Oracle® SQL Developer

Command-Line Quick Reference

Release 18.4

F12904-01

January 2019

About This Quick Reference

This Quick Reference discusses the following features of SQL Developer Command-Line (SQLcl):

Alphabetic List of SQLcl Commands

@{url | file_name[.ext]} [arg ...]

@@ { url | file_name[.ext] } [arg ...]

/ (slash)

ACC[EPT] *variable* [NUM[BER] | CHAR | DATE | BINARY_FLOAT | BINARY_DOUBLE] [FOR[MAT] *format*] [DEF[AULT] *default*] [PROMPT *text* | NOPR[OMPT]] [HIDE]

ALIAS [*<name>=<SQL statement>*;| LOAD [*<filename>*]|SAVE [*<filename>*] | LIST [*<NAME>*] | DROP *<name>* | DESC *<name> <Description String>*]

APEX [export <application_id>]

A[PPEND] *text*

ARCHIVE LOG LIST

BRE[AK] [ON *report_element* [*action* [*action*]]] ...

BRIDGE

BTI[TLE] [*printspec* [*text* | *variable*] ...] | [ON | OFF]

CD [<directory>]

C[HANGE] *sepchar old* [*sepchar* [*new* [*sepchar*]]]

CL[EAR] *option* ...

COL[UMN] [{*column* | *expr*} [*option* ...]]

COMP[UTE] [*function* [LAB[EL] *text*] ... OF {*expr* | *column* | *alias*} ...ON {*expr* | *column* | *alias* | REPORT | ROW} ...]

CONN[ECT] [{*<logon>*| / |*proxy*} [AS {SYSOPER | SYSDBA | SYSASM}] [edition=*value*]]

CTAS *table new_table*

COPY {FROM *database* | TO *database* | FROM *database* TO *database*} {APPEND | CREATE | INSERT | REPLACE | APPEND_BYTE | CREATE_BYTE | REPLACE_BYTE} *destination_table*[(*column*, *column*, *column*, ...)] USING *query*

DDL [*object_name* [*type*] [SAVE *filename*]]

DEF[INE] [*variable*] | [*variable = text*]

DEL [*n* | *n m* | *n* * | *n* LAST | * | * *n* | * LAST | LAST]

DESC[RIBE] {[*schema.*]*object*[@*connect_identifier*]}

DISC[ONNECT]

ED[IT] [*file_name*[.*ext*]]

EXEC[UTE] *statement*

{EXIT | QUIT} [SUCCESS | FAILURE | WARNING | *n* | *variable* | *:BindVariable*] [COMMIT | ROLLBACK]

FIND [<filename>]

FORMAT [BUFFER | RULES <filename> | FILE <input_file> <output_file>]

GET [FILE] *file_name*[.*ext*] [LIST | NOLIST]

HELP | ? [*topic*]

HISTORY [index | FULL | USAGE | SCRIPT | TIME | CLEAR (SESSION)?]

HO[ST] [*command*]

INFO[RMATION] {[*schema.*]object[@*connect_identifier*]}

I[NPUT] [*text*]

L[IST] [*n* | *n m* | *n* * | *n* LAST | * | * *n* | * LAST | LAST]

LOAD [*schema.*]*table_name*[@*db_link*] *file_name*

OERR *<facility> <error>*

PASSW[ORD] [*username*]

PAU[SE] [*text*]

PRINT [*variable* ...]

PRO[MPT] [*text*]

{QUIT | EXIT} [SUCCESS | FAILURE | WARNING | *n* | *variable* | *:BindVariable*]
[COMMIT | ROLLBACK]

REM[ARK]

REPEAT *<iterations> <sleep>*

REST [*export* [<module_name> | <module_prefix>] | *modules* | *privileges* | *schemas*]

R[UN]

SAV[E] *[FILE] file_name*[*.ext*] [CRE[ATE] | REP[LACE] | APP[END]]

SCRIPT *<script file>*

SET *system_variable value*

SHO[W] [*option*]

SHUTDOWN [ABORT | IMMEDIATE | NORMAL | TRANSACTIONAL [LOCAL]]

SODA

SPO[OL] [*filename*[*.ext*] [CRE[ATE] | REP[LACE] | APP[END]] | OFF | OUT]

SSHTUNNEL *<username>@<hostname>* -i *<identity_file>* [-L
localPort:Remotehost:RemotePort]

STA[RT] { *url* | *file_name*[*.ext*] } [*arg* ...]

STARTUP *db_options* | *cdb_options* | *upgrade_options*

STORE {SET} *file_name*[*.ext*] [CRE[ATE] | REP[LACE] | APP[END]]

TNSPING *<address>*

TTI[TLE] [*printspec* [*text* | *variable*] ...] | [ON | OFF]

UNDEF[INE] *variable* ...

