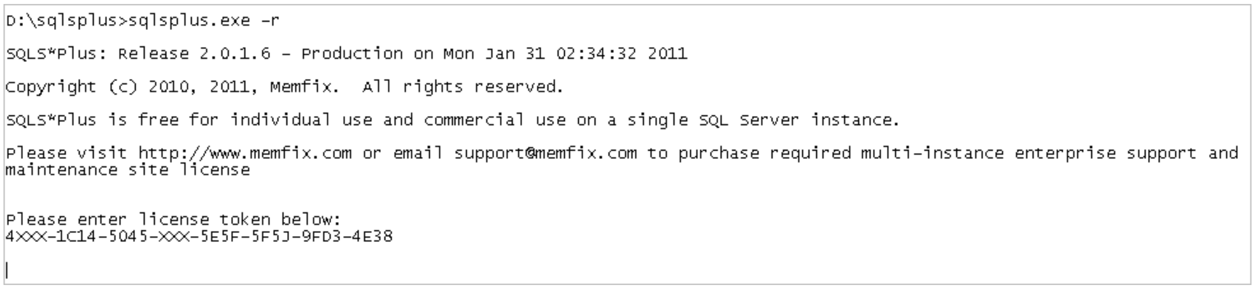
SQLS*Plus for SQL Server

Contents

Chapter 1

Register License:

Issue “sqlsplus.exe –r” command and paste your license token to register SQLS*Plus



```
D:\sqlsplus>sqlsplus.exe -r

SQLS*Plus: Release 2.0.1.6 - Production on Mon Jan 31 02:34:32 2011

Copyright (c) 2010, 2011, Memfix.  All rights reserved.

SQLS*Plus is free for individual use and commercial use on a single SQL Server instance.

Please visit http://www.memfix.com or email support@memfix.com to purchase required multi-instance enterprise support and
maintenance site license


Please enter license token below:
4XXX-1C14-5045-XXX-5E5F-5F5J-9FD3-4E38

|
```

Startup Scripts

When SQLS*Plus starts, and after CONNECT commands, the two sql files are being executed:

1) login.ssp - SQLS*Plus profile

2) login.sql - User profile

The files may contain SQLS*Plus commands.

Connecting to database server

Database server connect command:

“connect username/password@server\instance:database”

When connecting from inside database session use \\ to prefix instance name:

“connect username/password@server\\instance:database”

Example:

D:\ >sqlsplus sa/password@192.168.1.160 SQLS*Plus: Release 2.0.1.8 - Production on Tue Nov 1 17:07:45 2011 Copyright (c) 2010, 201x, Memfix. All rights reserved. SQLS*Plus is free for an individual use and a commercial use on a single SQL Server instance. Please visit http://www.memfix.com or email support@memfix.com to purchase required… Connected to: Microsoft SQL Server RTM, version 9.00.1399.06, Developer Edition (64-bit), current database: tempdb	**Connect to SQL Server instance using database username and password**

0:sa@192.168.1.160> show dbs;
  Database Name
  AdventureWorks
  BusinessServiceIISRepository
	“show“ command (sqlsplus) to list all accessible databases

BusinessServiceRepository CapacityPlanner CompositeWebAppRepository master model msdb OrderProcessorForwardRepository OrderProcessorRepository StockTraderDB StockTraderWebAppRepository tempdb 0:sa@192.168.1.160> use AdventureWorks; 0:sa@192.168.1.160> show db; database is "AdventureWorks" 0:sa@192.168.1.160>	**“use” command (SQL) to select database** **“show“ command (sqlsplus) to show current database**

Set default database for connection:

1) Command line:

   sqlsplus.exe sa/<pwd>@192.168.1.160:AdventureWorks

   or

   sqlsplus.exe sa/<pwd>@192.168.1.160\ SQLSERVER2008:AdventureWorks

2) SQLSDBNAME environment variable

   SET SQLSDBNAME=AdventureWorks

3) SQLCMDDBNAME environment variable (sqlcmd variable)

   SET SQLCMDDBNAME =AdventureWorks

Connecting with trusted connection / windows authentication

1) Connect from command line:

   sqlsplus.exe –E

   – connect to default local database instance

   or

   sqlsplus.exe -E@HOST\SQLSERVER2008

   – connect to specified remote database instance
   or

   sqlsplus.exe -E@HOST\SQLSERVER2008: AdventureWorks

   – connect to specified remote database instance and database

2) Connect from SQLS*Plus session

   connect -E

   connect -E@HOST\\SQLSERVER2008

   connect -E@HOST\\SQLSERVER2008: AdventureWorks

Connecting with username or password that contains '@'

Unless password is entered interactively, prefix @ with \.

