File

You can create a script file that contains `SET` commands that customize a session when you launch TrafCI:



For more information, see [“Creating a Script File” \(page 40\)](#).

Example of Running a Script File With -s or -script

- On Windows, in the Command Prompt window, enter:

```
cd trafci-installation-directory\Trafodion Command Interface\bin
trafci.cmd -s settings.txt
```

Specify the full path of the script file if it is outside the directory of `trafci.cmd`.

- On Linux, in the terminal window, enter:

```
cd trafci-installation-directory/trafci/bin
./trafci.sh -s settings.txt
```

Specify the full path of the script file if it is outside the directory of `trafci.sh`.

TrafCI launches and prompts you to log in by default (if you did not specify `-h`, `-u`, and `-p` on the command line), and runs the commands in the script file:

```
Welcome to Trafodion Command Interface
Copyright(C) 2013-2105 Apache Software Foundation

Host Name/IP Address: 16.123.456.78:37800
User Name: user1
Password:

Connected to Trafodion

SQL>SET IDLETIMEOUT 0

SQL>SET SQLPROMPT *

*SET TIME ON

14:14:57 *SET TIMING ON

2:14:57 PM *SET SQLTERMINATOR .
```

Launching TrafCI Without Connecting to the Database

To start TrafCI without connecting to a Trafodion database, use the `-noconnect` option. See [Table 2 \(page 58\)](#) for a list of interface commands that can be run without a connection.

Example of Launching TrafCI With -noconnect

- On Windows, in the Command Prompt window, enter:

```
cd trafci-installation-directory\Trafodion Command Interface\bin
trafci.cmd -noconnect
```
- On Linux, in the terminal window, enter:

```
cd trafci-installation-directory/trafci/bin
./trafci.sh -noconnect
```

Running TrafCI With -version

To display the build version of TrafCI and the Trafodion JDBC Type 4 Driver, use the `-version` option. If other parameters are included with the `-version` parameter, they are ignored.

Example of Running TrafCI With -version

- On Windows, in the Command Prompt window, enter:

```
cd trafci-installation-directory\Trafodion Command Interface\bin
trafci.cmd -version
```
- On Linux, in the terminal window, enter:

```
cd trafci-installation-directory/trafci/bin
./trafci.sh -version
```

Welcome to Trafodion Command Interface
Copyright (C) 2013-2105 Apache Software Foundation

Trafodion JDBC Type 4 Driver	: Traf_JDBC_Type4_Build_40646
Trafodion Command Interface	: TrafCI_Build_40646

Running TrafCI With -help

To display a list of acceptable list of parameters, including proper usage information, use the `-help` option. After displaying this information the application exits.

Example of Running TrafCI With -help

- On Windows, in the Command Prompt window, enter:

```
cd trafci-installation-directory\Trafodion Command Interface\bin
trafci -help
```
- On Linux, in the terminal window, enter:

```
cd trafci-installation-directory/trafci/bin
./trafci.sh -help
```

Exiting TrafCI

To exit TrafCI, enter one of these commands at a prompt:

- EXIT
- QUIT

For example:

```
SQL>quit
```

These commands are not case-sensitive and do not require a terminator before you press Enter. After you enter one of these commands, TrafCI immediately quits running and disappears from the screen.

4 Running Commands Interactively in TrafCI

After launching TrafCI, you can run SQL statements and interface commands in the command-line interface.

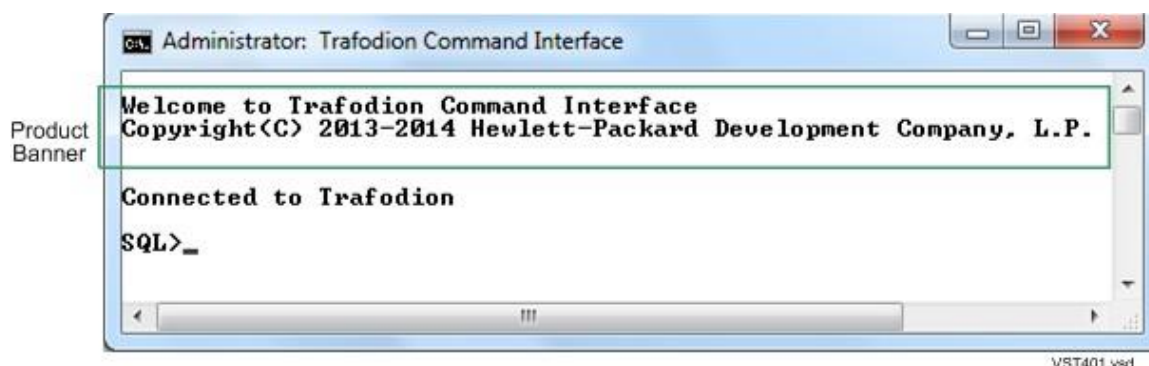
- “User Interface” (page 28)
- “Using the Interface Commands” (page 29)
- “Running SQL Statements” (page 33)
- “Logging Output” (page 38)

User Interface

- “Product Banner” (page 28)
- “Interface Prompt” (page 28)
- “Breaking the Command Line” (page 28)
- “Case Sensitivity” (page 29)

Product Banner

After you launch TrafCI and connect to the database platform, the product banner appears in the command-line interface:



Interface Prompt

The standard prompt is `SQL>`. You can change the prompt, `SQL>`, to something else by using the `SET SQLPROMPT` or `SET PROMPT` command. For more information, see the “[Customizing the Standard Prompt](#)” (page 30).

Breaking the Command Line

You cannot break an interface command over multiple lines. Each interface command must be entered on one line. If you accidentally break an interface command across more than one line, enter the SQL terminator and then reenter the command on one line.

You can continue any SQL statement over multiple lines, breaking that statement at any point except within a word, a numeric literal, or a multicharacter operator (for example, `<=`). To break a string literal in a DML statement, use a concatenation operator (`||`). For more information, see the concatenation operator in the *Trafodion SQL Reference Manual*.

To terminate an SQL statement that spans multiple lines, use the SQL terminator for the session. You can also include several SQL statements on the same command line provided that each one is terminated by the SQL terminator. For more information, see “[Setting and Showing the SQL Terminator](#)” (page 30).

Case Sensitivity

In the command-line interface, you can enter SQL statements and interface commands in uppercase, lowercase, or mixed-case characters. All parts of statements and commands are case-insensitive except for parts that you enclose in single-quotes (') or double-quotes (").

Using the Interface Commands

The interface commands allow you to customize TrafCI (for example, by using SET commands) or return information about the interface settings or database objects (for example, by using SHOW commands):

- “Showing the Session Attributes” (page 29)
- “Setting and Showing the Idle Timeout Value for the Session” (page 30)
- “Customizing the Standard Prompt” (page 30)
- “Setting and Showing the SQL Terminator” (page 30)
- “Displaying the Elapsed Time” (page 31)
- “Setting and Showing the Current Schema” (page 31)
- “Limiting the Result Set of a Query” (page 32)
- “Displaying Executed Commands” (page 32)
- “Editing and Reexecuting a Command” (page 33)
- “Clearing the Interface Window” (page 33)
- “Obtaining Help” (page 33)

For more information about the interface commands, see [Appendix A \(page 48\)](#).

NOTE: Each interface command must be entered on one line. If you accidentally break an interface command across more than one line, enter the SQL terminator and then reenter the command on one line.

Showing the Session Attributes

To display the attributes and settings of the current TrafCI session, use the ENV, SHOW SESSION, or SESSION command. For example, this SESSION command displays the session attributes:

```
SQL>session

COLSEP           " "
HISTOPT          DEFAULT [No expansion of script files]
IDLETIMEOUT      0 min(s) [Never Expires]
LIST_COUNT       0 [All Rows]
LOG_FILE         c:\session.txt
LOG_OPTIONS      APPEND,CMDTEXT ON
MARKUP           RAW
PROMPT           SQL>
SCHEMA           SEABASE
SERVER           sqws135.houston.host.com:378
00 SQLTERMINATOR ;
STATISTICS       OFF
TIME             OFF
TIMING           OFF
USER             user1

SQL>
```

For more information, see the “ENV Command” (page 59) or “SHOW SESSION Command” (page 129).

Setting and Showing the Idle Timeout Value for the Session

The idle timeout value of a session determines when the session expires after a period of inactivity. To set the idle timeout value of a session, enter the `SET IDLETIMEOUT` command. For example, this `SET IDLETIMEOUT 0` command sets the idle timeout to an infinite amount of time so that the session never expires:

```
SQL>set idletimeout 0
```

```
SQL>
```

To show the idle timeout value that is in effect for the session, enter the `SHOW IDLETIMEOUT` command. For example, this `SHOW IDLETIMEOUT` command displays an idle timeout of zero minutes, which means that the session never expires:

```
SQL>show idletimeout
IDLETIMEOUT 0 min(s) [Never Expires]
```

```
SQL>
```

For more information, see the [“SET IDLETIMEOUT Command” \(page 95\)](#) and the [“SHOW IDLETIMEOUT Command” \(page 120\)](#).

Customizing the Standard Prompt

To change the standard prompt in the command-line interface, use one or both of these commands:

- [“SET PROMPT Command” \(page 104\)](#)
- [“SET TIME Command” \(page 30\)](#)

SET PROMPT Command

The `SET PROMPT` command changes the default prompt to a specified character or string. For example, this `SET PROMPT` command changes the prompt to the current user (`user1`) and `ENTER>`:

```
SQL>set prompt "%USER ENTER>"
```

```
user1 ENTER>
```

For more information, see the [“SET PROMPT Command” \(page 104\)](#).

SET TIME Command

The `SET TIME ON` command causes the current time of the client workstation to be displayed in the prompt:

```
SQL ENTER>set time on
```

```
20:32:26 SQL ENTER>
```

The `SET TIME OFF` command removes the current time from the prompt:

```
20:32:26 SQL ENTER>set time off
```

```
SQL ENTER>
```

For more information, see the [“SET TIME Command” \(page 110\)](#).

Setting and Showing the SQL Terminator

The SQL terminator symbolizes the end of an SQL statement. By default, the SQL terminator is a semicolon (;).

To change the SQL terminator, enter the `SET SQLTERMINATOR` command. For example, this `SET SQLTERMINATOR` command sets the SQL terminator to a period (.):

```
SQL>set sqlterminator .
```

```
SQL>insert into sales.custlist
```



```
+>(select * from invent.supplier
+>where suppnun=8).
```

```
--- 1 row(s) inserted.
```

```
SQL>
```

To show the SQL terminator that is in effect for the session, enter the **SHOW SQLTERMINATOR** command. For example, this **SHOW SQLTERMINATOR** command displays **SQLTERMINATOR .**, where the period (.) is the SQL terminator for the session:

```
SQL>show sqlterminator
SQLTERMINATOR .
```

```
SQL>
```

For more information, see the [“SET SQLTERMINATOR Command” \(page 108\)](#) and the [“SHOW SQLTERMINATOR Command” \(page 132\)](#).

Displaying the Elapsed Time

By default, TrafCI does not display the elapsed time of an SQL statement after the statement executes. To display the elapsed time after each SQL statement executes, enter the **SET TIMING ON** command:

```
SQL>set timing on
```

```
SQL>select suppname, street, city, state, postcode
+>from invent.supplier
+>where suppnun=3;
```

SUPPNAME	STREET	CITY	STATE	POSTCODE
HIGH DENSITY INC	7600 EMERSON	NEW YORK	NEW YORK	10230

```
--- 1 row(s) selected.
```

```
Elapsed :00:00:00.111
```

```
SQL>
```

To prevent the elapsed time from being displayed after each SQL statement executes, enter the **SET TIMING OFF** command:

```
SQL>set timing off
```

```
SQL>/
```

SUPPNAME	STREET	CITY	STATE	POSTCODE
HIGH DENSITY INC	7600 EMERSON	NEW YORK	NEW YORK	10230

```
--- 1 row(s) selected.
```

```
SQL>
```

For more information, see the [“SET TIMING Command” \(page 111\)](#).

Setting and Showing the Current Schema

By default, the schema of the session is **USR**. The SQL statement, **SET SCHEMA**, allows you to set the schema for the TrafCI session. For example, this **SET SCHEMA** statement changes the default schema to **PERSNL** for the session:

```
SQL>set schema persnl;
```

```
--- SQL operation complete.
```

```
SQL>delete from employee
+>where first_name='TIM' and
+>last_name='WALKER';
```

```
--- 1 row(s) deleted.
```

```
SQL>
```

The schema that you specify with SET SCHEMA remains in effect until the end of the session or until you execute another SET SCHEMA statement.

If you execute this statement in a script file, it affects not only the SQL statements in the script file but all subsequent SQL statements that are run in the current session. If you set the schema in a script file, reset the default schema for the session at the end of the script file.

For more information about the SET SCHEMA statement, see the *Trafodion SQL Reference Manual*.

The SHOW SCHEMA command displays the current schema for the session. For example, this SHOW SCHEMA command displays SCHEMA PERSONL, where PERSONL is the name of the current schema for the session:

```
SQL>show schema
SCHEMA PERSONL
```

```
SQL>
```

For more information, see the [“SHOW SCHEMA Command” \(page 128\)](#).

Limiting the Result Set of a Query

To set the maximum number of rows to be returned by SELECT statements that are executed in the session, enter the SET LIST_COUNT command. For example, this SET LIST_COUNT command limits the result set of queries to 20 rows:

```
SQL>set list_count 20
```

To show the limit that is in effect for the session, enter the SHOW LIST_COUNT command. For example, this SHOW LIST_COUNT command shows that the number of rows returned by SELECT statements is unlimited:

```
SQL>show list_count
LISTCOUNT 0 [All Rows]
```

For more information, see the [“SET LIST_COUNT Command” \(page 96\)](#) and the [“SHOW LIST_COUNT Command” \(page 122\)](#).

Displaying Executed Commands

To display commands that were recently executed in the TrafCI session, enter the HISTORY command. The HISTORY command associates each command with a number that you can use to reexecute or edit the command with the FC command. See [“Editing and Reexecuting a Command” \(page 33\)](#).

For example, this HISTORY command displays a maximum of 100 commands that were entered in the session:

```
SQL>history
1>      set idletimeout 0
2>      set schema persnl;
3>      select * from project;
```

```
SQL>
```

To save the session history in a user-specified file, enter the SAVEHIST command. For example, this SAVEHIST command saves the session history in a file named `history.txt` in the local directory where you are running TrafCI:

```
SQL>savehist history.txt
```

For more information, see the [“HISTORY Command” \(page 69\)](#) and the [“SAVEHIST Command” \(page 90\)](#).

Editing and Reexecuting a Command

To edit and reexecute a command in the history buffer of an TrafCI session, enter the FC command. To display the commands in the history buffer, use the HISTORY command. See [“Displaying Executed Commands” \(page 32\)](#).

For example, this FC command and its delete (D) editing command correct a SELECT statement that was entered incorrectly:

```
SQL>fc
SQL>selecct * from employee;
....      d
SQL>select * from employee;
....
```

Pressing Enter executes the corrected SELECT statement.

For more information, see the [“FC Command” \(page 62\)](#).

Clearing the Interface Window

After entering commands in TrafCI, you can clear the interface window by using the CLEAR command. For example, this CLEAR command clears the interface window so that only the prompt appears at the top of the window:

```
SQL>clear
```

For more information, see the [“CLEAR Command” \(page 54\)](#).

Obtaining Help

To display help text for an interface command that is supported in TrafCI, enter the HELP command. For example, this HELP command displays syntax and examples of the FC command:

```
SQL>help fc
```

For more information, see the [“HELP Command” \(page 68\)](#).

Running SQL Statements

In TrafCI, you can run SQL statements interactively. TrafCI supports all the SQL statements, SQL utilities, and other SQL-related commands that the Trafodion database engine supports. For more information about those SQL statements, see the *Trafodion SQL Reference Manual*.

This subsection shows examples of:

- [“Executing an SQL Statement” \(page 33\)](#)
- [“Repeating an SQL Statement” \(page 34\)](#)
- [“Preparing and Executing SQL Statements” \(page 34\)](#)

To run SQL statements from script files in TrafCI, see [Chapter 5 \(page 40\)](#).

Executing an SQL Statement

For example, you can query the EMPLOYEE table and return an employee’s salary by executing this SELECT statement in TrafCI:

```
SQL>select salary
+>from persnl.employee
+>where jobcode=100;
```

```
SALARY
-----
175500.00
137000.10
139400.00
138000.40
75000.00
```

```

90000.00
118000.00
80000.00
70000.00
90000.00
56000.00

--- 11 row(s) selected.

```

SQL>

If the SQL statement executes successfully, TrafCI returns a message indicating that the SQL operation was successful, followed by the standard prompt. If a problem occurs during the execution of the SQL statement, TrafCI returns an error message.

Repeating an SQL Statement

To run a previously executed SQL statement, use the `/`, `RUN`, or `REPEAT` command.

SQL>/

```

SALARY
-----
175500.00
137000.10
139400.00
138000.40
75000.00
90000.00
118000.00
80000.00
70000.00
90000.00
56000.00

--- 11 row(s) selected.

```

SQL>

For more information, see the [“/ Command”](#) (page 52), [“RUN Command”](#) (page 89), or [“REPEAT Command”](#) (page 85).

Preparing and Executing SQL Statements

You can prepare, or compile, an SQL statement by using the `PREPARE` statement and later execute the prepared SQL statement by using the `EXECUTE` statement.

- [“Preparing an SQL Statement”](#) (page 34)
- [“Setting Parameters”](#) (page 35)
- [“Displaying the Parameters of the Session”](#) (page 36)
- [“Resetting the Parameters”](#) (page 36)
- [“Executing a Prepared SQL Statement”](#) (page 36)

Preparing an SQL Statement

Use the `PREPARE` statement to compile an SQL statement for later execution with the `EXECUTE` statement. You can also use the `PREPARE` statement to check the syntax of an SQL statement without executing the statement. For example, this `PREPARE` statement compiles a `SELECT` statement named `empsal` and detects a syntax error:

```

SQL>prepare empsal from
+>select salary from employee
+>where jobcode = 100;

```

SQL>

You can then correct the syntax of the SQL statement and prepare it again:

```
SQL>prepare empsal from
+>select salary from persnl.employee
+>where jobcode = 100;
```

--- SQL command prepared.

To specify a parameter to be supplied later, either in a SET PARAM statement or in the USING clause of an EXECUTE statement, use one of these types of parameters in the SQL statement:

- Named parameter, which is represented by *?param-name*
- Unnamed parameter, which is represented by a question mark (?) character

For example, this prepared SELECT statement specifies unnamed parameters for salary and job code:

```
SQL>prepare findemp from
+>select * from persnl.employee
+>where salary > ? and jobcode = ?;
```

--- SQL command prepared.

This PREPARE statement prepares another SELECT statement named *empcom*, which has one named parameter, *?dn*, for the department number, which appears twice in the statement:

```
SQL>prepare empcom from
+>select first_name, last_name, deptnum
+>from persnl.employee
+>where deptnum <> ?dn and salary <=
+>(select avg(salary)
+>from persnl.employee
+>where deptnum = ?dn);
```

--- SQL command prepared.

For the syntax of the PREPARE statement, see the *Trafodion SQL Reference Manual*.

Setting Parameters

In an TrafCI session, you can set a parameter of an SQL statement (either prepared or not) by using the SET PARAM command.

NOTE: The parameter name is case-sensitive. If you specify it in lowercase in the SET PARAM command, you must specify it in lowercase in other statements, such as DML statements or EXECUTE.

For example, this SET PARAM command sets a value for the parameter named *?sal*, which you can apply to one of the unnamed parameters in the prepared *findemp* statement or to a named parameter with an identical name in an SQL statement:

```
SQL>set param ?sal 40000.00
```

This SELECT statement uses *sal* as a named parameter:

```
SQL>select last_name
+>from persnl.employee
+>where salary = ?sal;
```

This SET PARAM command sets a value for the parameter named *dn*, which you can apply to the named parameter, *?dn*, in the prepared *empcom* statement or to a named parameter with an identical name in an SQL statement:

```
SQL>set param ?dn 1500
```

For the syntax of the SET PARAM command, see the [“SET PARAM Command” \(page 102\)](#).