WHENEVER OSERROR {EXIT [SUCCESS | FAILURE | *n* | *variable* | *:BindVariable*]
[COMMIT | ROLLBACK] | CONTINUE[COMMIT | ROLLBACK | NONE]}

WHENEVER SQLERROR {EXIT [SUCCESS | FAILURE | WARNING | *n* | *variable*
| *:BindVariable*] [COMMIT | ROLLBACK] | CONTINUE [COMMIT | ROLLBACK |
NONE]}

WHICH *<filename>*

XQUERY *xquery_statement*

> **Note:**
>
> You can use the up and down arrow keys to cycle through the previous 100 statements or scripts.

List of Unsupported Commands and Features in SQL*Plus Commands

- REPHEADER
- REPFOOTER
- TIMING

The TIMING command is replaced by the SET TIMING command.

System Variables and Environment Settings through the SET Command

- autoprint
- autorecovery
- cmdsep
- copytypecheck
- describe
- eschar
- flagger
- flush
- fullcolname
- logsource
- loboffset
- markup
- recsep
- recsepchar
- shiftinout
- sqlcase
- sqlprefix
- sqlterminator
- tab

- underline
- xmloptimizationcheck

Starting and Leaving SQLcl

Use the following commands to log in to and out of SQLcl.

SQLCL [[*option*] [*logon* | / NOLOG] [*start*]]

where *option* has the following syntax:

-H[ELP] | -V[ERSION] | [[-C[OMPATIBILITY] *x.y*[.*z*]]] [-L[OGON]] [-NOLOGINTIME] [-R[ESTRICT] {1 | 2 | 3}] [-S[ILENT]] [-AC]]

where *logon* has the following syntax:

{*username*[/*password*] [*@connect_identifier*] | /} [AS {SYSASM |SYSBACKUP | SYSDBA |SYSDG |SYSOPER |SYSRAC |SYSKM}] [edition=*value*]

and where *start* has the following syntax:

@{*url* | *file_name*[.*ext*]} [*arg* ...]

{EXIT | QUIT} [SUCCESS | FAILURE | WARNING | *n* | *variable* | *:BindVariable*] [COMMIT | ROLLBACK]

Commits or rolls back all pending changes, logs out of Oracle, terminates SQLcl and returns control to the operating system.

{QUIT | EXIT} [SUCCESS | FAILURE | WARNING | *n* | *variable* | *:BindVariable*] [COMMIT | ROLLBACK]

Commits or rolls back all pending changes, logs out of Oracle, terminates SQLcl and returns control to the operating system.

Starting Up and Shutting Down a Database

Starting up and shutting down a database requires DBA privileges.

STARTUP *db_options* | *cdb_options* | *upgrade_options*

where *db options* has the following syntax:

[FORCE] [RESTRICT] [PFILE=*filename*] [QUIET] [MOUNT [*dbname*] | [OPEN [*open_db_options*] [*dbname*]] | NOMOUNT]

where *open_db_options* has the following syntax:

READ {ONLY | WRITE [RECOVER]} | RECOVER

where *cdb_options* has the following syntax:

root_connection_options | *pdb_connection_options*

where *root_connection_options* has the following syntax:

PLUGGABLE DATABASE *pdbname* [FORCE] | [RESTRICT] [OPEN {*open_pdb_options*}]

where *pdb_connection_options* has the following syntax:

[FORCE] | [RESTRICT] [OPEN {*open_pdb_options*}]

where *open_pdb_options* has the following syntax:

READ WRITE | READ ONLY

and where *upgrade_options* has the following syntax:

[PFILE=*filename*] {UPGRADE | DOWNGRADE} [QUIET]

Starts an Oracle Database instance with several options, including mounting and opening a database.

SHUTDOWN [ABORT | IMMEDIATE | NORMAL | TRANSACTIONAL [LOCAL]]

Shuts down a currently running Oracle instance, optionally closing and dismounting a database.

Entering and Executing Commands

Use the following commands to execute and collect timing statistics on SQL commands and PL/SQL blocks:

/ (slash)

Executes the most recently executed SQL command or PL/SQL block which is stored in the SQL buffer. Does not list the command. Use slash (/) at the command prompt or line number prompt in SQLcl command line.

EXEC[UTE] *statement*

Executes a single PL/SQL statement or runs a stored procedure.

R[UN]

Lists and executes the most recently executed SQLcl command or PL/SQL block which is stored in the SQL buffer. The buffer has no command history list and does not record SQLcl commands.

TIMI[NG]

Timing is only available as a switch.

Use the following command to access the help system:

HELP | ? [*topic*]

Accesses the command-line help system. Enter HELP INDEX or ? INDEX for a list of topics. You can view the Oracle Database Library at http://www.oracle.com/technology/documentation.