Example:

connect user/pass\@word@ HOST\\SQLSERVER2008

When connecting from inside database session use \\ to prefix @:

0:sa@server\SQLSERVER2008> connect user/pass\\@word@ HOST\\SQLSERVER2008

Start SQLS*Plus with no connection

Use "/nolog" to start SQLS*Plus without connecting to database

This option is useful if connect statement is in the script and for security reasons should not be externalized in command line

For Example:

```
sqlsplus /nolog @t4
```

t4.sql:

```
connect sa/xxxx@prodsrvr1;

set pages 0;
use tempdb;
db;

define tbl = sys.objects;
select count(*) c1 from &tbl;
quit
```

Connectivity environmental variables

1) SQLSUSER / SQLCMDUSER
   Default connect user

2) SQLSPASSWORD / SQLCMDPASSWORD
   Default connect user password

3) SQLSSERVER / SQLCMDSERVER
   Default SQL server host and instance

4) SQLSDBNAME / SQLCMDDBNAME
   Default database to connect to

5) SQLSPATH / SQLPATH
   Environment variables that specify search locations of the SQL scripts. SQLS*Plus searches for the SQL scripts, including “login.ssp” and “login.sql”, starting from the current directory and after that in the directories specified by SQLSPATH first and SQLPATH after it. SQLSPATH and SQLPATH is a semicolon separated list of directories.

Batch execution of SQL Scripts

Make sure to use double slashes (“\\”) in the path, i.e “d:\\x1.sql”

You can call batch sql file as below:

sqlsplus.exe sa/<pwd>@192.168.1.160 @d:\\x1.sql

In this example we connect to default instance of SQL Server on a server and execute sql script x1.sql.

or

sqlsplus.exe sa/<pwd>@192.168.1.160\SQLSERVER2008 @d:\\x1.sql

In second case we connect to specific instance (in case there are more than one)

Sample x1.sql content - includes "quit" command to insure that program quits after script execution

```
--
set pages 200
set lines 200
select * from master.dbo.sysprocesses;
quit
```

Execute script with no connectivity information on a command line

Use “/nolog” on SQLS*Plus command line and include “connect” command into the SQL script

For example:

sqlsplus.exe /nolog @x1.sql

Note: x1.sql contains connect command, i.e.:

”connect sa/<pwd>@192.168.1.160”

Multiple database sessions support

SQL> connect sa/xxxx@192.168.1.160	**Connect to database**
Connected to: Microsoft SQL Server RTM, version 9.00.1399.06, Developer Edition (64-bit), current database: tempdb 0:sa@192.168.1.160> 0:sa@192.168.1.160>	
0:sa@192.168.1.160> connect -l 0 sa@192.168.1.160	**List current sessions (1)**
0:sa@192.168.1.160> help connect =====SessionMgr===== connect - create a new session disconnect - disconnect session nr try help -v or <command> -h for detailed help.	**Help on “connect”**
0:sa@192.168.1.160> help -v connect =====SessionMgr===== NAME connect - create a new session SYNOPSIS connect <connectString> alias: conn|c -a create additional session, don't disconnect -l list connections -s sessNo switch to sessNo (as reported by -l) DESCRIPTION Connect creates a database session. You can have multiple sessions at a time, which may save you some time, when you need to switch sessions frequently. The sqlsplus is useful when you run sqlsplus from within emacs and you started it with a bad connect string. NAME disconnect - disconnect session nr SYNOPSIS disconnect <sessionNr> alias: dis try help -v or <command> -h for detailed help.	**Detailed help on “connect”**
0:sa@192.168.1.160> connect -l 0 sa@192.168.1.160	**List current sessions (1)**
0:sa@192.168.1.160> connect -a sa/xxx@192.168.1.170 Connected to: Microsoft SQL Server RTM, version 9.00.1399.06, Developer Edition (64-bit), current database: tempdb	**Connect as additional session to new database**

1:sa@192.168.1.170> connect -l	**List current sessions (2)**
0 sa@192.168.1.160	
1 sa@192.168.1.170	
1:sa@192.168.1.170> connect -s 0	**Switch to session 0**
0:sa@192.168.1.160> connect -s 1	**Switch back to session 1**
1:sa@192.168.1.170> quit	

Multi-line SQLS*Plus commands

The SQLS*Plus commands can span multiple lines, as long as dash “-‘ is used at the end of each continuing line.