Displaying the Parameters of the Session

To determine what parameters you have set in the current session, use the `SHOW PARAM` command. For example, this `SHOW PARAM` command displays the recent `SET PARAM` settings:

```
SQL>show param
dn 1500
sal 40000.00
```

```
SQL>
```

For the syntax of the `SHOW PARAM` command, see the [“SHOW PARAM Command” \(page 124\)](#).

Resetting the Parameters

To change the value of a parameter, specify the name of the parameter in the `RESET PARAM` command and then use the `SET PARAM` command to change the setting. For example, suppose that you want to change the salary parameter to 80000.00:

```
SQL>reset param ?sal
```

```
SQL>set param ?sal 80000.00
```

```
SQL>
```

Entering the `RESET PARAM` command without specifying a parameter name clears all parameter settings in the session. For example:

```
SQL>reset param
```

```
SQL>show param
```

```
SQL>
```

To use the parameters that you had set before, you must reenter them in the session:

```
SQL>set param ?dn 1500
```

```
SQL>set param ?sal 80000.00
```

```
SQL>show param
dn 1500
sal 80000.00
```

```
SQL>
```

For the syntax of the `RESET PARAM` command, see the [“RESET PARAM Command” \(page 88\)](#).

Executing a Prepared SQL Statement

To execute a prepared SQL statement, use the `EXECUTE` statement.

For example, this `EXECUTE` statement executes the prepared `empsal` statement, which does not have any parameters:

```
SQL>execute empsal;
```

```
SALARY
-----
137000.10
 90000.00
 75000.00
138000.40
 56000.00
136000.00
 80000.00
 70000.00
175500.00
 90000.00
118000.00
```

--- 11 row(s) selected.

SQL>

This EXECUTE statement executes the prepared `empcom` statement, which has one named parameter, `?dn`, which was set by SET PARAM for the department number:

SQL>execute empcom;

FIRST_NAME	LAST_NAME	DEPTNUM
ALAN	TERRY	3000
DAVID	TERRY	2000
PETE	WELLINGTON	3100
JOHN	CHOU	3500
MANFRED	CONRAD	4000
DINAH	CLARK	9000
DAVE	FISHER	3200
GEORGE	FRENCHMAN	4000
KARL	HELMSTED	4000
JOHN	JONES	4000
JOHN	HUGHES	3200
WALTER	LANCASTER	4000
MARLENE	BONNY	4000
BILL	WINN	2000
MIRIAM	KING	2500
GINNY	FOSTER	3300
MARIA	JOSEF	4000
HERB	ALBERT	3300
RICHARD	BARTON	1000
XAVIER	SEDLMEYER	3300
DONALD	TAYLOR	3100
LARRY	CLARK	1000
JIM	HERMAN	3000
GEORGE	STRICKER	3100
OTTO	SCHNABL	3200
TIM	WALKER	3000
TED	MCDONALD	2000
PETER	SMITH	3300
MARK	FOLEY	4000
HEIDI	WEIGL	3200
ROCKY	LEWIS	2000
SUE	CRAMER	1000
MARTIN	SCHAEFFER	3200
HERBERT	KARAJAN	3200
JESSICA	CRINER	3500

--- 35 row(s) selected.

SQL>

This EXECUTE statement executes the prepared `findemp` statement, which has two unnamed parameters: `?sal`, which was set by SET PARAM for the salary, and a parameter that was not set in advance for the job code:

SQL>execute findemp using ?sal, 100;

EMPNUM	FIRST_NAME	LAST_NAME	DEPTNUM	JOBCODE	SALARY
213	ROBERT	WHITE	1500	100	90000.00
23	JERRY	HOWARD	1000	100	137000.10
1	ROGER	GREEN	9000	100	175500.00
29	JANE	RAYMOND	3000	100	136000.00
32	THOMAS	RUDLOFF	2000	100	138000.40
43	PAUL	WINTER	3100	100	90000.00

```
65 RACHEL          MCKAY          4000      100  118000.00
```

```
--- 7 row(s) selected.
```

```
SQL>
```

For the syntax of the EXECUTE statement, see the *Trafodion SQL Reference Manual*.

Logging Output

To log an TrafCI session, use the SPOOL or LOG command. The SPOOL and LOG commands record into a log file the commands that you enter in the command-line interface and the output of those commands.

- [“Starting the Logging Process” \(page 38\)](#)
- [“Stopping the Logging Process” \(page 39\)](#)
- [“Viewing the Contents of a Log File” \(page 39\)](#)

Starting the Logging Process

To start logging, enter one of these commands:

- SPOOL ON or LOG ON
- SPOOL *log-file* or LOG *log-file*

For more information, see the [“LOG Command” \(page 74\)](#) and the [“SPOOL Command” \(page 136\)](#).

SPOOL ON or LOG ON Command

The SPOOL ON or LOG ON command logs information about a session in the `sqlspool.lst` file, which TrafCI stores in the `bin` directory:

- On Windows:

```
trafci-installation-directory\Trafodion Command Interface\bin\sqlspool.lst
```

trafci-installation-directory is the directory where you installed the TrafCI software files.
- On Linux:

```
trafci-installation-directory/trafci/bin/sqlspool.lst
```

trafci-installation-directory is the directory where you installed the TrafCI software files.

For example, this SPOOL ON command starts logging the session in the `sqlspool.lst` file:

```
SQL>spool on
```

SPOOL *log-file* or LOG *log-file* Command

The SPOOL *log-file* and LOG *log-file* commands record information about a session in a log file that you specify. If you specify a directory for the log file, the directory must exist as specified. Otherwise, an error occurs when you try to run the SPOOL or LOG command. If you do not specify a directory for the log file, TrafCI uses the `bin` directory.

For example, this SPOOL *log-file* command starts logging the session in the `persnl_updates.log` file in the `C:\log` directory:

```
SQL>spool C:\log\persnl_updates.log
```

Using the CLEAR Option

The CLEAR option clears the contents of an existing log file before logging new information to the file. If you omit CLEAR, TrafCI appends new information to existing information in the log file.

For example, this `SPOOL log-file CLEAR` command clears existing information from the specified log file and starts logging the session in the log file:

```
SQL>spool C:\log\persnl_updates.log clear
```

Logging Concurrent the TrafCI Sessions

If you plan to run two or more TrafCI sessions concurrently on the same workstation, use the `SPOOL log-file` or `LOG log-file` command and specify a unique name for each log file. Otherwise, each session writes information to the same log file, making it difficult to determine which information belongs to each session.

Stopping the Logging Process

To stop logging, enter one of these commands:

- `SPOOL OFF`
- `LOG OFF`

For example, this `SPOOL OFF` command stops logging in an TrafCI session:

```
SQL>spool off
```

Viewing the Contents of a Log File

The log file is an ASCII text file that contains all the lines in TrafCI from the time you start logging to the time you stop logging. The logged lines include prompts, entered commands, output from commands, and diagnostic messages.

For example, this log file contains information from when you started logging to when you stopped logging:

```
=====
Spooling started at May 29, 2105 4:52:23 PM
=====

SQL>set transaction isolation level serializable;

--- SQL operation complete.

SQL>begin work;

--- SQL operation complete.

SQL>delete from employee where empnum=32;

-- 1 row(s) deleted.

SQL>insert into employee
(empnum, first_name, last_name, deptnum, salary)
values(51, 'JERRY', 'HOWARD', 1000, 137000.00);

-- 1 row(s) inserted.

SQL>update dept
set manager=50
where deptnum=1000;

--- 1 row(s) updated.

SQL>commit work;

--- SQL operation complete.

SQL>log off
```

5 Running Scripts in TrafCI

In TrafCI, you can run script files.

- [“Creating a Script File” \(page 40\)](#)
- [“Running a Script File” \(page 41\)](#)
- [“Logging Output” \(page 42\)](#)
- [“Running Scripts in Parallel” \(page 42\)](#)

Creating a Script File

A script file that you run in TrafCI must be an ASCII text file that contains only these elements:

- [“SQL Statements” \(page 40\)](#)
- [“Interface Commands” \(page 40\)](#)
- [“Comments” \(page 40\)](#)
- [“Section Headers” \(page 40\)](#)

For an example, see [“Example of a Script File” \(page 41\)](#).

NOTE: You cannot use shell commands in a script file that you run in TrafCI. To create shell scripts that run TrafCI, see [Chapter 6 \(page 43\)](#).

SQL Statements

Script files support any of the various SQL statements that you can run in TrafCI. For more information about SQL statements, see the *Trafodion SQL Reference Manual*.

Interface Commands

Most interface commands are supported in script files except the FC command. For a list of the interface commands, see [Appendix A \(page 48\)](#).

Comments

You can include comments anywhere in a script file. SQL also supports comments. Comments are useful for documenting the functionality of the script file and for debugging. When debugging, use comments to disable specific statements or commands without removing them from the script file.

To denote a comment in a script file, use two hyphens before the comment:

```
-- comment
```

The end of the line marks the end of the comment.

Section Headers

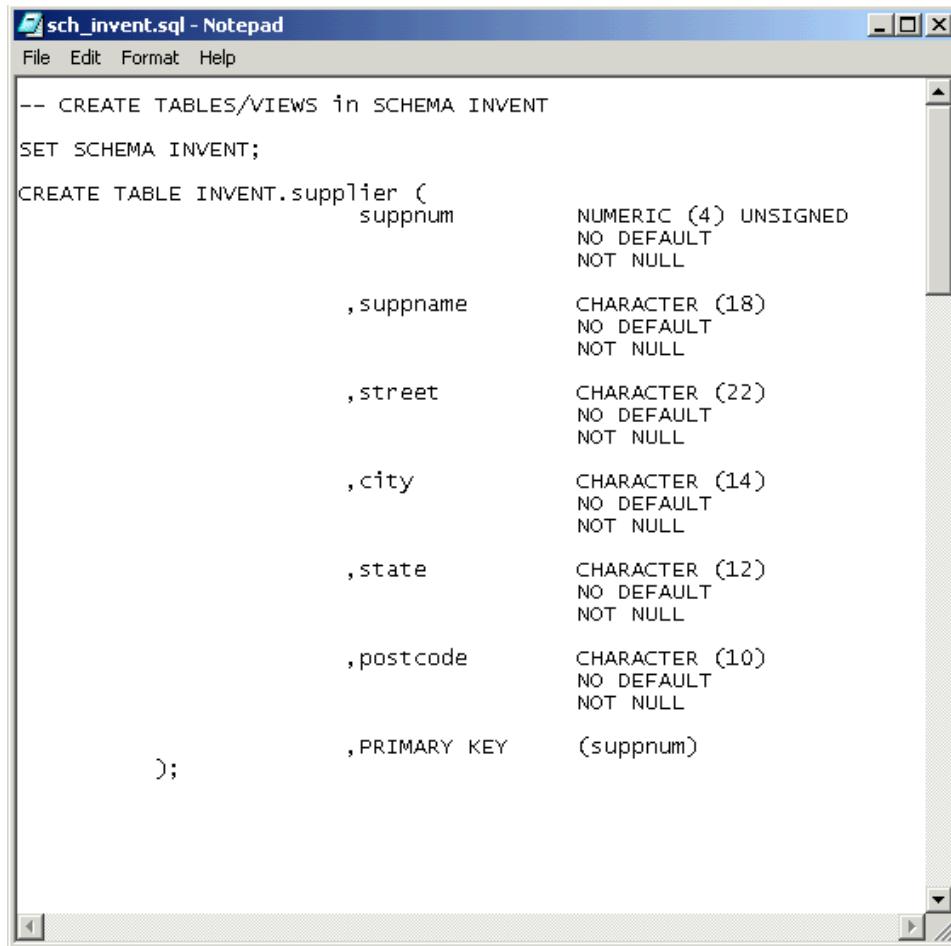
To create sections of commands within a script file, put a section header at the beginning of each section:

```
?SECTION section-name
```

The *section-name* cannot begin with a number or an underscore. Each section name in a script file should be unique because TrafCI executes the first section that it finds that matches the section name in the @ or OBEY command. For more information, see the [“@ Command” \(page 51\)](#) or the [“OBEY Command” \(page 77\)](#).

Example of a Script File

This script file creates tables in the inventory schema:



```
sch_invent.sql - Notepad
File Edit Format Help

-- CREATE TABLES/VIEWS in SCHEMA INVENT
SET SCHEMA INVENT;
CREATE TABLE INVENT.supplier (
    suppnym          NUMERIC (4) UNSIGNED
                     NO DEFAULT
                     NOT NULL
    ,suppname        CHARACTER (18)
                     NO DEFAULT
                     NOT NULL
    ,street          CHARACTER (22)
                     NO DEFAULT
                     NOT NULL
    ,city            CHARACTER (14)
                     NO DEFAULT
                     NOT NULL
    ,state           CHARACTER (12)
                     NO DEFAULT
                     NOT NULL
    ,postcode        CHARACTER (10)
                     NO DEFAULT
                     NOT NULL
    ,PRIMARY KEY     (suppnym)
);
```

Running a Script File

To run a script file in TrafCI, use the @ or OBEY command. The @ and OBEY commands run one script file at a time in TrafCI. To run a script file when launching TrafCI, see [“Running a Script File When Launching TrafCI” \(page 25\)](#).

For example, this @ command runs a script file, `sch_invent.sql`, that creates tables in the inventory schema:

```
@C:\ddl_scripts\sch_invent.sql
```

NOTE: If the script file is outside the directory of the `trafci.cmd` or `trafci.sh` file (by default, the `bin` directory), you must specify the full path of the script file in the @ or OBEY command.

```
SQL>@C:\ddl_scripts\sch_invent.sql
```

```
SQL>-- CREATE SCHEMA
```

```
SQL>CREATE SCHEMA INVENT;
```

```
--- SQL operation complete.
```

```
SQL>-- CREATE TABLES/VIEWS in SCHEMA INVENT
```

```
SQL>SET SCHEMA INVENT;
```

```
--- SQL operation complete.
```

```

SQL>CREATE TABLE INVENT.supplier (
+>          suppnnum          NUMERIC (4) UNSIGNED
+>                               NO DEFAULT
+>                               NOT NULL
+>          ,suppname         CHARACTER (18)
+>                               NO DEFAULT
+>                               NOT NULL
+>          ,street           CHARACTER (22)
+>                               NO DEFAULT
+>                               NOT NULL
+>          ,city             CHARACTER (14)
+>                               NO DEFAULT
+>                               NOT NULL
+>          ,state            CHARACTER (12)
+>                               NO DEFAULT
+>                               NOT NULL
+>          ,postcode         CHARACTER (10)
+>                               NO DEFAULT
+>                               NOT NULL
+>          ,PRIMARY KEY      (suppnnum)
+>      );

```

--- SQL operation complete.

For more information about the @ and OBEY commands, see the “[@ Command](#)” (page 51) and the “[OBEY Command](#)” (page 77).

Logging Output

To log output of an TrafCI session while running one script file at a time, use the SPOOL or LOG command. When you run an OBEY or @ command, TrafCI displays each command in the script file, the output for each command, and diagnostic messages in TrafCI. The SPOOL or LOG command captures this output as it appears in TrafCI and logs it in a log file.

For more information, see “[Logging Output](#)” (page 38).

Running Scripts in Parallel

In TrafCI, the @ and OBEY commands allow you to run only one script file at a time. However, the PRUN command allows you to run multiple script files simultaneously.

The PRUN command is most useful for running sets of data definition language (DDL) statements simultaneously, which speeds up the process of creating large databases. Put all dependent or related DDL statements in the same script file. For more information on running scripts in parallel using the PRUN command, see the “[PRUN Command](#)” (page 80).

6 Running TrafCI From Perl or Python

You can execute SQL statements in Perl or Python by invoking the TrafCI Perl or Python wrapper script. To use the Perl or Python wrapper script, see:

- “Setting the Login Environment Variables” (page 43)
- “Perl and Python Wrapper Scripts” (page 46)
- “Launching TrafCI From the Perl or Python Command Line” (page 46)

These instructions assume that you installed the TrafCI product. For more information, see [Chapter 2](#) (page 14).

Setting the Login Environment Variables

Before launching TrafCI from Perl or Python, set these login environment variables:

Environment Variable	Description
<code>TRAFCI_PERL_JSERVER=JavaServer_jar_path</code>	Specifies the Perl JavaServer JAR location.
<code>TRAFCI_PYTHON_JSERVER=Jython_jar_path</code>	Specifies the Jython JAR file location.
<code>TRAFCI_PERL_JSERVER_PORT=port_number</code>	Specifies the port on which the JavaServer is listening.

The Trafodion Command Interface Installer Wizard can attempt to automatically download and install both the Perl JavaServer and Jython open source extensions. If you wish to download and install them manually, refer to the instructions in the README in the samples directory.

To set the login environment variables, see the instructions for the operating system of the client workstation:

- “Setting the Login Environment Variables on Windows” (page 43)
- “Setting the Login Environment Variables on Linux or UNIX” (page 45)

NOTE: The Perl and Python wrapper scripts do not require these environment variables:

- `TRAFCI_SERVER`
- `TRAFCI_USER`
- `TRAFCI_PASSWORD`

Setting the Login Environment Variables on Windows

You can set the login environment variables for the session at command prompts, or you can set the login environment variables for the system or user by including them in the System Properties.