Use the following command to execute operating system commands:

HO[ST] [*command*]

Executes an operating system command without leaving SQLcl. Enter HOST without command to display an operating system prompt. You can then enter multiple operating system commands.

With some operating systems, you can use another character instead of HOST such as "!" (UNIX) and "$" (Windows). See the Oracle installation and user's manuals provided for your operating system for details.

Use the following command to recall the history of SQLcl commands:

HISTORY [*index* | FULL | USAGE | SCRIPT | TIME | CLEAR (SESSION)?] | FAILS

- Use the Up and Down arrow keys to navigate through history items at the prompt.

- Use the HISTORY command to print the history contents.

- History is limited to the last 100 statements.

- SET HISTORY LIMIT N allows you to change the default limit, where N is the maximum number.

- History is retained between SQLcl sessions.

- By default, the SHOW, HISTORY, CONNECT, and SET commands are not saved in history.

- SET HISTORY BLACKLIST allows you to set the commands that should not be recorded in history.

Manipulating SQL, SQLcl, and PL/SQL Commands

Use the following commands to edit SQL commands and PL/SQL blocks:

A[PPEND] *text*

Adds specified text to the end of the current line in the SQL buffer. To separate *text* from the preceding characters with a space, enter two spaces. To append *text* that ends with a semicolon, end the command with two semicolons (a single semicolon is interpreted as a command terminator).

C[HANGE] *sepchar old* [*sepchar* [*new* [*sepchar*]]]

Changes first occurrence of *old* on the current line of the SQL buffer. The buffer has no command history list and does not record SQLcl commands. You can use any non-alphanumeric character such as "/" or "!" as a *sepchar*. You can omit the space between CHANGE and the first *sepchar*.

DEL [*n* | *n m* | *n* * | *n* LAST | * | * *n* | * LAST | LAST]

Deletes one or more lines of the SQL buffer ("*" indicates the current line). You can omit the space between DEL and *n* or *, but not between DEL and LAST. Enter DEL with no clauses to delete the current line of the buffer. The buffer has no command history list and does not record SQLcl commands.

I[NPUT] [*text*]

Adds one or more new lines of text after the current line in the SQL buffer. The buffer has no command history list and does not record SQLcl commands.

L[IST] [*n* | *n m* | *n* * | *n* LAST | * | * *n* | * LAST | LAST]

Lists one or more lines of the most recently executed SQL command or PL/SQL block which is stored in the SQL buffer. Asterisk (*) indicates the current line. You can omit the space between LIST and *n* or *, but not between LIST and LAST. Enter LIST with no clauses to list all lines.

In SQLcl, you can also use ";" to list all the lines in the SQL buffer. The buffer has no command history list and does not record SQLcl commands.

Use the following commands to run scripts:

@ { *url* | *file_name*[*.ext*] } [*arg* ...]

Runs the SQLcl statements in the specified script. The script can be called from the local file system or a web server. You can pass values to script variables in the usual way.

@@ { *url* | *file_name*[*.ext*] } [*arg* ...]

Runs the SQLcl statements in the specified script. This command is almost identical to the @ command. It is useful for running nested scripts because it has the additional functionality of looking for the specified script in the same path or *url* as the calling script.

REPEAT *<iterations> <sleep>*

Repeats the current SQL in the buffer at the specified times with sleep intervals. The maximum sleep interval is 120 seconds.

SCRIPT *<script file>*

Runs the SQLcl statements in the specified script.

STA[RT] { *url* | *file_name*[*.ext*] } [*arg* ...]

Runs the SQLcl statements in the specified script. The script can be called from the local file system or a web server. You can pass values to script variables in the usual way.

Use the following commands to create and modify scripts:

ED[IT] [*file_name*[*.ext*]]

Invokes an operating system text editor on the contents of the specified file or on the contents of the SQL buffer. To edit the buffer contents, omit the file name.

The DEFINE variable _EDITOR can be used to set the editor to use. In SQLcl, _EDITOR can be set to any editor that you prefer. *Inline* will set the editor to be the SQLcl editor. This supports the following shortcuts:

- ^R - Run the current buffer
- ^W - Go to top of buffer
- ^S - Go to bottom of buffer
- ^A - Go to start of line
- ^E - Go to end of line

FORMAT

- FORMAT BUFFER - formats the script in the SQLcl Buffer
- FORMAT RULES *<filename>* - Loads SQLDeveloper Formatter rules file to formatter
- FORMAT FILE *<input_file>* *<output_file>*

GET *file_name*[.*ext*] [LIST | NOLIST]

Loads a SQL statement or PL/SQL block from a file into the SQL buffer. The buffer has no command history list and does not record SQLcl commands.

REM[ARK]

Begins a comment in a script. The REMARK command must appear at the beginning of a line, and the comment ends at the end of the line (a line cannot contain both a comment and a command). SQLcl does not interpret the comment as a command.

SAV[E] [FILE] *file_name*[.*ext*] [CRE[ATE] | REP[LACE] | APP[END]]

Saves the contents of the SQL buffer in a script. The buffer has no command history list and does not record SQLcl commands.

STORE {SET} *file_name*[.*ext*] [CRE[ATE] | REP[LACE] | APP[END]]

Saves attributes of the current SQLcl environment in a file.

WHENEVER OSERROR {EXIT [SUCCESS | FAILURE | *n* | *variable* | *:BindVariable*] [COMMIT | ROLLBACK] | CONTINUE [COMMIT | ROLLBACK | NONE]}

Performs the specified action (exits SQLcl by default) if an operating system error occurs (such as a file writing error).

WHENEVER SQLERROR {EXIT [SUCCESS | FAILURE | WARNING | *n* | *variable* | *:BindVariable*] [COMMIT | ROLLBACK] | CONTINUE [COMMIT | ROLLBACK | NONE]}

Performs the specified action (exits SQLcl by default) if a SQL command or PL/SQL block generates an error.

Use the following commands to write interactive commands:

ACC[EPT] *variable* [NUM[BER] | CHAR | DATE | BINARY_FLOAT | BINARY_DOUBLE] [FOR[MAT] *format*] [DEF[AULT] *default*] [PROMPT *text* | NOPR[OMPT]] [HIDE]

Reads a line of input and stores it in a given substitution variable.

DEF[INE] [*variable*] | [*variable* = *text*]

Specifies a substitution variable and assigns a CHAR value to it, or lists the value and variable type of a single variable or all variables.

PAU[SE] [*text*]

Displays the specified text then waits for the user to press RETURN.

PRO[MPT] [*text*]

Sends the specified message or a blank line to the user's screen.

UNDEF[INE] *variable* ...

Deletes one or more substitution variables that you defined either explicitly (with the DEFINE command) or implicitly (with a START command argument).

Use the following commands to create and display bind variables:

PRINT [*variable* ...]

Displays the current values of bind variables, or lists all bind variables.

Use the following symbols to create substitution variables and parameters for use in scripts:

&*n*

Specifies a parameter in a script you run using the START command. START substitutes values you list after the script name as follows: the first for &1, the second for &2, and so on.

&*user_variable*, &&*user_variable*

Indicates a substitution variable in a SQL or SQLcl command. SQLcl substitutes the value of the specified substitution variable for each substitution variable it encounters. If the substitution variable is undefined, SQLcl prompts you for a value *each* time an "&" variable is found, and the *first* time an "&&" variable is found.

. (period)

Terminates a substitution variable followed by a character that would otherwise be part of the variable name.

Formatting Query Results

Use the following commands to format, store and print your query results.

BRE[AK] [ON *report_element* [*action* [*action*]]] ...

Specifies where changes occur in a report and the formatting action to perform, such as:

- suppressing the display of duplicate values for a given column
- skipping a line each time a given column value changes
- printing computed figures each time a given column value changes or at the end of the report

Enter BREAK with no clauses to list the current BREAK definition.

Where *report_element* has the following syntax:

{*column* | *expression* | ROW | REPORT}

and where *action* has the following syntax:

[SKI[P] *n* | [SKI[P]] PAGE] [NODUP[LICATES] | DUP[LICATES]]

BTI[TLE] [*printspec* [*text* | *variable*] ...] | [ON | OFF]

Places and formats a title at the bottom of each report page, or lists the current BTITLE definition. Use one of the following clauses in place of *printspec*:

BOLD
CE[NTER]
COL *n*
FORMAT *text*
LE[FT]
R[IGHT]
S[KIP] [*n*]
TAB *n*

CL[EAR] *option* ...

Resets or erases the current value or setting for the specified option.

Where *option* represents one of the following clauses:

BRE[AKS]
BUFF[ER]
COL[UMNS]
COMP[UTES]
CONTEXT
SCR[EEN]
SQL
TIMI[NG]

COL[UMN] [{*column* | *expr*} [*option* ...]]

Specifies display attributes for a given column, such as:

•   text for the column heading

•   alignment for the column heading

•   format for NUMBER data

•   wrapping of column data

Also lists the current display attributes for a single column or for all columns.

Where *option* represents one of the following clauses:

ALI[AS] *alias*
CLE[AR]
ENTMAP {ON | OFF}
FOR[MAT] *format*
HEA[DING] *text*
JUS[TIFY] {L[EFT] | C[ENTER] | R[IGHT]}
LIKE {*expr* | *alias*}
NEWL[INE]
NEW_V[ALUE] *variable*
NOPRI[NT] | PRI[NT]

NUL[L] text
OLD_V[ALUE] *variable*
ON | OFF
WRA[PPED] | WOR[D_WRAPPED] | TRU[NCATED]

> **✏ Note:**
>
> Currently only NEW_V[ALUE] variable syntax is supported.