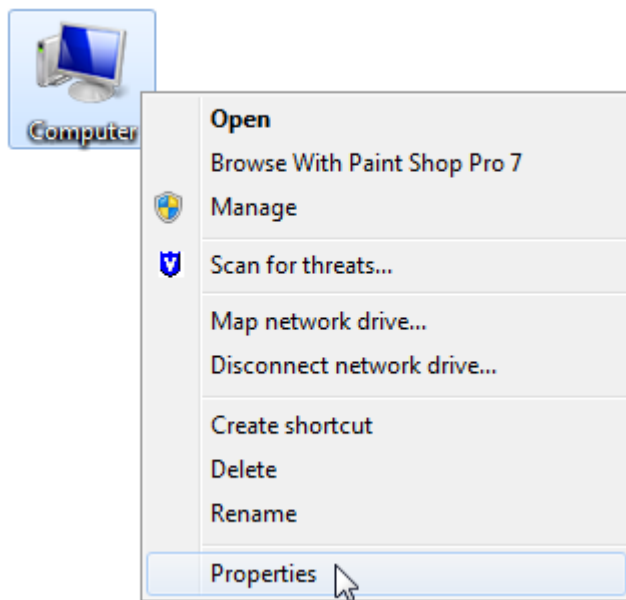
Setting Login Environment Variables on the Command Line

At each command prompt, enter one of these commands:

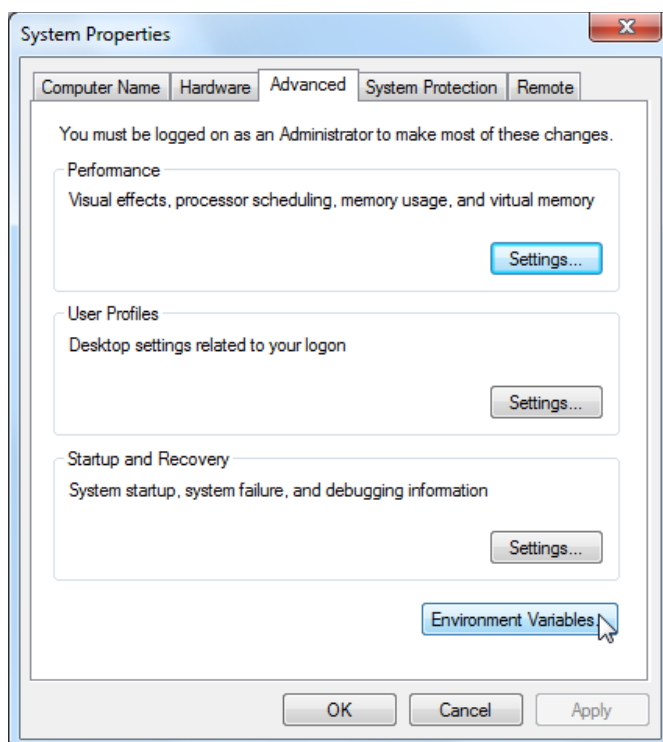
```
set TRAFCI_PERL_JSERVER=absolute-path-of-JavaServer.jar
set TRAFCI_PYTHON_JSERVER=absolute-path-of-Jython.jar
set TRAFCI_PERL_JSERVER_PORT=portnumber
```

Setting Login Environment Variables in the System Properties

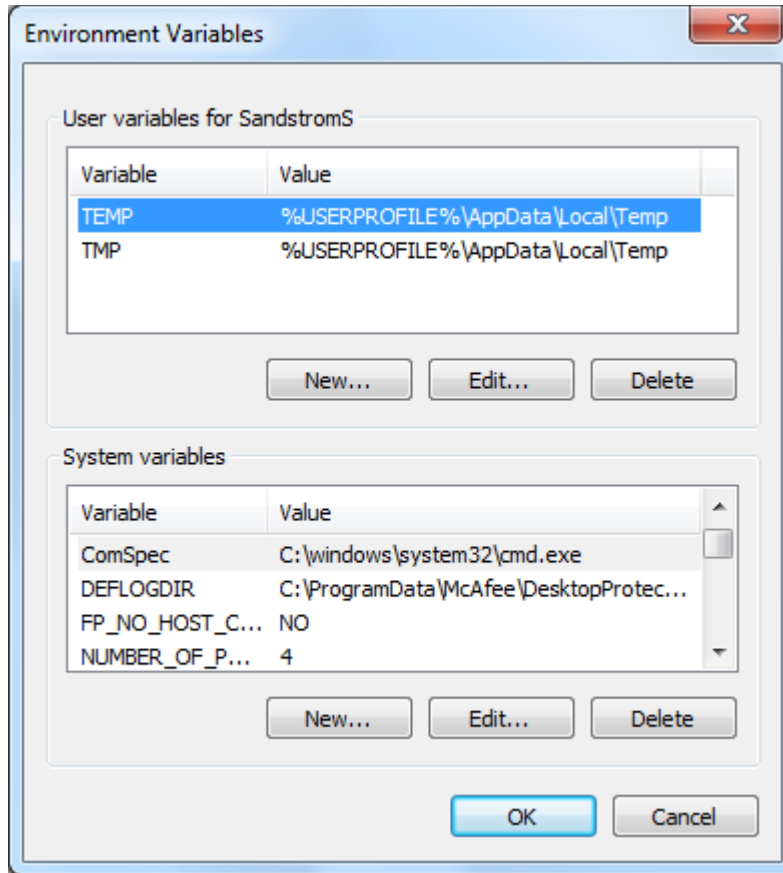
1. Right-click the Computer icon on your desktop, and then select Properties:



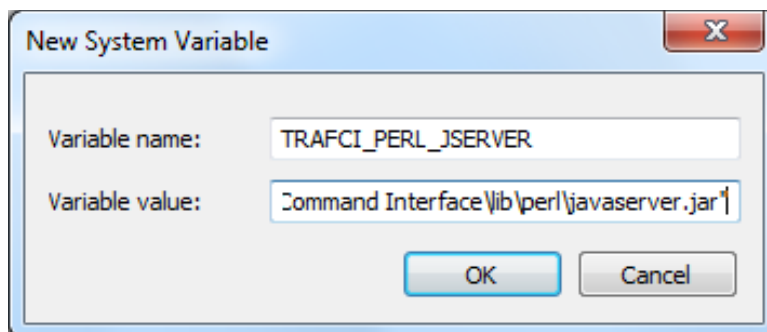
2. In the Control Panel, click the Advanced system settings.
3. In the System Properties dialog box, click the Advanced tab.
4. Click the Environment Variables button:



5. In the Environment Variables dialog box, click New under System or User variables, whichever you prefer.



6. In the New User Variable dialog box, type the name of the login environment variable for the Variable Name and the required value for the Variable Value, and then click OK:



7. Verify that the environment variable appears under System or User variables.
8. Repeat [Step 5](#) to [Step 7](#) for each login environment variable.
9. After adding all three environment variables, click OK in the Environment Variables and System Properties dialog boxes to accept the changes.

Setting the Login Environment Variables on Linux or UNIX

You can set the login environment variables for the session at command prompts, or you can set the login environment variables for each user by including the variables in the user profile on a Linux or UNIX client workstation.

Setting Login Environment Variables on the Command Line

At each command prompt in any shell except the C shell, enter one of these commands:

```
export TRAFCI_PERL_JSERVER=absolute-path-of-JavaServer.jar
export TRAFCI_PYTHON_JSERVER=absolute-path-of-Jython.jar
export TRAFCI_PERL_JSERVER_PORT=portnumber
```

At each command prompt in the C shell, enter one of these commands:

```
setenv TRAFCI_PERL_SERVER=absolute-path-of-JavaServer.jar
setenv TRAFCI_PYTHON_JSERVER=absolute-path-of-Jython.jar
setenv TRAFCI_PERL_JSERVER_PORT=portnumber
```

Setting Login Environment Variables in the User Profile

To set the login environment variables in the user profile:

1. Open the user profile (`.profile` or `.bash_profile` for the Bash shell) in the `/home` directory. For example:
2. Add these `export` commands (or `setenv` commands for the C shell) to the user profile. For example:

```
vi .profile
```

```
export TRAFCI_PERL_JSERVER=absolute-path-of-JavaServer.jar
export TRAFCI_PYTHON_JSERVER=absolute-path-of-Jython.jar
export TRAFCI_PERL_JSERVER_PORT=portnumber
```

3. To activate the changes, either log out and log in again or execute the user profile. For example:

```
. .profile
```

Perl and Python Wrapper Scripts

The Perl or Python wrapper scripts enable you to run SQL statements and script files using a single connection or multiple connections within Perl or Python programs. The Perl wrapper script is `trafci.pl`, and the Python wrapper script is `trafci.py`. By default, these wrapper scripts are located in the `bin` directory:

- On Windows:
`trafci-installation-directory\Trafodion Command Interface\bin`
`trafci-installation-directory` is the directory where you installed the TrafCI software files.
- On Linux or UNIX:
`trafci-installation-directory/trafci/bin`
`trafci-installation-directory` is the directory where you installed the TrafCI software files.

Launching TrafCI From the Perl or Python Command Line

You can launch the Perl or Python wrapper scripts as shown below:

- Perl Wrapper Script:

```
perl trafci.pl perl-script-filename
```

To run a Perl program, enter the `perl` command at a command prompt. For example:

```
>perl trafci.pl example.pl
```

- Python Wrapper Script:

```
python trafci.py python-script-filename
```

To run a Python program, enter the `python` command at a command prompt. For example:


```
>python trafci.py example.py
```

Example of a Perl Program (sample.pl)

```
use lib 'C:\\Program Files (x86)\\Apache Software Foundation\\Trafodion Command
Interface\\lib\\perl'; use Session;

# create a new session
$sess = Session->new();

# connect to the database
$sess->connect("user1","password","16.123.456.78","37800");

$retval=$sess->execute(" set schema TRAFODION.CI_SAMPLE ");
print $retval;
$retval=$sess->execute("select * from employee");
print $retval;
$retval=$sess->execute("get statistics");
print $retval;

print "\\n\\nSession 1: Disconnecting first session. \\n\\n";
$sess->disconnect();
```

Example of a Python Program (sample.py)

```
import os
import sys
## Modify this path
sys.path.append("C:\\Program Files (x86)\\Apache Software Foundation\\Trafodion Command
Interface\\lib\\python") import Session

# create a new session
sess = Session.Session()

# Connect to the database
x=sess. connect ("user1","password","16.123.456.78","37800")

# Execute sample queries
# execute takes the query string as argument

setSchema      = "set schema TRAFODION.CI_SAMPLE";
selectTable    = "select * from employee"
getStats       = "get statistics"

#Construct a list of SQL statements to be executed
queryList = [setSchema, selectTable, getStats]
print "\\n";

for query in queryList:
    print sess. execute (query)

# disconnect the session
sess. disconnect ()
del sess
sess=None
```

A Interface Commands

TrafCI supports these commands in the command-line interface or in script files that you run from the command-line interface.

Command	Description	Syntax
@	Runs the SQL statements and interface commands contained in a specified script file.	See the “@ Command” (page 51).
/	Runs the previously executed SQL statement.	See the “/ Command” (page 52).
ALIAS	Maps a string to any interface or SQL command.	See the “ALIAS Command” (page 53).
CLEAR	Clears the command console so that only the prompt appears at the top of the screen.	See the “CLEAR Command” (page 54).
CONNECT	Creates a new connection to the Trafodion database from a current or existing TrafCI session.	See the “CONNECT Command” (page 55).
DELAY	Allows the TrafCI session to be in sleep mode for the specified interval.	“DELAY Command” (page 57)
DISCONNECT	Terminates the connection to the Trafodion database.	See the “DISCONNECT Command” (page 58).
ENV	Displays attributes of the current TrafCI session.	See the “ENV Command” (page 59).
EXIT	Disconnects from and exits the command-line interface.	See the “EXIT Command” (page 61).
FC	Edits and reexecutes a previous command. This command is restricted to the command-line interface and is disallowed in script files.	See the “FC Command” (page 62).
GET STATISTICS	Returns formatted statistics for the last executed SQL statement.	See the “GET STATISTICS Command” (page 65).
GOTO	Jumps to a point in the command history specified by the “LABEL Command” (page 72).	See the “GOTO Command” (page 67).
HELP	Displays help text for the interface commands.	See the “LOCALHOST Command” (page 73).
HISTORY	Displays recently executed commands.	See the “HISTORY Command” (page 69).
IF...THEN	Allows the conditional execution of actions specified within the IF...THEN conditional statement.	See the “IF...THEN Command” (page 70).
LABEL	Marks a point in the command history that you can jump to by using the GOTO command.	See the “LABEL Command” (page 72).
LOCALHOST	Executes client machine commands.	See the “LOCALHOST Command” (page 73).
LOG	Logs commands and output from TrafCI to a log file.	See the “LOG Command” (page 74).
OBEY	Runs the SQL statements and interface commands contained in a specified script file.	See the “OBEY Command” (page 77).
PRUN	Runs script files in parallel.	See the “PRUN Command” (page 80).
QUIT	Disconnects from and exits TrafCI.	See the “QUIT Command” (page 83).
RECONNECT	Creates a new connection to the Trafodion database using the login credentials of the last successful connection.	See the “RECONNECT Command” (page 84).
REPEAT	Reexecutes a command.	See the “REPEAT Command” (page 85).
RESET LASTERROR	Resets the last error code to 0.	See the “RESET LASTERROR Command” (page 87).

Command	Description	Syntax
RESET PARAM	Clears all parameter values or a specified parameter value in the current session.	See the “RESET PARAM Command” (page 88) .
RUN	Runs the previously executed SQL statement.	See the “RUN Command” (page 89) .
SAVEHIST	Saves the session history in a user-specified file.	See the “SAVEHIST Command” (page 90) .
SESSION	Displays attributes of the current TrafCI session.	See the “SHOW SESSION Command” (page 129) .
SET COLSEP	Sets the column separator and allows you to control the formatting of the result displayed for SQL queries.	See the “SET COLSEP Command” (page 91) .
SET FETCHSIZE	Changes the default fetchsize used by JDBC.	See the “SET FETCHSIZE Command” (page 92) .
SET HISTOPT	Sets the history option and controls how commands are added to the history buffer.	See the “SET HISTOPT Command” (page 93) .
SET IDLETIMEOUT	Sets the idle timeout value for the current session.	See the “SET IDLETIMEOUT Command” (page 95) .
SET LIST_COUNT	Sets the maximum number of rows to be returned by SELECT statements that are executed after this command.	See the “SET LIST_COUNT Command” (page 96) .
SET MARKUP	Sets the markup format and controls how results are displayed by TrafCI.	See the “SET MARKUP Command” (page 98) .
SET PARAM	Sets a parameter value in the current session.	See the “SET PARAM Command” (page 102) .
SET PROMPT	Sets the prompt of the current session to a specified string or to a session variable.	See the “SET PROMPT Command” (page 104) .
SET SQLPROMPT	Sets the SQL prompt of the current session to a specified string. The default is SQL.	See the “SET SQLPROMPT Command” (page 106) .
SET SQLTERMINATOR	Sets the SQL statement terminator of the current session to a specified string. The default is a semicolon (;).	See the “SET SQLTERMINATOR Command” (page 108) .
SET STATISTICS	Automatically retrieves the statistics information for a query being executed.	See the “SET STATISTICS Command” (page 109) .
SET TIME	Causes the local time of the client workstation to be displayed as part of the interface prompt.	See the “SET TIME Command” (page 110) .
SET TIMING	Causes the elapsed time to be displayed after each SQL statement executes.	See the “SET TIMING Command” (page 111) .
SHOW ACTIVITYCOUNT	Functions as an alias of “SHOW RECCOUNT Command” (page 126) .	See the “SHOW ACTIVITYCOUNT Command” (page 112) .
SHOW ALIAS	Displays all or a set of aliases available in the current TrafCI session.	See the “SHOW ALIAS Command” (page 113) .
SHOW ALIASES	Displays all the aliases available in the current TrafCI session.	See the “SHOW ALIASES Command” (page 114) .
SHOW CATALOG	Displays the current catalog of the TrafCI session.	See the “SHOW CATALOG Command” (page 115) .
SHOW COLSEP	Displays the value of the column separator for the current TrafCI session.	See the “SHOW COLSEP Command” (page 116) .
SHOW ERRORCODE	Functions as an alias for the “SHOW LASTERROR Command” (page 121) .	See the “SHOW ERRORCODE Command” (page 117) .
SHOW FETCHSIZE	Displays the fetch size value for the current TrafCI session.	See the “SHOW FETCHSIZE Command” (page 118) .
SHOW HISTOPT	Displays the value that has been set for the history option of the current setting.	See the “SHOW HISTOPT Command” (page 119) .

Command	Description	Syntax
SHOW IDLETIMEOUT	Displays the idle timeout value of the current session.	See the “SHOW IDLETIMEOUT Command” (page 120) .
SHOW LASTERROR	Displays the last error of the statement that was executed.	See the “SHOW LASTERROR Command” (page 121) .
SHOW LIST_COUNT	Displays the maximum number of rows to be returned by SELECT statements in the current session.	See the “SHOW LIST_COUNT Command” (page 122) .
SHOW MARKUP	Displays the value that has been set for the markup option for the current TrafCI session.	See the “SHOW MARKUP Command” (page 123) .
SHOW PARAM	Displays the parameters that are set in the current session.	See the “SHOW PARAM Command” (page 124) .
SHOW PREPARED	Displays the prepared statements in the current TrafCI session.	See the “SHOW PREPARED Command” (page 125) .
SHOW RECCOUNT	Displays the record count of the previous executed SQL statement.	See the “SHOW RECCOUNT Command” (page 126) .
SHOW REMOTEPROCESS	Displays the process name of the DCS server that is handling the current connection.	See the “SHOW REMOTEPROCESS Command” (page 127) .
SHOW SCHEMA	Displays the current schema of the TrafCI session.	See the “SHOW SCHEMA Command” (page 128) .
SHOW SESSION	Displays attributes of the current TrafCI session.	See the “SHOW SESSION Command” (page 129) .
SHOW SQLPROMPT	Displays the value of the SQL prompt for the current session.	See the “SHOW SQLPROMPT Command” (page 131) .
SHOW SQLTERMINATOR	Displays the SQL statement terminator of the current session.	See the “SHOW SQLTERMINATOR Command” (page 132) .
SHOW STATISTICS	Displays if statistics has been enabled or disabled for the current session	See the “SHOW STATISTICS Command” (page 133) .
SHOW TIME	Displays the setting for the local time in the SQL prompt.	See the “SHOW TIME Command” (page 134) .
SHOW TIMING	Displays the setting for the elapsed time.	See the “SHOW TIMING Command” (page 135) .
SPOOL	Logs commands and output from TrafCI to a log file.	See the “SPOOL Command” (page 136) .
VERSION	Displays the build versions of the platform, database connectivity services, JDBC Type 4 Driver, and TrafCI.	See the “VERSION Command” (page 139) .

@ Command

The @ command executes the SQL statements and interface commands contained in a specified script file. The @ command is executed the same as the OBEY command. For more information on syntax and considerations, see the “OBEY Command” (page 77).

Syntax

```
@{script-file | wild-card-pattern} [(section-name)]
```

script-file

is the name of an ASCII text file that contains SQL statements, interface commands, and comments. If the script file exists outside the local directory where you launch TrafCI (by default, the bin directory), specify the full directory path of the script file.

wild-card-pattern

is a character string used to search for script files with names that match the character string. *wild-card-pattern* matches a string, depending on the operating system for case-sensitivity, unless you enclose it within double quotes. To look for similar values, specify only part of the characters of *wild-card-pattern* combined with these wild-card characters:

*	Use an asterisk (*) to indicate zero or more characters of any type. For example, <i>*art*</i> matches SMART, ARTIFICIAL, and PARTICULAR.
?	Use a question mark (?) to indicate any single character. For example, <i>boo?</i> matches BOOK and BOOT but not BOO or BOOTS.

(*section-name*)

is the name of a section within the *script-file* to execute. If you specify *section-name*, the @ command executes the commands between the header line for the specified section and the header line for the next section (or the end of the script file). If you omit *section-name*, the @ command executes the entire script file. For more information, see “Section Headers” (page 40).

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- Space is disallowed between the @ sign and the first character of the script name.
- For additional considerations, see the “OBEY Command” (page 77).

Examples

- This @ command runs the script file from the local directory (the same directory where you are running TrafCI):
SQL>@ddl.sql
- This @ command runs the script file in the specified directory on a Windows workstation:
SQL>@c:\my_files\ddl.sql
- This @ command runs the script file in the specified directory on a Linux or UNIX workstation:
SQL>@./my_files/ddl.sql

/ Command

The / command executes the previously executed SQL statement. This command does not repeat an interface command.

Syntax

```
/
```

Considerations

- You must enter the command on one line.
- The command does not require an SQL terminator.

Example

This / command executes the previously executed SELECT statement:

```
SQL>select count(*) from persnl.employee;
```

```
(EXPR)
-----
                        62

--- 1 row(s) selected.
```

```
SQL>/
```

```
(EXPR)
-----
                        62

--- 1 row(s) selected.
```

```
SQL>
```

ALIAS Command

The ALIAS command allows you to map a string to any interface or SQL command. The syntax of the interface or SQL command is checked only when the mapped string is executed. This command replaces only the first token of a command string, which allows the rest of the tokens to be treated as parameters.

Syntax

```
ALIAS value AS command SQL-terminator
```

value

is a case-insensitive string without spaces. *Value* cannot be an interface command.

command

is an interface command or SQL command.

SQL-terminator

is the default terminator (;) or a string value defined for the statement terminator by the “[SET SQLTERMINATOR Command](#)” (page 108). For more information, see “[Setting and Showing the SQL Terminator](#)” (page 30).

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- The ALIAS command lasts only for the duration of the session.
- An alias on an alias is not supported.

Examples

- This command creates an alias named .OS to perform the LOCALHOST (LH) command:

```
SQL> ALIAS .OS AS LH;
```
- This command executes the new ALIAS with the ls option:

```
SQL>.OS ls

trafci-perl.pl
trafci-python.py
trafci.cmd
trafci.pl
trafci.py
trafci.sh
```
- This command creates an alias named .GOTO to perform the GOTO command:

```
SQL> ALIAS .GOTO AS GOTO;

SQL> .GOTO mylabel

GOTO statement executed, ignoring all commands until a 'LABEL
MYLABEL' command is encountered.
```
- This command creates an alias named USE to perform the SET SCHEMA operation, uses the alias to set the schema to TRAFODION.USR, and checks the current schema to verify that the alias worked correctly:

```
SQL> ALIAS USE AS "SET SCHEMA";

SQL> USE TRAFODION.USR;

SQL> SHOW SCHEMA
      SCHEMA USR
```

CLEAR Command

The CLEAR command clears the interface window so that only the prompt appears at the top of the window. CLEAR does not clear the log file or reset the settings of the session.