Enter COLUMN [{*column* |*expr*} FORMAT *format*] where the *format* element specifies the display format for the column.

To change the display format of a NUMBER column, use FORMAT followed by one of the elements in the following table:

Element	Examples	Description
, (comma)	`9,999`	Displays a comma in the specified position.
. (period)	`99.99`	Displays a period (decimal point) to separate the integral and fractional parts of a number.
$	`$9999`	Displays a leading dollar sign.
0	`0999` `9990`	Displays leading or trailing zeros (0).
9	`9999`	Displays a value with the number of digits specified by the number of 9s. Value has a leading space if positive, a leading minus sign if negative. Blanks are displayed for leading zeros. A zero (0) is displayed for a value of zero.
B	`B9999`	Displays blanks for the integer part of a fixed-point number when the integer part is zero, regardless of zeros in the format model.
C	`C999`	Displays the ISO currency symbol in the specified position.
D	`99D99`	Displays the decimal character to separate the integral and fractional parts of a number.
EEEE	`9.999EEEE`	Displays a value in scientific notation (format must contain exactly four "E"s).
G	`9G999`	Displays the group separator in the specified positions in the integral part of a number.
L	`L999`	Displays the local currency symbol in the specified position.
MI	`9999MI`	Displays a trailing minus sign after a negative value. Displays a trailing space after a positive value.

Element	Examples	Description
PR	`9999PR`	Displays a negative value in <angle brackets>. Displays a positive value with a leading and trailing space.
RN rn	`RN` `rn`	Displays uppercase Roman numerals. Displays lowercase Roman numerals. Value can be an integer between 1 and 3999.
S	`S9999` `9999S`	Displays a leading minus or plus sign. Displays a trailing minus or plus sign.
TM	`TM`	Displays the smallest number of decimal characters possible. The default is TM9. Fixed notation is used for output up to 64 characters, scientific notation for more than 64 characters. Cannot precede TM with any other element. TM can only be followed by a single 9 or E.
U	`U9999`	Displays the dual currency symbol in the specified position.

COMP[UTE] [*function* [LAB[EL] *text*] ... OF {*expr* | *column* | *alias*} ...ON {*expr* | *column* | *alias* | REPORT | ROW} ...]

In combination with the BREAK command, calculates and prints summary lines using various standard computations. It also lists all COMPUTE definitions. The following table lists valid functions. All functions except NUMBER apply to non-null values only. COMPUTE functions are always executed in the following sequence AVG, COUNT, MINIMUM, MAXIMUM, NUMBER, SUM, STD, VARIANCE.

Function	Computes	Applies to Datatypes
`AVG`	Average of non-null values	NUMBER
`COU[NT]`	Count of non-null values	All types
`MIN[IMUM]`	Minimum value	NUMBER, CHAR, NCHAR, VARCHAR2 (VARCHAR), NVARCHAR2 (NCHAR VARYING)
`MAX[IMUM]`	Maximum value	NUMBER, CHAR, NCHAR, VARCHAR2 (VARCHAR), NVARCHAR2 (NCHAR VARYING)
`NUM[BER]`	Count of rows	All types
`SUM`	Sum of non-null values	NUMBER
`STD`	Standard deviation of non-null values	NUMBER
`VAR[IANCE]`	Variance of non-null values	NUMBER

SET SQLFORMAT

SET SQLFORMAT {csv | html | xml | json | ansiconsole | insert | loader | fixed | default}

Outputs reports in various formats. The ansiconsole option formats and resizes data according to the column widths, for easier readability.

SET SQLFORMAT DELIMITED <delimiter> <left enclosure> <right enclosure> allows you to set a custom delimited format.

SPO[OL] [*filename*[.*ext*] [CRE[ATE] | REP[LACE] | APP[END]] | OFF | OUT]

Stores query results in a file, or optionally sends the file to a printer. OFF stops spooling. OUT stops spooling and sends the file to your computer's default printer. Enter SPOOL with no clauses to list the current spooling status. If no file extension is given, the default extension, *.lst* or *.lis*, is used.

TTI[TLE] [*printspec* [*text* | *variable*] ...] | [ON | OFF]

Places and formats a specified title at the top of each report page, or lists the current TTITLE definition. The old form of TTITLE is used if only a single word or a string in quotes follows the TTITLE command.

Where *printspec* represents one or more of the following clauses:

    BOLD
    CE[NTER]
    COL *n*
    FORMAT *text*
    LE[FT]
    R[IGHT]
    S[KIP] [*n*]
    TAB *n*

Accessing Databases

Use the following commands to access and copy data between tables on different databases:

CONN[ECT] [{*<logon>*| / |*proxy*} [AS {SYSOPER | SYSDBA | SYSASM}] [edition=*value*]]

where *logon* has the following syntax:

username[/*password*] [@*connect_identifier*]

where *proxy* has the following syntax:

proxyuser[*username*] [/*password*] [@*connect_identifier*]

> **✎ Note:**
>
> The brackets around *username* in *proxy* are required syntax.