Syntax

```
CLEA
```

Considerations

You must enter the command on one line. The command does not require an SQL terminator.

Example

This CLEAR command clears the interface window:

```
SQL>clear
```

After the CLEAR command executes, the interface window appears with only the prompt showing:

```
SQL>
```


CONNECT Command

The **CONNECT** command creates a new connection to the database from the current or existing TrafCI session.

Syntax

```
CONNECT [username [/password] [@hostname]]
```

username

specifies the user name for logging in to the database platform. If the user name is not specified, TrafCI prompts for the user name. If the user name contains spaces or special characters, such as a period (.), hyphen (-), or underscore (_), put the name within double quotes. For example: "sq.user-1".

password

specifies the password of the user for logging in to the database platform. If the password is not specified, TrafCI prompts for the password. If the password contains spaces or special characters, such as @ or a single quote ('), put the password within double quotes. For example: "Tr@f0d!0n".

hostname

specifies the host name or IP address of the database platform to which you want the client to connect. If the hostname is not specified, the value is automatically used from the current TrafCI session. If TrafCI was invoked with the `-noconnect` launch parameter, you are prompted for a *hostname* value.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If TrafCI was invoked with the `-noconnect` launch parameter, TrafCI prompts you for the values.
- If the user name or password contains space or special characters, you must put the name or password within double quotes.

Examples

- This command creates a new connection to the Trafodion database from the current or existing TrafCI session:

```
SQL>connect
User Name: user1
Password:

Connected to Trafodion
```
- This command creates a new connection to the Trafodion database from the current or existing TrafCI session:

```
SQL>connect user1/password

Connected to Trafodion
```
- This command creates a new connection to the Trafodion database from the current or existing TrafCI session:

```
SQL>connect user1/password@host0101

Connected to Trafodion
```
- This command creates a new connection to the Trafodion database from the current or existing TrafCI session:

```
SQL>connect user2  
Password:
```

```
Connected to Trafodion
```

DELAY Command

The DELAY command allows the TrafCI session to be in sleep mode for the specified interval.

Syntax

```
DELAY time [sec[ond][s] | min[ute][s]]
```

time

is an integer.

Considerations

- If seconds or minutes are not specified, the default is seconds.
- The maximum delay limit is 3600 seconds. You can override this value by setting `trafci.maxDelayLimit` in `_JAVA_OPTIONS`. The unit is seconds for `trafci.maxDelayLimit`.
- This command does not require an SQL terminator.

Examples

- This DELAY command puts the TrafCI session to sleep for 5 seconds before executing the next command:

```
SQL>delay 5 secs
```



```
SQL> show views
```
- This DELAY command puts TrafCI session to sleep for 5 minutes before executing the next command, which is to exit the session:

```
SQL>delay 5 mins
```



```
SQL> exit
```

DISCONNECT Command

The DISCONNECT command terminates the connection from the database, not from TrafCI.

Syntax

```
DISCONNECT [WITH] [status] [IF{condition}]
```

status

is any 1-byte integer. *status* is a shell return value, and the range of allowable values is platform dependent.

condition

is the same as the condition parameter defined for the “IF...THEN Command” (page 70). See “Condition Parameter” (page 70).

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- After you disconnect from the Trafodion database, you can still run these interface commands:

Table 2 Interface Commands That Can Be Run Without a Connection

ALIAS	HELP	SAVEHIST	SET/SHOW SQLTERMINATOR
CLEAR	HISTORY	SESSION	SET/SHOW TIME
CONNECT	LABEL	SET/SHOW COLSEP	SET/SHOW TIMING
DELAY	LOCALHOST	SET/SHOW HISTOPT	SHOW ALIAS/ALIASES
DISCONNECT	LOG	SET/SHOW IDLETIMEOUT	SHOW SESSION
ENV	QUIT	SET/SHOW MARKUP	SPOOL
EXIT	REPEAT	SET/SHOW PARAM	VERSION
FC	RESET LASTERROR	SET PROMPT	
GOTO	RESET PARAM	SET/SHOW SQLPROMPT	

Examples

This command terminates the connection to the Trafodion database. You can connect to the Trafodion database by using the CONNECT and RECONNECT commands:

```
SQL>disconnect
```

```
Session Disconnected. Please connect to the database by using  
connect/reconnect command.
```

ENV Command

ENV displays attributes of the current TrafCI session. You can also use the SESSION and SHOW SESSION commands to perform the same function.

Syntax

EN

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- ENV displays these attributes:

COLSEP	Current column separator, which is used to control how query results are displayed. For more information, see “SET COLSEP Command” (page 91) .
HISTOPT	Current history options, which controls how the commands are added to the history buffer. For more information, see “SET HISTOPT Command” (page 93) .
IDLETIMEOUT	Current idle timeout value, which determines when the session expires after a period of inactivity. By default, the idle timeout is 30 minutes. For more information, see “Setting and Showing the Idle Timeout Value for the Session” (page 30) and “SET IDLETIMEOUT Command” (page 95) .
LIST_COUNT	Current list count, which is the maximum number of rows that can be returned by SELECT statements. By default, the list count is all rows. For more information, see “SET LIST_COUNT Command” (page 96) .
LOG FILE	Current log file and the directory containing the log file. By default, logging during a session is turned off. For more information, see “Logging Output” (page 38) and “LOG Command” (page 74) or “SPOOL Command” (page 136) .
LOG OPTIONS	Current logging options. By default, logging during a session is turned off, and this attribute does not appear in the output. For more information, see the “LOG Command” (page 74) or “SPOOL Command” (page 136) .
MARKUP	Current markup option selected for the session. The default option is RAW. For more information, see “SET MARKUP Command” (page 98) .
PROMPT	Current prompt for the session. For example, the default is SQL>. For more information, see “Customizing the Standard Prompt” (page 30) and “SET PROMPT Command” (page 104) .
SCHEMA	Current schema. The default is USR. For more information, see “Setting and Showing the Current Schema” (page 31) .
SERVER	Host name and port number that you entered when logging in to the database platform. For more information, see “Logging In to the Database Platform” (page 21) .
SQLTERMINATOR	Current SQL statement terminator. The default is a semicolon (;). For more information, see “Setting and Showing the SQL Terminator” (page 30) and “SHOW SQLTERMINATOR Command” (page 132) .
STATISTICS	Current setting (on or off) of statistics. For more information, see the “SET STATISTICS Command” (page 109) .
TIME	Current setting (on or off) of the local time as part of the prompt. When this command is set to on, military time is displayed. By default, the local time is off. For more information, see “Customizing the Standard Prompt” (page 30) and “SET TIME Command” (page 110) .
TIMING	Current setting (on or off) of the elapsed time. By default, the elapsed time is off. For more information, see “Displaying the Elapsed Time” (page 31) and “SET TIMING Command” (page 111) .
USER	User name that you entered when logging in to the database platform. For more information, see “Logging In to the Database Platform” (page 21) .

Examples

- This ENV command displays the attributes of the current session:

```
SQL>env
```

```
COLSEP          " "
HISTOPT         DEFAULT [No expansion of script files]
IDLETIMEOUT     0 min(s) [Never Expires]
LIST_COUNT      0 [All Rows]
LOG_FILE        c:\session.txt
LOG_OPTIONS     APPEND,CMDTEXT ON
MARKUP          RAW
PROMPT          SQL>
SCHEMA          SEABASE
SERVER          sqws135.houston.host.com:378

00 SQLTERMINATOR ;
STATISTICS      OFF
TIME            OFF
TIMING          OFF
USER            user1
```

- This ENV command shows the effect of setting various session attributes:

```
4:16:43 PM >env
```

```
COLSEP          " "
HISTOPT         DEFAULT [No expansion of script files]
IDLETIMEOUT     30 min(s)
LIST_COUNT      0 [All Rows]
LOG             OFF
MARKUP          RAW
PROMPT          SQL>
SCHEMA          SEABASE
SERVER          sqws135.houston.host.com:378

00 SQLTERMINATOR ;
STATISTICS      OFF
TIME            OFF
TIMING          OFF
USER            user1
```

```
4:16:49 PM >
```

EXIT Command

The EXIT command disconnects from and exits TrafCI. EXIT can return a status code. If no status code is specified, zero is returned by default. In addition, a conditional statement can be appended to the command.

Syntax

```
EXIT [WITH] [status] [IF{condition}]
```

status

is any 1-byte integer. *status* is a shell return value, and the range of allowable values is platform dependent.

condition

is the same as the condition parameter defined for the “IF...THEN Command” (page 70). See “Condition Parameter” (page 70).

Considerations

You must enter the command on one line. The command does not require an SQL terminator.

Examples

- This command disconnects from and exits TrafCI, which disappears from the screen:
SQL>exit
- In a script file, the conditional exit command causes the script file to quit running and disconnect from and exit TrafCI when the previously run command returns error code 4082:

```
log c:\errorCode.log
select * from employee;
exit if errorcode=4082
log off
```

These results are logged when error code 4082 occurs:

```
SQL>select * from employee;
```

```
*** ERROR[4082] Table, view or stored procedure TRAFODION.USR.EMPLOYEE
    does not exist or is inaccessible.
```

```
SQL>exit if errorcode=4082
```

- The following two examples are equivalent:

```
SQL> EXIT -1 IF LASTERROR <> 0
SQL> EXIT WITH -1 IF LASTERROR != 0
```

This example exits TrafCI if the last error code is equal to 4082:

```
SQL> EXIT WITH 82 IF LASTERROR == 4082
```

```
SQL> EXIT -- default status is 0
```

FC Command

The FC command allows you to edit and reissue a command in the history buffer of an TrafCI session. You can display the commands in the history buffer by using the HISTORY command. For information about the history buffer, see the [“HISTORY Command” \(page 69\)](#).

Syntax

```
FC [text | [-]number]
```

text

is the beginning text of a command in the history buffer. Case is not significant in matching the text to a command.

[-]*number*

is either a positive integer that is the ordinal number of a command in the history buffer or a negative integer that indicates the position of a command relative to the most recent command.

Without text or number, FC retrieves the most recent command.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- You cannot execute this command in a script file. You can execute this command only at a command prompt.
- As each line of the command is displayed, you can modify the line by entering these editing commands (in uppercase or lowercase letters) on the line below the displayed command line:

D	Deletes the character immediately above the letter D. Repeat to delete more characters.
<i>lcharacters</i>	Inserts characters in front of the character immediately above the letter l.
<i>Rcharacters</i>	Replaces existing characters one-for-one with characters, beginning with the character immediately above the letter R.
<i>characters</i>	Replaces existing characters one-for-one with characters, beginning with the first character immediately above characters. <i>characters</i> must begin with a nonblank character.

To specify more than one editing command on a line, separate the editing commands with a double slash (/). The end of a line terminates an editing command or a set of editing commands.

After you edit a line of the command, TrafCI displays the line again and allows you to edit it again. Press Enter without specifying editing commands to stop editing the line. If that line is the last line of the command, pressing Enter executes the command.

To terminate a command without saving changes to the command, use the double slash (/), and then press Enter.

Examples

- Reexecute the most recent command that begins with SH:

```
SQL>fc sh
SQL>show schema
....
```

Pressing Enter executes the SHOW SCHEMA command and displays the current schema, PERSNL:

```
SQL>fc sh
SQL>show schema
....
```


SCHEMA PERSONNEL

SQL>

- **Correct an SQL statement that you entered incorrectly by using the delete (D) editing command:**

```
SQL>select * from persnl.employee;
```

```
*** ERROR[15001] A syntax error occurred at or before:
select * from persnl.employee;
      ^
```

SQL>fc

```
SQL>select * from persnl.employee;
```

```
....      d
```

```
SQL>select * from persnl.employee;
```

```
....
```

Pressing Enter executes the corrected SELECT statement.

- **Correct an SQL statement that you entered incorrectly by using more than one editing command:**

```
SQL>selt * fromm persnl.employee;
```

```
*** ERROR[15001] A syntax error occurred at or before:
selt * fromm persnl.employee;
      ^
```

SQL>fc

```
SQL>selt * fromm persnl.employee;
```

```
....   iec//  d
```

```
SQL>select * from persnl.employee;
```

```
....
```

Pressing Enter executes the corrected SELECT statement.

- **Modify a previously executed statement by replacing a value in the WHERE clause with another value:**

```
SQL>select first_name, last_name
+>from persnl.employee
+>where jobcode=111;
```

```
--- 0 row(s) selected.
```

SQL>fc

```
SQL>select first_name, last_name
```

```
....
```

```
SQL>from persnl.employee
```

```
....
```

```
SQL>where jobcode=111;
```

```
450
```

```
....
```

```
SQL>where jobcode=450;
```

```
....
```

Pressing Enter lists the first and last names of all of the employees whose job code is 450.

- **Modify a previously executed statement by replacing a column name in the select list with another column name:**

```
SQL>select first_name, last_name
+>from persnl.employee
+>where jobcode=450;
```

FIRST_NAME	LAST_NAME
MANFRED	CONRAD
WALTER	LANCASTER
JOHN	JONES
KARL	HELMSTED
THOMAS	SPINNER

```
--- 5 row(s) selected.
```

```
SQL>fc
```

```
SQL>select first_name, last_name
```

```
....      R      empnum,
```

```
SQL>select      empnum, last_name
```

```
....
```

```
SQL>from persnl.employee
```

```
....
```

```
SQL>where jobcode=450;
```

```
....
```

Pressing Enter lists the employee number and last names of all employees whose job code is 450:

```
EMPNUM  LAST_NAME
```

```
-----
```

```
180  CONRAD
```

```
215  LANCASTER
```

```
216  JONES
```

```
225  HELMSTED
```

```
232  SPINNER
```

```
--- 5 row(s) selected.
```

```
SQL>
```

GET STATISTICS Command

The GET STATISTICS command returns formatted statistics for the last executed SQL statement.

Syntax

```
GET STATISTICS
```

Description of Returned Values:

Records Accessed

number of rows returned by disk process to EID (Executor In Disk process).

Records Used

number of rows returned by EID after selection.

Disk IOs

number of actual disk IOs done by disk process.

Message Count

number of messages sent/received between filesystem and disk process.

Message Bytes

number of message bytes sent/received between filesystem and disk process.

Lock Escl

number of lock escalations.

Lock Wait

number of lock waits.

Disk Process Busy Time

cpu time for disk process processes for the specified table.

Considerations

The command requires an SQL terminator.

Examples

```
SQL>select * from job;
```

```
JOBCODE JOBDESC
```

```
-----
```

```
100 MANAGER
1234 ENGINEER
450 PROGRAMMER
900 SECRETARY
300 SALESREP
500 ACCOUNTANT
400 SYSTEM ANALYST
250 ASSEMBLER
420 ENGINEER
600 ADMINISTRATOR
200 PRODUCTION SUPV
```

```
--- 11 row(s)
```

```
selected. SQL> get
```

```
statistics;          2105/04/18 21:45:34.082329
End Time             2105/04/18 21:45:34.300265
Elapsed Time         00:00:00.217936
Compile Time         00:00:00.002423
Execution Time       00:00:00.218750
```

Table Name	Records	Records	Disk	Message	Message	Lock	Lock	Disk Process
------------	---------	---------	------	---------	---------	------	------	--------------

	Accessed	Used	I/Os	Count	Bytes	Escl	Wait	Busy Time
TRAFODION.TOI.JOB	2	2	0	4	15232	0	0	363

--- SQL operation complete.

GOTO Command

The GOTO command allows you to jump to a designated point in the command history. The point in the command history is designated by a LABEL command. All commands executed after a GOTO statement are ignored until the specified label is set. To set a label, use the [“LABEL Command”](#) (page 72).

Syntax

```
GOTO {label}
```

label

is a string of characters without quotes and spaces, or a quoted string.

Considerations

- You must enter the command on one line.
- The GOTO command cannot currently jump back in the command history; it is a forward-only command.

Examples

These examples show the use of the GOTO and LABEL commands:

```
SQL> GOTO ViewManagers
SQL> SELECT * FROM Employees; -- skipped
SQL> SHOW RECCOUNT;           -- skipped
SQL> LABEL ViewManagers
SQL> SELECT * FROM Managers;
SQL> GOTO "View Customers"
SQL> SELECT * FROM Invoices;  -- skipped
SQL> LABEL "View Customers"
SQL> SELECT * FROM Customers;
```

HELP Command

The HELP command displays help text for the interface commands. See [Appendix A \(page 48\)](#) for descriptions of the interface commands.

Syntax

```
HELP [command-name]
```

command-name

is the name of an interface command. If you do not specify a command, TrafCI returns a list of all interface commands. If you specify SET, TrafCI returns a list of all SET commands. If you specify SHOW, TrafCI returns a list of all SHOW commands.

Considerations

You must enter the command on one line. The command does not require an SQL terminator.

Examples

- This HELP command lists all the interface commands that are supported:

```
SQL>help
```
- This HELP command lists all the SET commands that are supported:

```
SQL>help set
```
- This HELP command lists all the SHOW commands that are supported:

```
SQL>help show
```
- This HELP command shows help text for SET IDLETIMEOUT:

```
SQL>help set idletimeout
```

HISTORY Command

The HISTORY command displays recently executed commands, identifying each command by a number that you can use to reexecute or edit the command.

Syntax

```
HISTORY [number]
```

number

is the number of commands to display. The default number is 10. The maximum number is 100.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- You can use the FC command to edit and reexecute a command in the history buffer, or use the REPEAT command to reexecute a command without modifying it. See the [“FC Command” \(page 62\)](#) or the [“REPEAT Command” \(page 85\)](#).

Example

Display the three most recent commands and use FC to redisplay one:

```
SQL>history 3
14>      set schema sales;
15>      show tables
16>      show views
```

```
SQL>fc 14
SQL>set schema sales
....
```

Now you can use the edit capabilities of FC to modify and execute a different SET SCHEMA statement.

IF...THEN Command

IF...THEN statements allow for the conditional execution of actions. If the condition is met, the action is executed; otherwise, no action is taken.

Syntax

```
IF {condition} THEN {action} {SQL-terminator}
```

Condition Parameter

The condition parameter (*condition*) is a Boolean statement structured as follows:

```
{variable-name|value}{operator}{variable-name|value}
```

variable-name

is one of:

```
{ LASTERROR  
| RECCOUNT  
| ACTIVITYCOUNT  
| ERRORCODE  
| [%]any ENV variable|any SQL parameter }
```

value

is any integer or a quoted string, where the quoted string is any non-quote character. \ is the optional escape character.

operator

is one of:

Operator	Meaning
== =	equal to
<> != ~= ^=	not equal to
>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to

Action Parameter

The action parameter (*action*) is any interface or SQL command.

SQL Terminator

The SQL terminator (*SQL-terminator*) is the default terminator (;) or a string value defined for the statement terminator by the “[SET SQLTERMINATOR Command](#)” (page 108). See “[Setting and Showing the SQL Terminator](#)” (page 30).

Considerations

- IF...THEN is itself an action. Thus, nested IF...THEN statements are allowed.
- An action must end with the SQL terminator, even if the action is an interface command.