Connects a given username to the Oracle Database. If you omit *connect_identifier*, SQLcl connects you to the default database. If you omit *username* and/or *password*,

SQLcl prompts you for them. CONNECT followed by a slash (/) connects you using a default (OPS$) logon.

When you run a CONNECT command, the site profile, glogin.sql, and the user profile, login.sql, are processed in that order. CONNECT does not reprompt for username or password if the initial connection does not succeed.

DISC[ONNECT]

Commits pending changes to the database and logs the current user out of Oracle, but does not exit SQLcl. In SQLcl command line, use EXIT or QUIT to log out of Oracle and return control to your computer's operating system.

COPY {FROM *database* | TO *database* | FROM *database* TO *database*} {APPEND | CREATE | INSERT | REPLACE | APPEND_BYTE | CREATE_BYTE | REPLACE_BYTE} *destination_table*[(*column*, *column*, *column*, ...)] USING *query*

where *database* has the following syntax:

username[/*password*]@*connect_identifier*

Copies data from a query to a table in the same or another database. APPEND, CREATE, INSERT or REPLACE specifies how COPY treats the existing copy of the destination table (if it exists). USING *query* identifies the source table and determines which rows and columns COPY copies from it. COPY supports CHAR, DATE, LONG, NUMBER and VARCHAR2 datatypes.

PASSW[ORD] [*username*]

Allows you to change a password without displaying it on an input device.

XQUERY *xquery_statement*

Allows you to run an XQuery from SQLcl.

Miscellaneous Commands

ALIAS [<*name*>=<*SQL statement*>;| LOAD [<*filename*>]|SAVE [<*filename*>] | LIST [<*NAME*>] | DROP <*name*> | DESC <*name*> <*Description String*>]

Alias is a command which allows you to save a SQL, PL/SQL or SQL*Plus script and assign it a shortcut command.

• ALIAS — Print a list of aliases

• ALIAS LIST <*alias_name*> — List the contents of the alias

The following example shows how to create a simple alias:

```
SQL> ALIAS action1=select :one from dual;
```

> **✏️ Note:**
>
> Define an alias simply by using the alias keyword followed by a single identifier name followed by an '='. Anything after the '=' will be used as the alias contents. If it is SQL, it will be terminated by ';'. If it is PL/SQL, it will be terminated by '/'.

```
APEX
```

Lists Application Express Applications. Use APEX EXPORT<app id> to export the application which could be combined with spool for writing to a file.

```
ARCHIVE LOG LIST
```

Displays information about redo log files.

```
BRIDGE <targetTableName> as "<jdbcURL>"(<sqlQuery>);
```

Used mainly to script data move between two connections/schemas. It also includes functionality to dynamically create Oracle tables which "fit" the data being received through JDBC. The following functionality is available:

1. Query tables in other connections

2. Query tables in multiple connections in the same statement

3. Insert data from one connection into another

4. Create a table and insert data into it from another connection

```
CTAS table new_table
```

Uses DBMS_METADATA to extract the DDL for the existing table, then modifies that into a create table as select * from.

```
DDL [object_name [type] [SAVE filename]]
```

Generates the code to reconstruct the object listed. Use the *type* option for materialized views. Use the SAVE option to save the DDL to a file.

```
DESC[RIBE] {[schema.]object[@connect_identifier]}
```

Lists the column definitions for a table, view or synonym, or the specifications for a function or procedure.

```
FIND [<filename>]
```

Searches the SQLPATH and its directories for the specified file name. `FIND where <filename>` lists all the SQLPATH locations where it finds files matching the specified file name.

```
INFO[RMATION] {[schema.]object[@connect_identifier]}
```

Lists more detailed information about the column definitions for a table, view or synonym, or the specifications for a function or procedure.

> **✎ Note:**
>
> INFORMATION+ will show column statistics.

```
LOAD [schema.]table_name[@db_link] file_name
```

Loads a comma separated value (csv) file into a table. The first row of the file must be a header row. The columns in the header row must match the columns defined on the table. The columns must be delimited by a comma and may optionally be enclosed in double quotes. Lines can be terminated with standard line terminators for Windows, UNIX, or Mac. File must be encoded UTF8.

The load is processed with 50 rows per batch. If AUTOCOMMIT is set in SQLcl, a commit is done every 10 batches. The load is terminated if more than 50 errors are found.

```
OERR <facility> <error>
```

Displays information about errors. Facility is identified by the prefix string in the error message. For example, if you get ORA-7300, "ora" is the facility and "7300" is the error. So you should type "oerr ora 7300".