Examples

These commands show multiple examples of IF...THEN statements:

```
SQL> INVOKE Employees

SQL> -- ERROR 4082 means the table does not exist

SQL> IF ERRORCODE != 4082 THEN GOTO BeginPrepare

SQL> CREATE TABLE Employees(SSN INT PRIMARY KEY NOT NULL NOT DROPPABLE, FName
VARCHAR(50),
LName VARCHAR(50), HireDate DATE DEFAULT CURRENT_DATE);

SQL> LABEL BeginPrepare

SQL> PREPARE empSelect FROM
+> SELECT * FROM
+> Employees
+> WHERE SSN=?empSSN;

SQL> IF USER == "alice" THEN SET PARAM ?empSSN 987654321;

SQL> IF %USER == "bob" THEN SET PARAM ?empSSN 123456789;

SQL> execute empSelect

SQL> IF USER == "alice" THEN
+> IF ACTIVITYCOUNT == 0 THEN GOTO insertAlice;

SQL> IF USER == "bob" THEN IF ACTIVITYCOUNT == 0 THEN GOTO insertBob;

SQL> EXIT

SQL> LABEL insertAlice

SQL> INSERT INTO Employees(SSN, FName, LName) VALUES(987654321, 'Alice',
'Smith');

SQL> EXIT

SQL> LABEL insertBob

SQL> INSERT INTO Employees(SSN, FName, LName) VALUES(123456789, 'Bob',
'Smith');

SQL> EXIT
```

LABEL Command

The LABEL command marks a point in the command history that you can jump to by using the GOTO command. For more information, see the [“GOTO Command” \(page 67\)](#).

Syntax

```
LABEL {label}
```

label

is a string of characters without quotes and spaces, or a quoted string.

Considerations

You must enter the command on one line.

Examples

This command creates a label using a string of characters:

```
SQL> LABEL MyNewLabel
```

This command creates a label using a quoted string:

```
SQL> LABEL "Trafodion Label"
```

LOCALHOST Command

The LOCALHOST command allows you to execute client machine commands.

Syntax

```
LOCALHOST | LH <client m/c commands>
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- The LOCALHOST command has a limitation. When input is entered for the operating system commands (for example, date, time, and cmd), the input is not visible until you hit the `enter` key.
- If the SET TIMING is set to ON, the elapsed time information is displayed.

Examples

- If you are using a Windows system, `dir` lists the contents of the directory name. Similarly, if you are on a UNIX system you enter `LOCALHOST LS` to display the contents of the folder.

```
SQL>LOCALHOST dir
Volume in drive C is E-Client
Volume Serial Number is DC4F-5B3B

Directory of c:\Program Files (x86)\Apache Software Foundation\Trafodion Command
Interface\bin 05/11/2105 01:17 PM <DIR>
05/11/2105 01:17 PM <DIR>
05/16/2105 09:47 AM          1,042 trafci-perl.pl
05/16/2105 09:47 AM          1,017 trafci-python.pl
05/16/2105 09:47 AM           752 trafci.cmd
05/16/2105 09:47 AM          1,416 trafci.pl
05/16/2105 09:47 AM          2,388 trafci.py
05/16/2105 09:47 AM          3,003 trafci.sh
        6 File(s)      19,491 bytes
        2 Dir (s) 57,686,646,784 bytes free

SQL> LH mkdir c:\trafci -> Will create a directory c:\trafci on your
local machine
```

- This command displays the elapsed time information because the SET TIMING command is set to ON:

```
SQL>set timing on

SQL>localhost ls
trafci-perl.pl
trafci-python.py
trafci.cmd
trafci.pl
trafci.py
trafci.sh

Elapsed :00:00:00.078
```

LOG Command

The LOG command logs the entered commands and their output from TrafCI to a log file. If this is an obey script file, then the command text from the obey script file is shown on the console.

Syntax

```
LOG { ON [CLEAR, QUIET, CMDTEXT {ON | OFF}]  
    | log-file [CLEAR, QUIET, CMDTEXT {ON | OFF}]  
    | OFF }
```

ON

starts the logging process and records information in the `sqlspool.lst` file in the `bin` directory.

CLEAR

instructs TrafCI to clear the contents of the `sqlspool.lst` file before logging new information to the file.

QUIET

specifies that the command text is displayed on the screen, but the results of the command are written only to the log file and not to the screen.

CMDTEXT ON

specifies that the command text and the log header are displayed in the log file.

CMDTEXT OFF

specifies that the command text and the log header are not displayed in the log file.

log-file

is the name of a log file into which TrafCI records the entered commands and their output. If you want the log file to exist outside the local directory where you launch TrafCI (by default, the `bin` directory), specify the full directory path of the log file. The log file does not need to exist, but the specified directory must exist before you execute the LOG command.

log-file CLEAR

instructs TrafCI to clear the contents of the specified *log-file* before logging new information to the file.

OFF

stops the logging process.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- Use a unique name for each log file to avoid writing information from different TrafCI sessions into the same log file.

Examples

- This command starts the logging process and records information to the `sqlspool.lst` file in the `bin` directory:
`SQL>log on`
- This command starts the logging process and appends new information to an existing log file, `persnl_updates.log`, in the local directory (the same directory where you are running TrafCI):
`SQL>log persnl_updates.log`
- This command starts the logging process and appends new information to a logfile, `sales_updates.log`, in the specified directory on a Windows workstation:
`SQL>log c:\log_files\sales_updates.log`

- This command starts the logging process and appends new information to a log file, sales_updates.log, in the specified directory on a Linux or UNIX workstation:
SQL>log ./log_files/sales_updates.log
- This command starts the logging process and clears existing information from the log file before logging new information to the file:
SQL>log persnl_ddl.log clear
- This command start the logging process, clears existing information from the log file, and specifies that the command text and log header is not displayed in the log file:
SQL>log c:\temp\a.txt clear, cmdtext off

```
SQL>select * from trafodion.toi.job
+>;
```

```
JOBCODE  JOBDESC
-----  -
      100  MANAGER
      450  PROGRAMMER
      900  SECRETARY
     300  SALESREP
     500  ACCOUNTANT
     400  SYSTEM ANALYST
     250  ASSEMBLER
     420  ENGINEER
     600  ADMINISTRATOR
     200  PRODUCTION SUPV
```

```
--- 10 row(s) selected.
```

```
SQL> log off
```

```
Output of c:\temp\a.txt
=====
```

```
JOBCODE  JOBDESC
-----  -
      100  MANAGER
      450  PROGRAMMER
      900  SECRETARY
     300  SALESREP
     500  ACCOUNTANT
     400  SYSTEM ANALYST
     250  ASSEMBLER
     420  ENGINEER
     600  ADMINISTRATOR
     200  PRODUCTION SUPV
```

```
--- 10 row(s) selected
```

- This command start the logging process, clears existing information from the log file, specifies that no output appears on the console window, and the quiet option is enabled:

```
SQL>log c:\temp\b.txt clear, cmdtext off, quiet
```

```
SQL>select *
+>from trafodion.toi.job;
```

```
SQL> log off
```

```
Output of c:\temp\b.txt
=====
```

```
JOBCODE  JOBDESC
-----  -
      100  MANAGER
```

```
450 PROGRAMMER
900 SECRETARY
300 SALESREP
500 ACCOUNTANT
400 SYSTEM ANALYST
250 ASSEMBLER
420 ENGINEER
600 ADMINISTRATOR
200 PRODUCTION SUPV
```

```
--- 10 row(s) selected
```

This command stops the logging process:

```
SQL>log off
```

For more information, see [“Logging Output” \(page 38\)](#).

OBEY Command

The OBEY command executes the SQL statements and interface commands of a specified script file or an entire directory. This command accepts a single filename or a filename with a wild-card pattern specified. Executing the OBEY command without optional parameters prompts you to enter a filename. If a filename is not specified, then *.sql is used.

Syntax

```
OBEY {script-file | wild-card-pattern} [(section-name)]
```

script-file

is the name of an ASCII text file that contains SQL statements, interface commands, and comments. If the script file exists outside the local directory where you launch TrafCI (by default, the `bin` directory), specify the full directory path of the script file.

wild-card-pattern

is a character string used to search for script files with names that match the character string. *wild-card-pattern* matches a string, depending on the operating system for case-sensitivity, unless you enclose it within double quotes. To look for similar values, specify only part of the characters of *wild-card-pattern* combined with these wild-card characters:

*	Use an asterisk (*) to indicate zero or more characters of any type. For example, <code>*art*</code> matches <code>SMART</code> , <code>ARTIFICIAL</code> , and <code>PARTICULAR</code> .
?	Use a question mark (?) to indicate any single character. For example, <code>boo?</code> matches <code>BOOK</code> and <code>BOOT</code> but not <code>BOO</code> or <code>BOOTS</code> .

(*section-name*)

is the name of a section within the *script-file* to execute. If you specify *section-name*, the OBEY command executes the commands between the header line for the specified section and the header line for the next section (or the end of the script file). If you omit *section-name*, the OBEY command executes the entire script file. For more information, see [“Section Headers” \(page 40\)](#).

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- Put a space between OBEY and the first character of the file name.
- You can execute this command in a script file.
- Before putting dependent SQL statements across multiple files, consider the order of the file execution. If a directory is not passed to the OBEY command, the file or wild card is assumed to be in the current working directory.
- If the (*) is issued in the OBEY command, all files are executed in the current directory. Some of the files in the directory could be binary files. The OBEY command tries to read those binary files and junk or invalid characters are displayed on the console. For example, this command causes invalid characters to be displayed on the console:

```
SQL> obey C:\trafci\bin\*
```

- OBEY detects recursive obey files (for example, an sql file that calls OBEY on itself) and prevents infinite loops using a max depth environment variable. If no variable is passed to the JVM, the default depth is set to 10. To change this depth (for example to a value of 20), pass a Java environment variable as follows:

```
-Dtrafci.obeydepth=20
```

Examples

- This OBEY command runs the script file from the local directory (the same directory where you are running TrafCI):

```
SQL>obey ddl.sql
```

- This OBEY command runs the script file in the specified directory on Windows.

```
SQL>obey c:\my_files\ddl.sql
```

- This OBEY command runs the script file in the specified directory on a Linux or UNIX workstation:

```
SQL>obey ./my_files/ddl.sql
```

- This sample file contains sections to be used in conjunction with the OBEY command:

```
?section droptable
DROP TABLE COURSE
```

```
?section create
CREATE TABLE COURSE
(
  CNO          VARCHAR(3)          NOT NULL,
  CNAME        VARCHAR(22)         NOT NULL,
  CDESCP       VARCHAR(25)         NOT NULL,
  CRED         INT,
  CLABFEE      NUMERIC(5,2),
  CDEPT        VARCHAR(4)          NOT NULL,
  primary key (cno)
) ;
```

```
?section insert
INSERT INTO COURSE VALUES
  ('C11', 'INTRO TO CS','FOR ROOKIES',3, 100, 'CIS');

INSERT INTO COURSE VALUES
  ('C22', 'DATA STRUCTURES','VERY USEFUL',3, 50, 'CIS');
```

```
INSERT INTO COURSE VALUES
  ('C33', 'DISCRETE MATHEMATICS',
    'ABSOLUTELY NECESSARY',3, 0,'CIS');
```

```
?section select
SELECT * FROM course;
```

```
?section delete
purgedata course;
```

To run only the commands in section create, execute the following :

```
SQL>obey C:\Command Interfaces\course.sql (create)
```

```
SQL>?section create
```

```
SQL>CREATE TABLE COURSE
+>(
+>  CNO          VARCHAR(3)          NOT NULL,
+>  CNAME        VARCHAR(22)         NOT NULL,
+>  CDESCP       VARCHAR(25)         NOT NULL,
+>  CRED         INT,
+>  CLABFEE      NUMERIC(5,2),
+>  CDEPT        VARCHAR(4)          NOT NULL,
+>  primary key (cno)
+>) ;
```


--- SQL Operation complete.

To run only the commands in the insert section, execute the following :

```
SQL>obey C:\Command Interfaces\course.sql (insert)
SQL>?section insert
```

```
SQL>INSERT INTO COURSE VALUES
+>    ('C11', 'INTRO TO CS','FOR ROOKIES',3, 100, 'CIS');
```

--- 1 row(s) inserted.

```
SQL>INSERT INTO COURSE VALUES
+>    ('C22', 'DATA STRUCTURES','VERY USEFUL',3, 50, 'CIS');
```

--- 1 row(s) inserted.

```
SQL>INSERT INTO COURSE VALUES
+>    ('C33', 'DISCRETE MATHEMATICS',
      'ABSOLUTELY NECESSARY',3, 0, 'CIS');
```

--- 1 row(s) inserted.

This command executes all files with .sql extension:

```
SQL> OBEY c:\trafci\*.sql;
SQL> OBEY c:\trafci
```

This command executes all files beginning with the word “script” and contains one character after the word script and ends with .sql extension. For example:

script1.sql, script2.sql, scriptZ.sql and so on:

```
SQL> OBEY C:\trafci\script?.sql
```

This command executes all files that contain the word “test”. This includes the files that do not end with .sql extension

```
SQL> OBEY C:\trafci\*test*
```

This command executes all files that begin with the word “script” and contains one character after the word “script” and ends with an extension prefixed by a dot. For example:

script1.sql, script2.bat, scriptZ.txt and so on.

```
SQL> OBEY C:\trafci\script?.*
```

This command executes all files that have .txt extension in the current directory, the directory in which the command interface was launched.

```
SQL> OBEY *.txt;
```

This command prompts the user to enter the script filename or a pattern. The default value *.sql

```
SQL> OBEY;
Enter the script filename [*.sql]:
```

PRUN Command

The PRUN command runs script files in parallel.

Syntax

```
PRUN {-d | -defaults} ] |  
[ {- | -scriptsdir} scripts-  
PRUN sd | directory]  
[ {-e | -extension} file-extension]  
[ {- | -logsdir} log-directory]  
[ ld | -overwrite} {y | n}]
```

-d | -defaults

Specify this option to have PRUN use these default settings:

Table 3 PRUN Default Settings

Parameter	Default Setting
-sd -scriptsdir	PRUN searches for the script files in the same directory as the <code>trafci.sh</code> or <code>trafci.cmd</code> file (<code>trafci-installation-directory/trafci/bin</code> or <code>trafci-installation-directory\trafci\bin</code>).
-e -extension	The file extension is <code>.sql</code> .
-ld -logsdir	PRUN places the log files in the same directory as the script files.
-o -overwrite	No overwriting occurs. PRUN keeps the original information in the log files and appends new information at the end of each file.
-c -connections	PRUN uses two connections.

{-sd | -scriptsdir} *scripts-directory*

In this directory, PRUN processes every file with the specified file extension. If you do not specify a directory or if you specify an invalid directory, an error message occurs, and you are prompted to reenter the directory. Before running PRUN, verify that this directory contains valid script files.

{-e | -extension} *file-extension*

Specify the file extension of the script files. The default is `.sql`.

{-ld | -logsdir} *log-directory*

In this directory, PRUN creates a log file for each script file by appending the `.log` extension to the name of the script file. If you do not specify a log file directory, PRUN places the log files in the same directory as the script files.

{-o | -overwrite} {y | n}

If you specify `y`, PRUN overwrites the contents of existing log files. By default, PRUN keeps the original information in the log files and appends new information at the end of each file.

{-c | -connections} *num*

Enter a number for the maximum number of connections. If you do not specify the maximum number of connections, PRUN uses two connections.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If you execute the PRUN command without any arguments, TrafCI prompts you for the PRUN arguments. If you specify one or more options, the PRUN command runs without prompting you for more input. In the non-interactive mode, if any options are not specified, PRUN uses the default values.
- The `-d` or `-defaults` option cannot be specified with any other option.

- The PRUN log files also contain the log end time.
- PRUN does not support the SPOOL or LOG commands. Those commands are ignored in PRUN script files.
- The environment values from the main session (which are available through the SET commands) are propagated to new sessions started via PRUN. However, prepared statements and parameters are bound only to the main user session.
- For a summary of all errors and warnings that occurred during the PRUN operation, go to the `error` subdirectory in the same directory as the log files (for example, `C:\log\error`) and open the `prun.err.log` summary file.
- For details about the errors that occurred during the execution of a script file, open each individual log file (`script-file.sql.log`).

Examples

- To use PRUN, enter the PRUN command in the TrafCI session:

```
SQL>prun
Enter * as input to stop the current prun session
-----

Enter the scripts directory      :      c:\ddl_scripts
Enter the script file extension[sql] :
Enter the logs directory[scripts dir] :      c:\log
Overwrite the log files (y/n)[n]?   :      y
Enter the number of connections(2-248)[2]:      3
```

After you enter the number of connections, PRUN starts to process the script files and displays this status:

```
Status: In Progress.....
```

After executing all the script files, PRUN returns a summary of the operation:

PARALLELRUN (PRUN) SUMMARY	
Total files present.....	3
Total files processed.....	3
Total queries processed.....	40
Total errors.....	4
Total warnings.....	0
Total successes.....	36
Total connections.....	5
Total connection failures.....	0

Please verify the error log file `c:\log\error\prun.err.log`

```
SQL>
```

NOTE: In the PRUN summary, the `Total queries processed` is the total number of commands that PRUN processes. Those commands can include SQL statements and interface commands. The total errors, warnings, and successes also include commands other than SQL statements.

- This PRUN command initiates a parallel run operation with the `-d` option:

```
SQL>prun -d

SQL> prun -scriptsdir ./prun/sql -e sql -ld ./prun/logs -o y -connections 5

PRUN options are -scriptsdir      c:/_trafci/prun
                  -logsdir        c:/_trafci/prun/logs
                  -extension       sql
                  -overwrite       y
                  -connections     5

Status: Complete
```

PARALLELRUN (PRUN) SUMMARY

Total files present.....	99
Total files processed.....	99
Total queries processed.....	198
Total errors.....	0
Total warnings.....	0
Total warnings.....	0
Total connections.....	5
Total connection failures.....	0

=====

PRUN completed at May 20, 2105 9:33:21 AM

=====

- PRUN can be started in non-interactive mode using the `-q` parameter of `trafci.cmd` or `trafci.sh`, thus requiring no input:

```
trafci.cmd -h 16.123.456.78
-u user1 -p host1
-q "prun -sd c:/_trafci/prun -o y -c 3"
```

- PRUN can be started in non-interactive mode from an OBEY file:

```
SQL>obey startPrun.txt
```

```
SQL>prun -sd c:/_trafci/prun -ld c:/_trafci/prun/logs -e sql -o y -c 5
```

```
PRUN options are -scriptsdir    c:/_trafci/prun
                  -logsdire      c:/_trafci/prun/logs
                  -extension     sql
                  -overwrite     yes
                  -connections   5
```

```
Status: Complete
```

QUIT Command

The QUIT command disconnects from and exits TrafCI.

Syntax

```
QUIT [WITH] [status] [IF{condition}]
```

status

is any 1-byte integer. *status* is a shell return value, and the range of allowable values is platform dependent.

condition

is the same as the condition parameter defined for the “IF...THEN Command” (page 70). See “Condition Parameter” (page 70).

Considerations

You must enter the command on one line. The command does not require an SQL terminator.

Examples

- This command disconnects from and exits TrafCI, which disappears from the screen:
SQL>quit
- In a script file, the conditional exit command causes the script file to quit running and disconnect from and exit TrafCI when the previously run command returns error code 4082:

```
log c:\errorCode.log
select * from employee;
quit if errorcode=4082
log off
```

These results are logged when error code 4082 occurs:

```
SQL>select * from employee;

*** ERROR[4082] Table, view or stored procedure TRAFODION.USR.EMPLOYEE
    does not exist or is inaccessible.
SQL>quit if errorcode=4082
```

RECONNECT Command

The RECONNECT command creates a new connection to the Trafodion database using the login credentials of the last successful connection.

Syntax

RECONNEC

Considerations

The host name (or IP address) and port number, plus the credentials (user name and password), are used from information previously entered. This is the information specified at launch or when the last CONNECT command was executed.

If TrafCI was invoked with the `-noconnect` launch parameter, TrafCI prompts you for the values.

Examples

This command creates a new connection to the Trafodion database using the login credentials of the last successful connection:

```
SQL>reconnect
```

```
Connected to Trafodion
```

REPEAT Command

The REPEAT command reexecutes a previous command.

Syntax