REST

REST allows you to export Oracle REST Data Services 3.x services. This is applicable for Oracle REST Data Services release 3.0.5 or later. If you have an earlier version of Oracle REST Data Services, you will need to upgrade. See the Installing Oracle REST Data Services section in *Oracle REST Data Services Installation, Configuration, and Development Guide* for details.

The options are:

- REST export — Export all Oracle REST Data Services 3.x service modules

- REST export *<module_name>* — Export a specific module

- REST export *<module_uri_uri prefix>* — Export a specific module related to the given prefix

- REST modules — List the available modules

- REST privileges — List the existing privileges

- REST schemas — List the available schemas

SODA

SODA allows schemaless application development using the JSON data model. The options are:

- SODA create *<collection_name>* — Create a new collection

- SODA list — List all the collections

- SODA get *<collection_name>* [-all | -f | -k | -klist] [{*<key>* | *<k1> <k2>* ... | *<qbe>*}] — List documents the collection. Optional arguments:

  - all : list the keys of all docs in the collection

- k : list docs matching the specific *<key>*

- klist : list docs matching the list of keys

- f : list docs matching the *<qbe>*

- SODA insert *<collection_name> <json_str | filename>* — Insert a new document within a collection

- SODA drop *<collection_name>* — Delete existing collection

- SODA count *<collection_name>* [*<qbe>*] — Count number of documents inside collection. Optional parameter *<qbe>* returns number of matching documents

- SODA replace *<collection_name> <oldkey> <new_*{str | doc}*>* — Replace one document with another

- SODA remove *<collection_name>* [-k | -klist | -f] {*<key>* | *<k1> <k2>* ...| *<qbe>*} — Remove documents from collection. Optional arguments:

  - k : Remove document in collection matching the specific *<key>*

  - klist : Remove document in collection matching the list *<key1> <key2>* ...

  - f : Remove document in collection matching *<qbe>*

SET *system_variable value*

Sets a system variable to alter the SQLcl environment settings for your current session. For example, to:

- Set the display width for data

- Customize HTML formatting

- Enable or disable printing of column headings

- Set the number of lines per page

Enter a system variable followed by a value as shown below:

```
SET APPI[NFO]{ON | OFF | text}
SET ARRAY[SIZE] {15 | n}
SET AUTO[COMMIT] {ON | OFF | IMM[EDIATE] | n}
SET AUTOP[RINT] {ON | OFF}
SET AUTORECOVERY {ON | OFF]
SET AUTOT[RACE] {ON | OFF | TRACE[ONLY]}
SET BLO[CKTERMINATOR] {. | c | ON | OFF}
SET CLEAR [ TOP | BOTTOM | SAME ]
SET CMDS[EP] {; | c | ON | OFF}
SET COLSEP {_ | text}
SET CON[CAT] {. | c | ON | OFF}
SET COPYC[OMMIT] {0 | n}
SET COPYTYPECHECK {ON | OFF}
SET DDL [[ PRETTY | SQLTERMINATOR | CONSTRAINTS |
REF_CONSTRAINTS | CONSTRAINTS_AS_ALTER|OID |
SIZE_BYTE_KEYWORD | PARTITIONING | SEGMENT_ATTRIBUTES |
STORAGE | TABLESPACE | SPECIFICATION | BODY | FORCE | INSERT | |
INHERIT | RESET] {on|off} ] | OFF ]
```

SET DEF[INE] {& | *c* | <u>ON</u> | OFF}

SET ECHO {ON | <u>OFF</u>}

SET EDITF[ILE] *file_name*[.*ext*]

SET EMB[EDDED] {ON | <u>OFF</u>}

SET ENCODING

SET ERRORL[OGGING] {ON | <u>OFF</u>} [TABLE [*schema.*]*tablename*] [TRUNCATE]
[IDENTIFIER *identifier*]

SET ESC[APE] {\ | *c* | ON | <u>OFF</u>}

SET ESCCHAR {@ | ? | % | $ | <u>OFF</u>}

SET EXITC[OMMIT] {<u>ON</u> | OFF}

SET [EXP[LAIN]] [STAT[ISTICS]]

SET FEED[BACK] {<u>6</u> | *n* | <u>ON</u> | OFF}

SET FLU[SH] {<u>ON</u> | OFF}

SET HEA[DING] {<u>ON</u> | OFF}

SET HEADS[EP] {<u>|</u> | *c* | <u>ON</u> | OFF}

SET INSTANCE [*instance_path* | <u>LOCAL</u>]

SET LDAPCON

SET LIN[ESIZE] {<u>80</u> | *n*}

SET LOBOF[FSET] {*n* | <u>1</u>}

SET LOGSOURCE [*pathname*]