```
REPEAT [ text | [-]number ]
```

text

specifies the text of the most recently executed command. The command must have been executed beginning with *text*, but *text* need be only as many characters as necessary to identify the command. TrafCI ignores leading blanks.

[-]*number*

is an integer that identifies a command in the history buffer. If number is negative, it indicates the position of the command in the history buffer relative to the current command; if number is positive, it is the ordinal number of a command in the history buffer.

The HISTORY command displays the commands or statements in the history buffer. See the [“HISTORY Command” \(page 69\)](#).

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- To reexecute the immediately preceding command, enter REPEAT without specifying a number. If you enter more than one command on a line, the REPEAT command reexecutes only the last command on the line.
- When a command is selected for repeat, and the SQL terminator value has changed since the execution of that command, TrafCI replaces the SQL terminator in the command with the current SQL terminator value and executes the command.

Examples

- Display the previously executed commands and reexecute the second to the last command:

```
SQL>history
1>      set idletimeout 0
2>      log on
3>      set schema persnl;
4>      select * from employee;
5>      show tables
6>      select * from dept;
7>      show views
8>      select * from emplist;
```

```
SQL>
```

```
SQL>repeat -2
show views
```

```
VIEW NAMES
```

```
-----
EMPLIST  MGRLIST
```

```
SQL>
```

- Reexecute the fifth command in the history buffer:

```
SQL>repeat 5
show tables
```

```
TABLE NAMES
```

```
-----
DEPT      EMPLOYEE  JOB      PROJECT
```

SQL>

- Reexecute the SHOW TABLES command:

```
SQL>repeat show  
show tables
```

TABLE NAMES

DEPT	EMPLOYEE	JOB	PROJECT
------	----------	-----	---------

SQL>

RESET LASTERROR Command

The RESET LASTERROR command resets the last error code to 0.

Syntax

```
RESET LASTERROR
```

Considerations

You must enter the command on one line. The command does not require an SQL terminator.

Examples

This command resets the last error in the current session:

```
SQL>select * from emp;  
*** ERROR[4082]Object TRAFODION.SCH.EMP does not exist or is inaccessible.
```

```
SQL>show lasterror  
LASTERROR 4082
```

```
SQL> reset lasterror
```

```
SQL>show lasterror  
LASTERROR 0
```

RESET PARAM Command

The RESET PARAM command clears all parameter values or a specified parameter value in the current session.

Syntax

```
RESET PARAM [param-name]
```

param-name

is the name of the parameter for which you specified a value. Parameter names are case-sensitive. For example, the parameter ?pn is not equivalent to the parameter ?PN.

param-name can be preceded by a question mark (?), such as ?*param-name*.

If you do not specify a parameter name, all of the parameter values in the current session are cleared.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- To clear several parameter values but not all, you must use a separate RESET PARAM command for each parameter.

Example

This command clears the setting of the ?sal (salary) parameter, and the SET PARAM command resets it to a new value:

```
SQL>reset param ?sal
```

```
SQL>set param ?sal 80000.00
```

For more information, see [“Resetting the Parameters” \(page 36\)](#).

RUN Command

The RUN command executes the previously executed SQL statement. This command does not repeat an interface command.

Syntax

```
RU
```

Considerations

- You must enter the command on one line.
- The command does not require an SQL terminator.

Example

This command executes the previously executed SELECT statement:

```
SQL>select count(*) from persnl.employee;
```

```
(EXPR)
-----
                62

--- 1 row(s) selected.
```

```
SQL>run
```

```
(EXPR)
-----
                62

--- 1 row(s) selected.
```

```
SQL>
```

SAVEHIST Command

The SAVEHIST command saves the session history in a user-specified file. The session history consists of a list of the commands that were executed in the TrafCI session before the SAVEHIST command.

Syntax

```
SAVEHIST file-name [CLEAR]
```

file-name

is the name of a file into which TrafCI stores the session history. If you want the history file to exist outside the local directory where you launch TrafCI (by default, the `bin` directory), specify the full directory path of the history file. The specified directory must exist before you execute the SAVEHIST command.

CLEAR

instructs TrafCI to clear the contents of the specified file before adding the session history to the file.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the specified file already exists, TrafCI appends newer session-history information to the file.

Examples

- This command clears the contents of an existing file named `history.txt` in the local directory (the same directory where you are running TrafCI) and saves the session history in the file:

```
SQL>savehist history.txt clear
```



```
SQL>
```
- This command saves the session history in a file named `hist.txt` in the specified directory on a Windows workstation:

```
SQL>savehist c:\log_files\hist.txt
```



```
SQL>
```
- This command saves the session history in a file named `hist.txt` in the specified directory on a Linux or UNIX workstation:

```
SQL>savehist ./log_files/hist.txt
```



```
SQL>
```

For more information, see [“Displaying Executed Commands”](#) (page 32).

SET COLSEP Command

The SET COLSEP command sets the column separator and allows you to control the formatting of the result displayed for SQL queries. The SET COLSEP command specifies a delimiter value to use for separating columns in each row of the results. The default delimiter is " "(white space).

Syntax

```
SET COLSEP [separator]
```

Considerations

- You must enter the command on one line.
- The SET COLSEP command has no effect if the markup is set to HTML, XML, or CSV.

Examples

- This command specifies the separator as a "|" (pipe):

```
SQL>set colsep |
```

```
SQL>show colsep  
COLSEP "|"
```

```
SQL>select * from employee;  
EMPNUM|EMPNAME          |REGNUM|BRANCHNUM|JOB  
-----|-----|-----|-----|-----  
      1|ROGER GREEN          |    99|          1|MANAGER  
     23|JERRY HOWARD          |     2|          1|MANAGER  
     29|JACK RAYMOND          |     1|          1|MANAGER  
     32|THOMAS RUDLOFF        |     5|          3|MANAGER  
     39|KLAUS SAFFERT         |     5|          2|MANAGER  
  
--- 5 row(s) selected.
```

SET FETCHSIZE Command

The SET FETCHSIZE command allows you to change the default fetchsize used by JDBC. Setting the value to 0 sets the fetchsize to the default value used in JDBC.

Syntax

```
SET FETCHSIZE value
```

value

is an integer representing the fetch size as a number of rows. Zero (0) represents the default value of fetch size set in JDBC.

Considerations

- You must enter the command on one line.
- The command does not require an SQL terminator.

Examples

This command sets the fetchsize to 1:

```
SQL>SET fetchsize 1
```

```
SQL>SHOW fetchsize
```

```
FETCHSIZE 1
```

```
SQL>select * from stream(t1);
```

C1	C2	C3
TEST1	TEST2	TEST3
AAA	BBB	CCC

SET HISTOPT Command

The SET HISTOPT command sets the history option and controls how commands are added to the history buffer. By default, commands within a script file are not added to history. If the history option is set to "ALL," all the commands in the script file are added to the history buffer. If no options are specified, DEFAULT is used.

Syntax

```
SET HISTOPT [ALL|DEFAULT]
```

Considerations

You must enter the command on one line.

Examples

This command shows only the obey commands added to the history buffer.

```
SQL> show histopt
HISTOPT DEFAULT [No expansion of script files]

SQL> obey e:\scripts\nobey\insert2.sql

SQL> ?section insert

SQL> set schema trafodion.sch;

--- SQL operation complete.

SQL> INSERT INTO COURSE1 VALUES
+>   ('C11', 'INTRO TO CS','FOR ROOKIES',3, 100,'CIS');

--- 1 row(s) inserted.

SQL> INSERT INTO COURSE1 VALUES
+>   ('C55', 'COMPUTER ARCH.','VON NEUMANN''S MACH.',3, 100, 'CIS');

--- 1 row(s) inserted.

SQL> history;
1>      show histopt
2>      obey e:\scripts\nobey\insert2.sql
```

This command shows all the commands added to the history buffer.

```
SQL> set histopt all

SQL> obey e:\scripts\nobey\insert2.sql

?section insert

SQL> set schema trafodion.sch;

--- SQL operation complete.

SQL>      INSERT INTO COURSE1 VALUES
+>      ('C11','INTRO TO CS','FOR ROOKIES',3, 100, 'CIS');

---1 row(s) inserted.

SQL> INSERT INTO COURSE1 VALUES
+>   ('C55', 'COMPUTER ARCH.','VON NEUMANN''S MACH.',3,100,
'CIS');
```

---1 row(s) inserted.

```
SQL> history;
1>     show histopt
2>     obey e:\scripts\nobey\insert2.sql
3>     history;
4>     set histopt all
5>     set schema trafodion.sch;
6>     INSERT INTO COURSE1 VALUES
      ('C11','INTRO TO CS','FOR ROOKIES',3, 100, 'CIS');
7>     INSERT INTO COURSE1 VALUES
      ('C55','COMPUTER ARCH.','VON NEUMANN'S MACH.',3,100,
'CIS');
```


SET IDLETIMEOUT Command

The SET IDLETIMEOUT command sets the idle timeout value for the current session. The idle timeout value of a session determines when the session expires after a period of inactivity. The default is 30 minutes.

Syntax

```
SET IDLETIMEOUT value
```

value

is an integer representing the idle timeout value in minutes. Zero represents an infinite amount of time, meaning that the session never expires.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If you execute this command in a script file, it affects the session in which the script file runs. You can specify this command in PRUN script files. However, running this command from a PRUN script file does not affect the idle timeout value for the current session.
- To reset the default timeout value, enter this command:

```
SET IDLETIMEOUT 30
```

Examples

- This command sets the idle timeout value to four hours:
- This command sets the idle timeout value to an infinite amount of time so that the session never expires:

```
SQL>set idletimeout 240
```

```
SQL>set idletimeout 0
```

- To reset the idle timeout to the default, enter this command:

```
SQL>set idletimeout 30
```

```
SQL>
```

For more information, see [“Setting and Showing the Idle Timeout Value for the Session”](#) (page 30).

SET LIST_COUNT Command

The `SET LIST_COUNT` command sets the maximum number of rows to be returned by `SELECT` statements that are executed after this command. The default is zero, which means that all rows are returned.

Syntax

```
SET LIST_COUNT num-rows
```

num-rows

is a positive integer that specifies the maximum number of rows of data to be displayed by `SELECT` statements that are executed after this command. Zero means that all rows of data are returned.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- To reset the number of displayed rows, enter this command:

```
SET LIST_COUNT 0
```

Examples

- This command specifies that the number of rows to be displayed by `SELECT` statements is five:

```
SQL>set list_count 5
```

```
SQL>select empnum, first_name, last_name
from persnl.employee
order by empnum;
```

EMPNUM	FIRST_NAME	LAST_NAME
1	ROGER	GREEN
23	JERRY	HOWARD
29	JANE	RAYMOND
32	THOMAS	RUDLOFF
39	KLAUS	SAFFERT

```
--- 5 row(s) selected.  LIST_COUNT was reached.
```

```
SQL>
```

- This command resets the number of displayed rows to all rows:

```
SQL>set list_count 0
```

```
SQL>select empnum, first_name, last_name
+>from persnl.employee
+>order by empnum;
```

EMPNUM	FIRST_NAME	LAST_NAME
1	ROGER	GREEN
23	JERRY	HOWARD
29	JANE	RAYMOND
32	THOMAS	RUDLOFF
39	KLAUS	SAFFERT
43	PAUL	WINTER
65	RACHEL	MCKAY
...		
995	Walt	Farley

```
--- 62 row(s) selected.
```

```
SQL>
```

SET MARKUP Command

The SET MARKUP command sets the markup format and controls how results are displayed by TrafCI.

Syntax

```
SET MARKUP [RAW|HTML|XML|CSV|COLSEP]
```

The supported options enable results to be displayed in XML, HTML, CSV (Comma Separated Values), and COLSEP format. The default format is RAW.

Considerations

- You must enter the command on one line.
- If the MARKUP format is CSV or COLSEP, the column header information and status messages are not displayed.
- For the XML and HTML markup format, the syntax and interface errors have been reformatted so that consistent XML and HTML markup is displayed. In previous releases, the structure of the XML and HTML error depended on whether the error was generated by the application or an SQL error that was received from the database. Also, specific characters are also escaped when the markup is set to XML or HTML. For XML markup, any occurrence of “]]>” that appear in the error message or invalid query are replaced with “]]>”. When error messages are output as HTML markup, both the “>” (greater than) and “<” (less than) symbols are replaced with their escaped versions: “>” and “<”, respectively. An example of the formatted error messages are show below.

Examples

- This command specifies results be displayed in HTML:

```
SQL>set markup html
```

```
SQL>select c.custnum, c.custnum, ordernum, order_date  
+>from customer c, orders o where c.custnum=o.custnum;
```

```
<TABLE>  
<!--select c.custnum, c.custname,ordernum,order_date  
from customer c, orders o where c.custnum=o.custnum;-->  
<tr>  
  <th>CUSTNUM</th>  
  <th>CUSTNAME</th>  
  <th>ORDERNUM</th>  
  <th>ORDER_DATE</th>  
</tr>  
<tr>  
  <td>143</td>  
  <td>STEVENS SUPPLY</td>  
  <td>700510</td>  
  <td>2105-05-01</td>  
</tr>  
<tr>  
  <td>3333</td>  
  <td>NATIONAL UTILITIES</td>  
  <td>600480</td>  
  <td>2105-05-12</td>  
</tr>  
<tr>  
  <td>7777</td>  
  <td>SLEEP WELL HOTELS</td>  
  <td>100250</td>  
  <td>2105-01-23</td>  
</tr>  
<!-- --- 3 row(s) selected.-->
```

```
</TABLE>
```

```
SQL>select c.custnum, c.custname,ordernum,order_date,
+>from customer c, orders o where c.custnum=o.custnum;
```

```
<TABLE>
```

```
<!-- select c.custnum, c.custname,ordernum,order_date,
from customer c, orders o where c.custnum=o.custnum;-->
```

```
<tr>
```

```
  <th>Error Id</th>
```

```
  <th>Error Code</th>
```

```
  <th>Error Message</th>
```

```
<tr>
```

```
  <td>1</td>
```

```
  <td>4082</td>
```

```
  <td>Object TRAFODION.NVS.CUSTOMER does not exist or is inaccessible.</td>
```

```
</tr>
```

```
</TABLE>
```

- To set the application to format output as HTML:

```
SQL>set markup HTML
```

HTML formatted error message example:

```
SQL>set markup <invalid>
```

```
<?xml version="1.0"?>
```

```
<Results>
```

```
  <Query>
```

```
    <![CDATA[set markup <invalid ]]>
```

```
  </Query>
```

```
  <ErrorList>
```

```
    <Error id="1">
```

```
      <ErrorCode>NVCIO001</ErrorCode>
```

```
      <ErrorMsg> <![CDATA[
```

```
ERROR: A syntax error occurred at or before:
```

```
set markup <invalid>
```

```
      ^ ]]></ErrorMsg>
```

```
    </Error>
```

```
  </ErrorList>
```

```
</Results>
```

- This command specifies results be displayed in CSV:

```
SQL>set markup CSV
```

```
SQL>select c.custnum, c.custnum, ordernum, order_date
+>from customer c,orders o where c.custnum=o.custnum;
```

```
143,STEVENS SUPPLY      ,700510,2105-05-01
3333,NATIONAL UTILITIES,600480,2105-05-12
7777,SLEEPWELL HOTELS  ,100250,2105-01-23
324,PREMIER INSURANCE ,500450,2105-04-20
926,METALL-AG.         ,200300,2105-02-06
123,BROWN MEDICAL CO  ,200490,2105-03-19
123,BROWN MEDICAL CO  ,300380,2105-03-19
543,FRESNO STATE BANK ,300350,2105-03-03
5635,ROYAL CHEMICALS   ,101220,2105-05-21
21,CENTRAL UNIVERSITY,200320,2105-02-17
1234,DATASPEED         ,100210,2105-04-10
3210,BESTFOOD MARKETS  ,800660,2105-05-09
```

- This command specifies results be displayed in XML:

```
SQL>set markup xml
```

```
SQL>select * from author
```

```
<?xml version="1.0"?>
```

```
<Results>
```

```

<Query>
  <![CDATA[select * from author;]]>
</Query>
<rowid="1">
  <AUTHORID>91111</AUTHORID>
  <AUTHORNAME>Bjarne Stroustrup</AUTHORNAME>
</row>
<rowid="2">
  <AUTHORID>444444</AUTHORID>
  <AUTHORNAME>John Steinbeck</AUTHORNAME>
</row>
<rowid="3">
  <AUTHORID>2323423</AUTHORID>
  <AUTHORNAME>Irwin Shaw</AUTHORNAME>
</row>
<rowid="4">
  <AUTHORID>93333</AUTHORID>
  <AUTHORNAME>Martin Fowler</AUTHORNAME>
</row>
<rowid="5">
  <AUTHORID>92222</AUTHORID>
  <AUTHORNAME>Grady Booch</AUTHORNAME>
</row>
<rowid="6">
  <AUTHORID>84758345</AUTHORID>
  <AUTHORNAME>Judy Blume</AUTHORNAME>
</row>
<rowid="7">
  <AUTHORID>89832473</AUTHORID>
  <AUTHORNAME>Barbara Kingsolver</AUTHORNAME>
</row>

<Status> <![CDATA[-- 7 row(s) selected .]]></Status>

</Results>

```

- To set the application to format output as XML:

```
SQL>set markup XML
```

XML formatted error message examples:

```

SQL>set markup <]>
<?xml version="1.0"?>
<Results>
  <Query>
    <![CDATA[set markup <]]&#62; ]]]>
  </Query>
  <ErrorList>
    <Error id="1">
      <ErrorCode>UNKNOWN ERROR CODE</ErrorCode>
      <ErrorMessage> <![CDATA[
ERROR: A syntax error occurred at or before:
set markup <]]&#62;
              ^ ]]]></ErrorMsg>
    </Error>
  </ErrorList>
</Results>

```

- This command displays CSV like output using the COLSEP value as a separator.

```
SQL>set colsep |
```

```
SQL>set markup colsep
```

```

SQL>select * from employee;
32|THOMAS          |RUDLOFF          |2000|100|138000.40

```

39		KLAUS			SAFFERT			3200		100		75000.00	
89		PETER			SMITH				3300		300		37000.40
29		JANE			RAYMOND			3000		100		136000.00	
65		RACHEL			MCKAY			4000		100		118000.00	
75		TIM			WALKER			3000		300		320000.00	
11		ROGER			GREEN			9000		100		175500.00	
93		DONALD			TAYLOR			3100		300		33000.00	

SET PARAM Command

The SET PARAM command associates a parameter name with a parameter value in the current session. The parameter name and value are associated with one of these parameter types:

- Named parameter (represented by `?param-name`) in a DML statement or in a prepared SQL statement
- Unnamed parameter (represented by `?`) in a prepared SQL statement only

A prepared statement is one that you SQL compile by using the PREPARE statement. For more information about PREPARE, see the *Trafodion SQL Reference Manual*.

After running SET PARAM commands in the session:

- You can specify named parameters (`?param-name`) in a DML statement.
- You can execute a prepared statement with named parameters by using the EXECUTE statement without a USING clause.
- You can execute a prepared statement with unnamed parameters by using the EXECUTE statement with a USING clause that contains literal values and/or a list of the named parameters set by SET PARAM.

The EXECUTE statement substitutes parameter values for the parameters in the prepared statement. For more information about EXECUTE, see the *Trafodion SQL Reference Manual*.

Syntax

```
SET PARAM param-name [UTF8]param-value
```

param-name

is the name of the parameter for which a value is specified. Parameter names are case-sensitive. For example, the parameter `?pn` is not equivalent to the parameter `?PN`. *param-name* can be preceded by a question mark (`?`), such as `?param-name`.

UTF8

specifies that a character string specified for the parameter value, *param-value*, uses the UTF8 character set. If the character string is in UTF8 format, it must be prefixed by UTF8.

param-value

is a numeric or character literal that specifies the value for the parameter. If you do not specify a value, TrafCI returns an error.

If *param-value* is a character literal and the target column type is a character string, you do not have to enclose the value in single quotation marks. Its data type is determined from the data type of the column to which the literal is assigned. Character strings specified as parameter values are always case-sensitive even if they are not enclosed in quotation marks. If the character string is in UTF8 format, it must be prefixed by UTF8.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- Use separate SET PARAM commands to name and assign values to each unique parameter in a prepared SQL statement before running the EXECUTE statement.
- Parameter names are case-sensitive. If you specify a parameter name in lowercase in the SET PARAM command, you must specify it in lowercase in other statements, such as DML statements or EXECUTE.
- The name of a named parameter (`?param-name`) in a DML statement must be identical to the parameter name (*param-name*) that you specify in a SET PARAM command.

Examples

- This command sets a value for the ?sal (salary) parameter:
SQL>set param ?sal 40000.00
- This command sets a character string value, GREEN, for the ?lastname parameter:
SQL>set param ?lastname GREEN
- These commands set values for named parameters in a subsequent SELECT statement:

```
SQL>set param ?sal 80000.00
```

```
SQL>set param ?job 100
```

```
SQL>select * from persnl.employee  
where salary = ?sal  
and jobcode = ?job;
```

EMPNUM	FIRST_NAME	LAST_NAME	DEPTNUM	JOBCODE	SALARY
72	GLENN	THOMAS	3300	100	80000.00

```
--- 1 row(s) selected.
```

```
SQL>
```

NOTE: The names of the named parameters, ?sal and ?job, in the SELECT statement are identical to the parameter names, sal and job, in the SET PARAM command.

- This command sets a character string value, Peña, which is in UTF8 format, for the ?lastname parameter:
SQL>set param ?lastname utf8'Peña'
- This command sets a character string value, which uses the UTF8 character set and is in hexadecimal notation, for the ?lastname parameter:
SQL>set param ?lastname utf8x'5065266e74696c64653b61'

For more information, see [“Setting Parameters” \(page 35\)](#).

SET PROMPT Command

The SET PROMPT command sets the prompt of the current session to a specified string and/or to the session variables, which start with %. The default prompt is SQL>.

Syntax

```
SET PROMPT [string] [%USER] [%SERVER] [%SCHEMA]
```

string

is a string value to be displayed as the prompt. The string may contain any characters. Spaces are allowed if you enclose the string in double quotes. If you do not enclose the string in double quotes, the prompt is displayed in uppercase.

%USER

displays the session user name as the prompt.

%SERVER

displays the session host name and port number as the prompt.

%SCHEMA

displays the session schema as the prompt.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- To reset the default prompt, enter this command:

```
SET PROMPT
```

Examples

- This SET PROMPT command sets the SQL prompt to ENTER>:

```
SQL>set prompt Enter>
```

```
ENTER>
```
- To reset the SQL prompt to the default, enter this SET PROMPT command:

```
ENTER>set prompt
```

```
SQL>
```
- This command displays the session user name for the prompt:

```
SQL>set prompt %user>
```

```
user1>
```
- This command displays the session host name and port number for the prompt:

```
SQL>set prompt %server>
```

```
sqws135.houston.host.com:22900>
```
- This command displays the session schema for the prompt:

```
SQL>set prompt "Schema %schema:"
```

```
Schema USR:
```
- This command displays multiple session variables:

```
SQL>set prompt %USER@%SCHEMA>
```

```
user1@USR>
```

```
user1@USR> set prompt %SERVER:%USER>
```

```
sqws135.houston.host.com:22900:user1>
```

```
sqws135.houston.host.com:22900:user1>set prompt
```

```
"%schema CI> "
```

```
USR CI>
```

For more information, see [“Customizing the Standard Prompt” \(page 30\)](#).

SET SQLPROMPT Command

The SET SQLPROMPT command sets the SQL prompt of the current session to a specified string. The default is SQL>.

Syntax

```
SET SQLPROMPT [string] [%USER] [%SERVER] [%SCHEMA]
```

string

is a string value to be displayed as the SQL prompt. The string may contain any characters. Spaces are allowed if you enclose the string in double quotes. If you do not enclose the string in double quotes, the prompt is displayed in uppercase.

%USER

displays the session user name as the prompt.

%SERVER

displays the session host name and port number as the prompt.

%SCHEMA

displays the session schema as the prompt.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- To reset the default SQL prompt, enter this command:

```
SET SQLPROMPT
```

Examples

- This command sets the SQL prompt to ENTER>:

```
SQL>set sqlprompt Enter>
```

```
ENTER>
```
- To reset the SQL prompt to the default, enter this command:

```
ENTER>set sqlprompt
```

```
SQL>
```
- This command displays the session user name for the prompt:

```
SQL>set sqlprompt %user>
```

```
user1>
```
- This command displays the session host name and port number for the prompt:

```
SQL>set sqlprompt %server>
```

```
sqws135.houston.host.com:22900>
```
- This command displays the session schema for the prompt:

```
SQL>set sqlprompt "Schema %schema:"
```

```
Schema USR:
```
- This command displays multiple session variables:

```
SQL>set sqlprompt %USER@%SCHEMA>
```

```
user1@USR>
```

```
SQL> set sqlprompt %SERVER:%USER>
```

```
sqws135.houston.host.com:22900:user1>
```

```
sqws135.houston.host.com:22900:user1>set sqlprompt
```

```
"%schema CI> "
```

```
USR CI>
```

For more information, see [“Customizing the Standard Prompt” \(page 30\)](#).

SET SQLTERMINATOR Command

The SET SQLTERMINATOR command sets the SQL statement terminator of the current session. The default is a semicolon (;).

Syntax

```
SET SQLTERMINATOR string
```

string

is a string value for the SQL terminator. The string may contain any characters except spaces. Spaces are disallowed even if you enclose the string in double quotes. Lowercase and uppercase characters are accepted, but the SQL terminator is always shown in uppercase.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- Do not include a reserved word as an SQL terminator.
- If you execute this command in a script file, it affects not only the SQL statements in the script file but all subsequent SQL statements that are run in the current session. If you set the SQL terminator in a script file, reset the default terminator at the end of the script file.
- To reset the default SQL terminator (;), enter this command:

```
SET SQLTERMINATOR ;
```

Examples

- This command sets the SQL terminator to a period (.):

```
SQL>set sqlterminator .
```
- This command sets the SQL terminator to a word, go:

```
SQL>set sqlterminator go
```

This query ends with the new terminator, go:

```
SQL>select * from persnl.employee go
```
- To reset the SQL terminator to the default, enter this command:

```
SQL>set sqlterminator ;
```

For more information, see [“Setting and Showing the SQL Terminator”](#) (page 30).

SET STATISTICS Command

The SET STATISTICS command automatically retrieves the statistics information for a query being executed. The results returned are the same as would have been returned if the GET STATISTICS command was executed. The default is OFF which means the statistics information is not automatically printed for any queries.

Syntax

```
SET STATISTICS {ON | OFF}
```

Considerations

You must enter the command on one line.

Examples

This command shows the default output format as PERTABLE:

```
SQL>set statistics on
```

```
SQL>select * from job;
```

```
JOBCODE JOBDESC
```

```
-----
```

```
100 MANAGER
```

```
1234
```

```
450 PROGRAMMER
```

```
900 SECRETARY
```

```
300 SALESREP
```

```
500 ACCOUNTANT
```

```
400 SYSTEM ANALYST
```

```
250 ASSEMBLER
```

```
420 ENGINEER
```

```
600 ADMINISTRATOR
```

```
200 PRODUCTION SUPV
```

```
--- 11 row(s) selected.
```

```
Start Time          2105/05/18 21:45:34.082329
```

```
End Time            2105/05/18 21:45:34.300265
```

```
Elapsed Time        00:00:00.217936
```

```
Compile Time        00:00:00.002423
```

```
Execution Time      00:00:00.218750
```

Table Name	Records Accessed	Records Used	Disk I/Os	Message Count	Message Bytes	Lock Escl	Lock Wait	Disk Process Busy	Process Time
TRAFODION.TOI.JOB	2	2	0	4	15232	0	0		363

```
SQL>
```

For more information on the STATISTICS command, see the *Trafodion SQL Reference Manual*.

SET TIME Command

The SET TIME command causes the local time of the client workstation to be displayed as part of the interface prompt. By default, the local time is not displayed in the interface prompt.

Syntax

```
SET TIME { ON[12H] | OFF }
```

ON

specifies that the local time be displayed as part of the prompt.

OFF

specifies that the local time not be displayed as part of the prompt. OFF is the default.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- The default is a 24-hour military style display. The additional argument of 12h allows the time to be displayed in a 12-hour AM/PM style.

Examples

- This command causes the local time to be displayed in the SQL prompt:

```
SQL>set time on
```



```
14:17:17 SQL>
```
- This command causes the local time to be displayed in 12-hour AM/PM style in the SQL prompt:

```
SQL>set time on 12h
```



```
2:17:17 PM SQL>
```
- This command turns off the local time in the SQL prompt:

```
2:17:17 PM SQL>set time off
```



```
SQL>
```

For more information, see [“Customizing the Standard Prompt”](#) (page 30).

SET TIMING Command

The SET TIMING command causes the elapsed time to be displayed after each SQL statement executes. This command does not cause the elapsed time of interface commands to be displayed. By default, the elapsed time is off.

Syntax

```
SET TIMING { ON | OFF }
```

ON

specifies the elapsed time be displayed after each SQL statement executes.

OFF

specifies that the elapsed time not be displayed after each SQL statement executes. OFF is the default.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- The elapsed time value includes compile and execution time plus any network I/O time and client-side processing time.

Examples

- This command displays the elapsed time of SQL statements:
`SQL>set timing on`
- This command turns off the elapsed time:
`SQL>set timing off`

For more information, see [“Displaying the Elapsed Time” \(page 31\)](#).

SHOW ACTIVITYCOUNT Command

The SHOW ACTIVITYCOUNT command provides an alias for SHOW RECCOUNT. ACTIVITYCOUNT is an alias for RECCOUNT. For more information, see the [“SHOW RECCOUNT Command” \(page 126\)](#).

Syntax

```
SHOW ACTIVITYCOUNT
```

Examples

This command shows the record count of the previous executed SQL statement:

```
SQL> SHOW ACTIVITYCOUNT
```

```
ACTIVITYCOUNT 0
```

SHOW ALIAS Command

The SHOW ALIAS command displays all or a set of aliases available in the current TrafCI session. If a pattern is specified, all aliases matching the pattern are displayed. By default, all aliases in the current session are displayed.

Syntax

```
SHOW ALIAS [alias-name | wild-card-pattern]
```

alias-name

is any alias name that is used with the ALIAS command. See [“ALIAS Command” \(page 53\)](#).

wild-card-pattern

is a character string used to search for and display aliases with names that match the character string. *wild-card-pattern* matches an uppercase string unless you enclose it within double quotes. To look for similar values, specify only part of the characters of *wild-card-pattern* combined with these wild-card characters.

%	Use a percent sign (%) to indicate zero or more characters of any type. For example, %art% matches SMART, ARTIFICIAL, and PARTICULAR but not smart or Hearts. "%art%" matches smart and Hearts but not SMART, ARTIFICIAL, or PARTICULAR.
*	Use an asterisk (*) to indicate zero or more characters of any type. For example, *art* matches SMART, ARTIFICIAL, and PARTICULAR but not smart or Hearts. "*art*" matches smart and Hearts but not SMART, ARTIFICIAL, or PARTICULAR.
_	Use an underscore (_) to indicate any single character. For example, boo_ matches BOOK and BOOT but not BOO or BOOTS. "boo_" matches book and boot but not boo or boots.
?	Use a question mark (?) to indicate any single character. For example, boo? matches BOOK and BOOT but not BOO or BOOTS. "boo?" matches book and boot but not boo or boots.

Considerations

You must enter the command on one line. The command does not require an SQL terminator.

Examples

This command displays a list of the available aliases:

```
SQL> SHOW ALIAS

.OS AS LH
.GOTO AS GOTO
USE AS SET SCHEMA
```

This command displays the .GOTO alias:

```
SQL> SHOW ALIAS .GOTO

.GOTO AS GOTO
```

This command displays the .FOO alias:

```
SQL> SHOW ALIAS .FOO
```

No aliases found.

This command displays all aliases beginning with the letter “S”:

```
SQL> SHOW ALIAS S*
SEL AS SELECT
SHOWTIME AS SHOW TIME
ST AS SHOW TABLES
```

SHOW ALIASES Command

The SHOW ALIASES command displays all the aliases available in the current TrafCI session.

Syntax

```
SHOW ALIASES
```

Considerations

You must enter the command on one line. The command does not require an SQL terminator.

Examples

This command displays all the aliases in the current TrafCI session:

```
SQL> SHOW ALIASES
```

```
.OS AS LH
```

```
.GOTO AS GOTO
```

```
USE AS SET SCHEMA
```

SHOW CATALOG Command

The SHOW CATALOG command displays the current catalog of the TrafCI session.

Syntax

```
SHOW CATALOG
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Example

This command shows that the current catalog of the session is TRAFODION:

```
SQL>show catalog  
CATALOG TRAFODION
```

SHOW COLSEP Command

The SHOW COLSEP command displays the value of the column separator for the current TrafCI session.

Syntax

```
SHOW
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Examples

- This command displays the column separator.

```
SQL> show colsep  
COLSEP " "
```

```
SQL> set colsep *
```

```
SQL> show colsep  
COLSEP "***"
```

- This command displays the column separator.

```
SQL> show colsep  
COLSEP " "
```

```
SQL> set colsep *
```

```
SQL> show colsep  
COLSEP "***"
```

SHOW ERRORCODE Command

The SHOW ERRORCODE command is an alias for the SHOW LASTERROR command. ERRORCODE is an alias for LASTERROR. For more information, see [“SHOW LASTERROR Command” \(page 121\)](#).

Syntax

```
SHOW ERRORCODE
```

Examples

This command displays the error of the last SQL statement that was executed:

```
SQL> SHOW ERRORCODE
```

```
ERRORCODE 29481
```

SHOW FETCHSIZE Command

The SHOW FETCHSIZE command displays the fetch size value for the current TrafCI session.

Syntax

```
SHOW FETCHSIZE
```

Considerations

You must enter the command on one line.

Examples

These commands display the fetch size in the current TrafCI session, set the fetch size to a new value, and then redisplay the fetch size:

```
SQL>show fetchsize
```

```
FETCHSIZE 0 [Default]
```

```
SQL>SET fetchsize 1
```

```
SQL>SHOW fetchsize
```

```
FETCHSIZE 1
```


SHOW HISTOPT Command

The SHOW HISTOPT command displays the value that has been set for the history option.

Syntax

```
SHOW HISTOPT
```

Considerations

- You must enter the command on one line.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Examples

This command displays the value set for the history option:

```
SQL>show histopt  
HISTOPT DEFAULT [No expansion of script files]
```

```
SQL>set histopt all
```

```
SQL>show histopt  
HISTOPT ALL
```

SHOW IDLETIMEOUT Command

The SHOW IDLETIMEOUT command displays the idle timeout value of the current TrafCI session. The idle timeout value of a session determines when the session expires after a period of inactivity. The default is 30 minutes.

Syntax

```
SHOW IDLETIMEOUT
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Examples

- This command shows that the idle timeout value of the session is 30 minutes, which is the default:

```
SQL>show idletimeout
IDLETIMEOUT 30 min(s)

Elapsed time:00:00:00:078
```
- This command shows that the idle timeout value of the session is four hours:

```
SQL>show idletimeout
IDLETIMEOUT 240 min(s)
```
- This command shows that the idle timeout value is an infinite amount of time, meaning that the session never expires:

```
SQL>show idletimeout
IDLETIMEOUT 0 min(s) [Never Expires]
```
- This command displays the elapsed time information because SET TIMING command is enabled:

```
SQL>set timing on

SQL>show idletimeout
IDLETIMEOUT 0 min(s) [Never Expires]

Elapsed time:00:00:00:078
```

For more information, see [“Setting and Showing the Idle Timeout Value for the Session” \(page 30\)](#).

SHOW LASTERROR Command

The SHOW LASTERROR command displays the error of the last SQL statement that was executed. If the query was successful, 0 is returned; otherwise an SQL error code is returned.

Syntax

```
SHOW LASTERROR
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Examples

This command shows the last error in the current session:

```
SQL>select * from emp;  
*** ERROR[4082]Object TRAFODION.SCH.EMP does not exist or is inaccessible.
```

```
SQL>show lasterror  
LASTERROR 4082
```

SHOW LIST_COUNT Command

The SHOW LIST_COUNT command displays the maximum number of rows to be returned by SELECT statements in the current TrafCI session. The default is zero, which means that all rows are returned.

Syntax

```
SHOW LIST_COUNT
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Examples

- This command shows that SELECT statements return all rows in the current session:

```
SQL>show list_count  
LISTCOUNT 0 [All Rows]  
  
Elapsed time:00:00:00:078
```
- This command shows that the maximum number of rows to be displayed by SELECT statements in the session is five:

```
SQL>set list_count 5  
  
SQL>show list_count  
LISTCOUNT 5  
  
Elapsed time:00:00:00:078
```

SHOW MARKUP Command

The SHOW MARKUP command displays the value set for the markup option.

Syntax

```
SHOW
```

Considerations

- You must enter the command on one line.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Examples

This command displays the value set for the markup option:

```
SQL>show markup  
MARKUP RAW
```

```
Elapsed time:00:00:00:078
```

SHOW PARAM Command

The SHOW PARAM command displays the parameters that are set in the current TrafCI session.

Syntax

```
SHOW
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Example

- This command shows that parameters that are set for the current session:

```
SQL>show param
lastname GREEN
dn 1500
sal 40000.00
```
- This command shows that when no parameters exist, the SHOW PARAM command displays an error message:

```
SQL>show param
No parameters found.
```

For more information, see [“Displaying the Parameters of the Session” \(page 36\)](#).

SHOW PREPARED Command

The SHOW PREPARED command displays the prepared statements in the current TrafCI session. If a pattern is specified, all prepared statements matching the prepared statement name pattern are displayed. By default, all prepared statements in the current session are displayed.

Syntax

```
SHOW PREPARED [stmtNamePattern]
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Examples

This command shows all the prepared statements, by default:

```
SQL>show prepared
S1
    select * from t1

S2
    select * from student

T1
    select * from test123
```

```
SQL> show prepared s%

S1
    select * from t1

S2
    select * from student
```

```
SQL> show prepared t%

T1
    select * from test123
```

SHOW RECCOUNT Command

The SHOW RECCOUNT command displays the record count of the previously executed SQL statement. If the previously executed command was an interface command, TrafCI returns zero.

Syntax

```
SHOW RECCOUNT
```

Considerations

- You must enter the command on one line. The command does not need an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Examples

This command displays the record count of the SQL statement that was executed last:

```
SQL> select * from employee;
```

```
SQL>show reccount  
RECCOUNT 62
```


SHOW REMOTEPROCESS Command

The SHOW REMOTEPROCESS command displays the process name of the DCS server that is handling the current connection.

Syntax

```
SHOW REMOTEPROCESS
```

Considerations

- You must enter the command on one line. The command does not need an SQL terminator.
- The command does not need an SQL terminator.

Example

This command displays the process name, \g4t3028.houston.host.com:0.\$Z0000M2, of the DCS server that is handling the current connection:

```
SQL>show remoteprocess
```

```
REMOTE PROCESS
```

```
\g4t3028.houston.host.com:0.$Z0000M2 SQL>
```

SHOW SCHEMA Command

The SHOW SCHEMA command displays the current schema of the TrafCI session.