SET LONG {<u>80</u> | *n*}

SET LONGC[HUNKSIZE] {<u>80</u> | *n*}

SET NET {ON | <u>OFF</u> | READONLY}

SET NEWP[AGE] {<u>1</u> | *n* | NONE}

SET NOVERWRITE {ON | <u>OFF</u> | WARN}

SET NULL *text*

SET NUMF[ORMAT] *format*

SET NUM[WIDTH] {<u>10</u> | *n*}

SET PAGES[IZE] {<u>14</u> | *n*}

SET PAU[SE] {ON | <u>OFF</u> | *text*}

SET RECSEP {<u>WR[APPED]</u> | EA[CH] | OFF}

SET RECSEPCHAR {<u> </u> | *c*}

SET SERVEROUT[PUT] {ON | OFF} [SIZE {*n* | <u>UNL[IMITED]</u>}] [FOR[MAT]
{WRA[PPED] | WOR[D_WRAPPED] | TRU[NCATED]}]

SET SHIFT[INOUT] {VIS[IBLE] | <u>INV[ISIBLE]</u>}

SET SHOW[MODE] {ON | <u>OFF</u>}

SET SQLBL[ANKLINES] {ON | <u>OFF</u>}

SET SQLC[ASE] {<u>MIX[ED]</u> | LO[WER] | UP[PER]}

SET SQLCO[NTINUE] {<u>></u> | *text*}

SET SQLFORMAT {csv | html | xml | json | ansiconsole | insert | loader | fixed |
default}

SET SQLN[UMBER] {<u>ON</u> | OFF}

SET SQLPLUSCOMPAT[IBILITY] {*x.y*[.*z*]}

SET SQLPRE[FIX] {<u>#</u> | *c*}

SET SQLP[ROMPT] {<u>SQL></u> | *text*}

SET SQLT[ERMINATOR] {<u>;</u> | *c* | <u>ON</u> | OFF}

SET SUF[FIX] {<u>SQL</u> | *text*}

SET TAB {ON | OFF}
SET TERM[OUT] {ON | OFF}
SET TI[ME] {ON | OFF}
SET TIMI[NG] {ON | OFF}
SET TRIM[OUT] {ON | OFF}
SET TRIMS[POOL] {ON | OFF}
SET UND[ERLINE] {- | c | ON | OFF}
SET VER[IFY] {ON | OFF}
SET WRA[P] {ON | OFF}

SET DDL [[PRETTY | SQLTERMINATOR | CONSTRAINTS | REF_CONSTRAINTS | CONSTRAINTS_AS_ALTER|OID | SIZE_BYTE_KEYWORD | PARTITIONING | SEGMENT_ATTRIBUTES | STORAGE | TABLESPACE | SPECIFICATION | BODY | FORCE | INSERT | |INHERIT | RESET] {on|off}] | OFF]

Allows you to set the DDL transform option on DBMS_METADATA.

SET ENCODING <encoding>

Allows you to set the encoding for the current session. Use SHOW ENCODING to view the encoding set for the current session. Use SHOW ENCODINGS to list the encodings available on your platform.

SHO[W] [option]

Shows the value of a SQLcl system variable, or the current SQLcl environment. Enter any system variable set by the SET command in place of system_variable. SHOW SGA can only be used by a DBA user. Use one of the following terms or clauses in place of option:

system_variable
ALL
BTI[TLE]
CON_ID
CON_NAME
CONNECTION
DDL
EDITION
ENCODING
ENCODINGS
ERR[ORS] [{FUNCTION | PROCEDURE | PACKAGE | PACKAGE BODY | TRIGGER | VIEW | TYPE | TYPE BODY | DIMENSION | JAVA CLASS} [schema.]name]
INSTANCE
JAVA
JDBC
LNO
NLS
PARAMETER[S] [parameter_name]
PDBS
PNO

RECYC[LEBIN] [*original_name*]
REL[EASE]
REPF[OOTER]
REPH[EADER]
SGA
SPOO[L]
SPPARAMETER[S] [*parameter_name*]
SQLCODE
SQLPATH
TNS
TTI[TLE]
USER
VERSION

SHOW ENCODING

Shows the encoding which is set for the client.

SHOW ENCODINGS

Shows the available encodings for the client.

SSHTUNNEL *<username>@<hostname>* -i *<identity_file>* [-L
localPort:Remotehost:RemotePort]

Creates a tunnel using standard ssh options such as port forwarding like option -L of
the given port on the local host will be forwarded to the given remote host and port on
the remote side. It also supports identity files, using the ssh -i option. If passwords are
required, they will be prompted for.

TNSPING *<address>*

The TNSPING utility determines whether the listener for a service on an Oracle Net
network can be reached successfully.

WHICH

Searches the SQLPATH and its directories for the specified file name and prints the
name of the first file matching the specified file name in the SQLPATH.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle
Accessibility Program website at http://www.oracle.com/pls/topic/lookup?
ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support
through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/
lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs
if you are hearing impaired.