Syntax

```
SHOW
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Example

This command shows that the current schema of the session is PERSNL:

```
SQL>show schema  
SCHEMA PERSNL
```

For more information, see [“Setting and Showing the Current Schema”](#) (page 31).

SHOW SESSIONCommand

SHOW SESSION or SESSION displays attributes of the current TrafCI session. You can also use the ENV command to perform the same function.

Syntax

```
[SHOW] SESSION
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.
- SHOW SESSION or SESSION displays these attributes:

COLSEP	Current column separator, which is used to control how query results are displayed. For more information, see “SET COLSEP Command” (page 91) .
HISTOPT	Current history options, which controls how the commands are added to the history buffer. For more information, see “SET HISTOPT Command” (page 93) .
IDLETIMEOUT	Current idle timeout value, which determines when the session expires after a period of inactivity. By default, the idle timeout is 30 minutes. For more information, see “Setting and Showing the Idle Timeout Value for the Session” (page 30) and “SET IDLETIMEOUT Command” (page 95) .
LIST_COUNT	Current list count, which is the maximum number of rows that can be returned by SELECT statements. By default, the list count is all rows. For more information, see “SETLIST_COUNT Command” (page 96) .
LOG FILE	Current log file and the directory containing the log file. By default, logging during a session is turned off. For more information, see “Logging Output” (page 38) and “LOG Command” (page 74) or “SPOOL Command” (page 136) .
LOG OPTIONS	Current logging options. By default, logging during a session is turned off, and this attribute does not appear in the output. For more information, see the “LOG Command” (page 74) or “SPOOL Command” (page 136) .
MARKUP	Current markup option selected for the session. The default option is RAW. For more information, see “SET MARKUP Command” (page 98) .
PROMPT	Current prompt for the session. For example, the default is SQL>. For more information, see “Customizing the Standard Prompt” (page 30) and “SET PROMPT Command” (page 104) .
SCHEMA	Current schema. The default is USR. For more information, see “Setting and Showing the Current Schema” (page 31) .
SERVER	Host name and port number that you entered when logging in to the database platform. For more information, see “Logging In to the Database Platform” (page 21) .
SQLTERMINATOR	Current SQL statement terminator. The default is a semicolon (;). For more information, see “Setting and Showing the SQL Terminator” (page 30) and “SHOW SQLTERMINATOR Command” (page 132) .
STATISTICS	Current setting (on or off) of statistics. For more information, see the “SET STATISTICS Command” (page 109) .
TIME	Current setting (on or off) of the local time as part of the prompt. When this command is set to on, military time is displayed. By default, the local time is off. For more information, see “Customizing the Standard Prompt” (page 30) and “SET TIME Command” (page 110) .
TIMING	Current setting (on or off) of the elapsed time. By default, the elapsed time is off. For more information, see “Displaying the Elapsed Time” (page 31) and “SET TIMING Command” (page 111) .
USER	User name that you entered when logging in to the database platform. For more information, see “Logging In to the Database Platform” (page 21) .

Examples

- This **SHOW SESSION** command displays the attributes of the current session:

```
SQL>show session
```

```
COLSEP           " "
HISTOPT          DEFAULT [No expansion of script files]
IDLETIMEOUT      0 min(s) [Never Expires]
LIST_COUNT       0 [All Rows]
LOG_FILE         c:\session.txt
LOG_OPTIONS      APPEND,CMDTEXT ON
MARKUP           RAW
PROMPT           SQL>
SCHEMA           SEABASE
SERVER           sqws135.houston.host.com:378
00 SQLTERMINATOR ;
STATISTICS       OFF
TIME             OFF
TIMING           OFF
USER             user1
```

- This **SESSION** command shows the effect of setting various session attributes:

```
SQL>session
```

```
COLSEP           " "
HISTOPT          DEFAULT [No expansion of script files]
IDLETIMEOUT      30 min(s)
LIST_COUNT       0 [All Rows]
LOG              OFF
MARKUP           RAW
PROMPT           SQL>
SCHEMA           SEABASE
SERVER           sqws135.houston.host.com:378
00 SQLTERMINATOR ;
STATISTICS       OFF
TIME             OFF
TIMING           OFF
USER             user1

SQL>
```

SHOW SQLPROMPT Command

The SHOW SQLPROMPT command displays the value of the SQL prompt for the current TrafCI session.

Syntax

```
SHOW SQLPROMPT
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Example

This command shows that the SQL prompt for the current session is `SQL>`:

```
SQL>show sqlprompt  
SQLPROMPT SQL>
```

SHOW SQLTERMINATOR Command

The SHOW SQLTERMINATOR command displays the SQL statement terminator of the current TrafCI session.

Syntax

```
SHOW SQLTERMINATOR
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Example

This command shows that the SQL terminator for the current session is a period (.):

```
SQL>show sqlterminator  
SQLTERMINATOR .
```

For more information, see [“Setting and Showing the SQL Terminator” \(page 30\)](#).

SHOW STATISTICS Command

The SHOW STATISTICS command displays if statistics has been enabled or disabled for the current session.

Syntax

```
SHOW STATISTICS
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Example

This command shows SHOW STATISTICS disabled and then enabled:

```
SQL>show statistics  
STATISTICS OFF
```

```
SQL>set statistics on
```

```
SQL>show statistics  
STATISTICS ON
```

SHOW TIME Command

The SHOW TIME command displays whether the setting for the local time in the interface prompt is ON or OFF.

Syntax

```
SHOW
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Example

This command shows that the setting for the local time in the SQL prompt is OFF:

```
SQL>show time  
TIME OFF
```


SHOW TIMING Command

The SHOW TIMING command displays whether the setting for the elapsed time is ON or OFF.

Syntax

```
SHOW
```

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- If the SET TIMING command is set to ON, the elapsed time information is displayed.

Example

- This command displays the elapsed time information because the SET TIMING command is enabled:

```
SQL>set timing on
```

```
SQL>show time  
TIME OFF
```

```
Elapsed :00:00:00.000
```

SPOOL Command

The SPOOL command logs the entered commands and their output from TrafCI to a log file.

Syntax

```
SPOOL { ON [CLEAR, QUIET, CMDTEXT {ON | OFF}]  
      | log-file [CLEAR, QUIET, CMDTEXT {ON | OFF}]  
      | OFF }
```

ON

starts the logging process and records information in the `sqlspool.lst` file in the `bin` directory.

ON CLEAR

instructs TrafCI to clear the contents of the `sqlspool.lst` file before logging new information to the file.

QUIET

specifies that the command text is displayed on the screen, but the results of the command are written only to the log file and not to the screen.

CMDTEXT ON

specifies that the command text and the log header are displayed in the log file.

CMDTEXT OFF

specifies that the command text and the log header are not displayed in the log file.

log-file

is the name of a log file into which TrafCI records the entered commands and their output. If you want the log file to exist outside the local directory where you launch TrafCI (by default, the `bin` directory), specify the full directory path of the log file. The log file does not need to exist, but the specified directory must exist before you execute the SPOOL command.

log-file CLEAR

instructs TrafCI to clear the contents of the specified *log-file* before logging new information to the file.

OFF

stops the logging process.

Considerations

- You must enter the command on one line. The command does not require an SQL terminator.
- Use a unique name for each log file to avoid writing information from different TrafCI sessions into the same log file.

Examples

- This command starts the logging process and records information to the `sqlspool.lst` file in the `bin` directory:
`SQL>spool on`
- This command starts the logging process and appends new information to an existing log file, `persnl_updates.log`, in the local directory (the same directory where you are running TrafCI):
`SQL>spool persnl_updates.log`
- This command starts the logging process and appends new information to a log file, `sales_updates.log`, in the specified directory on a Windows workstation:
`SQL>spool c:\log_files\sales_updates.log`

- This command starts the logging process and appends new information to a log file, `sales_updates.log`, in the specified directory on a Linux or UNIX workstation:
SQL>spool ./log_files/sales_updates.log
- This command starts the logging process and clears existing information from the log file before logging new information to the file:
SQL>spool persnl_ddl.log clear
- This command starts the logging process and records information to the `sqlspool.lst` file in the `bin` directory:
SQL>log on
- This command starts the logging process and appends new information to an existing log file, `persnl_updates.log`, in the local directory (the same directory where you are running TrafCI):
SQL>log persnl_updates.log
- This command starts the logging process and appends new information to a log file, `sales_updates.log`, in the specified directory on a Windows workstation:
SQL>log c:\log_files\sales_updates.log
- This command starts the logging process and appends new information to a log file, `sales_updates.log`, in the specified directory on a Linux or UNIX workstation:
SQL>log ./log_files/sales_updates.log
- This command starts the logging process and clears existing information from the log file before logging new information to the file:
SQL>log persnl_ddl.log clear
- This command start the logging process, clears existing information from the log file, and specifies that the command text and log header is not displayed in the log file:
SQL>log c:\temp\a.txt clear, cmdtext off

```
SQL>select * from trafodion.toi.job
+>;
```

```
JOBCODE  JOBDESC
-----  -
      100  MANAGER
      450  PROGRAMMER
      900  SECRETARY
     300  SALESREP
     500  ACCOUNTANT
     400  SYSTEM ANALYST
     250  ASSEMBLER
     420  ENGINEER
     600  ADMINISTRATOR
     200  PRODUCTION SUPV
```

```
--- 10 row(s) selected.
```

```
SQL> log off
```

```
Output of c:\temp\a.txt
```

```
=====
JOBCODE  JOBDESC
-----  -
      100  MANAGER
      450  PROGRAMMER
      900  SECRETARY
     300  SALESREP
     500  ACCOUNTANT
```

```
400 SYSTEM ANALYST
250 ASSEMBLER
420 ENGINEER
600 ADMINISTRATOR
200 PRODUCTION SUPV
```

```
--- 10 row(s) selected
```

- This command start the logging process, clears existing information from the log file, and specifies that no output appears on the console window:

```
SQL>log c:\temp\b.txt clear, cmdtext off, quiet
```

```
SQL>select *
+>from trafodion.toi.job;
```

```
SQL> log off
```

```
Output of c:\temp\b.txt
```

```
=====
```

```
JOBCODE JOBDESC
```

```
-----
```

```
100 MANAGER
450 PROGRAMMER
900 SECRETARY
300 SALESREP
500 ACCOUNTANT
400 SYSTEM ANALYST
250 ASSEMBLER
420 ENGINEER
600 ADMINISTRATOR
200 PRODUCTION SUPV
```

```
--- 10 row(s) selected
```

This command stops the logging process:

```
SQL>log off
```

For more information, see [“Logging Output” \(page 38\)](#).

VERSION Command

The VERSION command displays the build versions of the Trafodion database, Trafodion Connectivity Service, Trafodion JDBC Type 4 Driver, and TrafCI.

Syntax

```
VERSION
```

Considerations

You must enter the command on one line. The command does not require an SQL terminator.

Example

- This command shows versions of the Trafodion database, Trafodion Connectivity Service, Trafodion JDBC Type 4 Driver, and TrafCI:

```
SQL>version
```

```
Trafodion Platform           : Release 0.8.0
Trafodion Connectivity Services : Version 1.0.0 Release 0.8.0
Trafodion JDBC Type 4 Driver  : Traf_JDBC_Type4_Build_40646)
Trafodion Command Interface   : TrafCI_Build_40646
```

```
SQL>
```

- If TrafCI is started with the `-noconnect` parameter, the VERSION command displays only TrafCI and the Trafodion JDBC Type 4 Driver versions.

```
C:\Program Files (x86)\Apache Software Foundation\Trafodion Command Interface\bin>trafci
-noconnect Welcome to Trafodion Command Interface
Copyright(C) 2013-2105 Apache Software Foundation
```

```
SQL>version
```

```
Trafodion Platform           : Information not available.
Trafodion Connectivity Services : Information not available.
Trafodion JDBC Type 4 Driver  : Traf_JDBC_Type4_Build_40646
Trafodion Command Interface   : TrafCI_Build_40646
```

Index

Symbols

- h parameter, [24](#)
- help
 - description of, [24](#)
 - examples of, [27](#)
- host parameter, [24](#)
- noconnect
 - description of, [24](#)
 - examples of, [26](#)
- p parameter, [24](#)
- password parameter, [24](#)
- q parameter
 - description of, [24](#)
 - examples of, [24](#), [25](#)
- r parameter, [24](#)
- role parameter, [24](#)
- s parameter
 - description of, [24](#)
 - examples of, [26](#)
- script parameter, [24](#)
- sql parameter, [24](#)
- u parameter, [24](#)
- user parameter, [24](#)
- version
 - description of, [24](#)
 - examples of, [27](#)
- / command
 - example of, [34](#)
 - syntax of, [52](#)
- @ command
 - example of, [41](#)
 - syntax of, [51](#)

A

ALIAS command, [53](#)

C

Case sensitivity, [29](#)
CLEAR command, [54](#)
CLEAR option, [38](#)
Command line, breaking, [28](#)
Comments, [40](#)
Conditional exit, [61](#), [83](#)
CONNECT command, [55](#)
CREATE SCHEMA statement, example of, [24](#), [41](#)
CREATE TABLE statement, example of, [41](#)

D

Data definition language (DDL) statements, running simultaneously, [42](#)
Database, creating, [42](#)
DCS server, showing the process name, [127](#)
Default schema, [31](#)
DELAY command, [57](#)
DISCONNECT command, [58](#)

E

Editing commands, FC, [62](#)
Elapsed time, displaying, [31](#)
ENV command, [59](#)
Environment variables

- PATH, [20](#)
- TRAFCL_PERL_JSERVER, [43](#)
- TRAFCL_PERL_JSERVER_PORT, [43](#)
- TRAFCL_PYTHON_JSERVER, [43](#)

Error messages, [34](#)
EXECUTE statement, examples of, [36](#)
EXIT command, [61](#)

F

FC command, [62](#)

G

GET STATISTICS command, [65](#)
GOTO command, [67](#)

H

HELP command, [68](#)
HISTORY command, [69](#)
Host name, [21](#)

I

Idle timeout value, [30](#)
IF...THEN command, [70](#)
INSERT statement, example of, [30](#)
Installation procedures

- testing the launch of TrafCl, [16](#)

Interface commands

- descriptions of, [48](#)
- using in a script file, [40](#)

IP address, [21](#)

L

LABEL command, [72](#)
Launch parameters

- descriptions of, [23](#)
- presetting on Linux or UNIX, [21](#)
- presetting on Windows, [19](#)

LOCALHOST command, [73](#)
LOG command, [74](#)
Log files

- PRUN operation, [80](#)
- TrafCl session, [39](#)

Logging in

- default method, [21](#)
- using login parameters, [22](#)

Logging output

- concurrent sessions, [39](#)
- script file execution, [42](#)
- starting, [38](#)
- stopping, [39](#)

- viewing a log file, [39](#)
- Login environment variables
 - description of, [43](#)
 - setting in the user profile, [46](#)
 - setting in Windows System Properties, [44](#)
 - setting on a Linux or UNIX command line, [46](#)
 - setting on a Windows command line, [43](#)
- Login parameters
 - presetting on Linux or UNIX, [21](#)
 - presetting on Windows, [19](#)
 - specifying on the command line, [22](#)

M

MXOSRVR, showing the process name, [127](#)

O

OBEY command, [41](#), [77](#)

P

Parameters, SQL

- displaying, [36](#)
- resetting, [36](#)
- setting, [35](#)

PATH

- Trafodion Command Interface (TrafCI) setting
 - Linux, [15](#)
 - Windows, [14](#)

PATH environment variable, [20](#)

Perl command line, invoking TrafCI, [46](#)

Perl program, running, [46](#)

Perl wrapper script, description of, [46](#)

Port number, default, [21](#)

PREPARE statement, examples of, [34](#)

Product banner, [28](#)

Prompt, [28](#)

PRUN command, [80](#)

Python command line, invoking TrafCI, [46](#)

Python program, running, [46](#)

Python wrapper script, description of, [46](#)

Q

QUIT command, [83](#)

R

RECONNECT command, [84](#)

REPEAT command, [85](#)

RESET LASTERROR command, [87](#)

RESET PARAM command

- examples of, [36](#)
- syntax of, [88](#)

RUN command, [89](#)

S

SAVEHIST command, [90](#)

Schema

- setting the current schema, [31](#)
- showing the current schema, [32](#)

Script file

- comments, [40](#)

creating, [40](#)

example of, [41](#)

running multiple files in parallel, [42](#)

running one file at a time, [41](#)

running when launching TrafCI, [25](#)

SELECT statement, example of, [33](#)

Session, [30](#)

SESSION command, [129](#)

SET COLSEP command, [91](#)

SET commands, in a script file, [25](#)

SET FETCHSIZE command, [92](#)

SET HISTOPT command, [93](#)

SET IDLETIMEOUT command

- example of, [30](#)
- syntax of, [95](#)

SET LIST_COUNT command, [96](#)

SET MARKUP command, [98](#)

SET PARAM command

- examples of, [35](#), [36](#)
- syntax of, [102](#)

SET PROMPT command

- example of, [30](#)
- syntax of, [104](#)

SET SCHEMA statement, example of, [31](#), [41](#)

SET SQLPROMPT command, [106](#)

SET SQLTERMINATOR command

- example of, [30](#)
- syntax of, [108](#)

SET STATISTICS command, [109](#)

SET TIME command

- examples of, [30](#)
- syntax of, [110](#)

SET TIMING command

- examples of, [31](#)
- syntax of, [111](#)

SHOW ACTIVITYCOUNT command, [112](#)

SHOW ALIAS command, [113](#)

SHOW ALIASES command, [114](#)

SHOW CATALOG command

- syntax of, [115](#)

SHOW COLSEP command, [116](#)

SHOW ERRORCODE command, [117](#)

SHOW FETCHSIZE command, [118](#)

SHOW HISTOPT command, [119](#)

SHOW IDLETIMEOUT command

- example of, [30](#)
- syntax of, [120](#)

SHOW LASTERROR command, [121](#)

SHOW LIST_COUNT command, [122](#)

SHOW MARKUP command, [123](#)

SHOW PARAM command

- example of, [36](#)
- syntax of, [124](#)

SHOW PREPARED command, [125](#)

SHOW RECOUNT command, [126](#)

SHOW REMOTEPROCESS command, [127](#)

SHOW SCHEMA command

- example of, [32](#)
- syntax of, [128](#)

- SHOW SESSION command, [129](#)
- SHOW SQLPROMPT command, [131](#)
- SHOW SQLTERMINATOR command, [132](#)
- SHOW STATISTICS command, [133](#)
- SHOW TIME command, [134](#)
- SHOW TIMING command, [135](#)
- SPOOL command
 - examples of, [38](#)
 - syntax of, [136](#)
- Spooling see Logging output
- SQL parameters
 - displaying, [36](#)
 - resetting, [36](#)
 - setting, [35](#)
- SQL prompt
 - description of, [28](#)
- SQL statement
 - breaking across lines, [28](#)
 - displaying the elapsed time, [31](#)
 - editing, [62](#), [65](#)
 - preparing and executing, [34](#)
 - repeating, [34](#), [85](#)
 - running in TrafCI, [33](#)
 - running when launching TrafCI, [24](#)
 - terminating, [28](#)
- SQL terminator
 - setting, [30](#)
 - showing, [31](#)
- Standard prompt
 - description of, [28](#)
- Standard prompts
 - customizing, [30](#)
 - displaying the current time, [30](#)

T

- Timeout, idle session, [30](#)
- trafci.cmd, creating a shortcut to, [17](#)
- trafci.pl, description of, [46](#)
- trafci.py, description of, [46](#)
- trafci.sh, setting the path of, [20](#)
- TRAFCI_PERL_JSERVER environment variable, [43](#)
- TRAFCI_PERL_JSERVER_PORT environment variable, [43](#)
- TRAFCI_PYTHON_JSERVER environment variable, [43](#)
- Trafodion Command Interface (TrafCI), [26](#), [27](#)
 - breaking across lines, [28](#)
 - case sensitivity, [29](#)
 - description of, [13](#), [28](#)
 - exiting, quitting, or disconnecting, [27](#)
 - launching and running a command, [24](#)
 - launching and running a script file, [25](#)
 - launching from a Perl or Python command line, [46](#)
 - launching on a Linux client workstation, [20](#)
 - launching on a Windows client workstation, [17](#)
 - logging output, [38](#)
 - product banner, [28](#)
 - prompt, [28](#)
 - testing the launch of, [16](#)
- Transaction, example of, [39](#)

U

- User profile
 - setting login environment variables, [46](#)
 - setting the PATH, [20](#)

V

- VERSION command, [139](#)

W

- Wild-card characters, [51](#), [77](#)