For example:

```
TTITLE LEFT 'User Report' -
> RIGHT 'PAGE:' -
>  SQL.PNO SKIP 2
```

Special data selection functionality

- Vertical Output – allows to see large column sets as a vertical output

  “set vout on”

- Table data “grep” – search for data across all columns

- Data purge – purge table data in a small chunks

Chapter 2

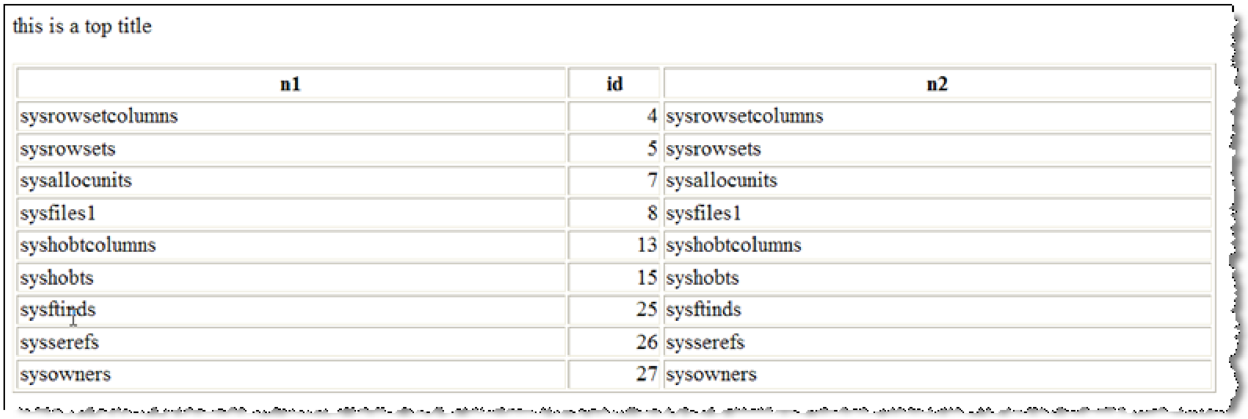
HTML Data Output

Use “set markup html on|off” command to output data in HTML format

Sample SQL script for HTML output:

```
D:\sqlsplus>type t2.sql

set pages 10

ttitle LEFT 'this is a top title'

set markup html on
spool xx.htm

select top 25 name n1, id, name n2 from sysobjects;

spool off

set markup html off

host xx.htm
```

HTML output:



this is a top title

n1	id	n2
sysrowsetcolumns	4	sysrowsetcolumns
sysrowsets	5	sysrowsets
sysallocunits	7	sysallocunits
sysfiles1	8	sysfiles1
syshobtcolumns	13	syshobtcolumns
syshobts	15	syshobts
sysftinds	25	sysftinds
sysserefs	26	sysserefs
sysowners	27	sysowners

CSV Data Output

Use “set output csv” command to output data in CSV format

Sample SQL script for CSV output:

```
0:sa@192.168.1.160> set output csv
0:sa@192.168.1.160> set head off
0:sa@192.168.1.160> set pages 0
```

CSV output:

```
0:sa@192.168.1.160> select name,crdate from sys.sysobjects;
"sysrowsetcolumns","2005-10-14 01:36:15.923",
"sysrowsets","2005-10-14 01:36:15.910",
"sysallocunits","2005-10-14 01:36:15.910",
"sysfiles1","2003-04-08 09:13:38.093",
"syshobtcolumns","2005-10-14 01:36:15.940",
"syshobts","2005-10-14 01:36:15.923",
"sysftinds","2005-10-14 01:36:17.063",
"sysserefs","2005-10-14 01:36:15.940",
"sysowners","2005-10-14 01:36:17.050",
"sysprivs","2005-10-14 01:36:15.877",
"sysschobjs","2005-10-14 01:36:15.987",
…
```

JSON Data Output

Use “set output json” command to output data in JSON format

Sample SQL script for JSON output:

```
0:sa@192.168.1.160> set output json
0:sa@192.168.1.160> set head off
0:sa@192.168.1.160> set pages 0
```

JSON output:

```
0:sa@192.168.1.160\SQLSERVER2008> select * from department order by dept_id;
{ "dept_id" : "10", "last_name" : "Jackson", "salary" : "50000", "bonus" : "12501.78" }
{ "dept_id" : "10", "last_name" : "Sally", "salary" : "55000", "bonus" : "13750" }
{ "dept_id" : "10", "last_name" : "Major", "salary" : "30000", "bonus" : "7500" }
{ "dept_id" : "10", "last_name" : "Mimon", "salary" : "38000", "bonus" : "9500" }
{ "dept_id" : "10", "last_name" : "Karla", "salary" : "58000", "bonus" : "14500" }
{ "dept_id" : "10", "last_name" : "Major", "salary" : "34000", "bonus" : "8500" }
{ "dept_id" : "10", "last_name" : "Mason", "salary" : "39000", "bonus" : "9750" }
{ "dept_id" : "10", "last_name" : "Mason", "salary" : "39000", "bonus" : "9750" }
{ "dept_id" : "10", "last_name" : "Jackson", "salary" : "50000", "bonus" : "12501.78" }
{ "dept_id" : "10", "last_name" : "Sally", "salary" : "55000", "bonus" : "13750" }
{ "dept_id" : "10", "last_name" : "Major", "salary" : "30000", "bonus" : "7500" }
{ "dept_id" : "10", "last_name" : "Mimon", "salary" : "38000", "bonus" : "9500" }
{ "dept_id" : "10", "last_name" : "Karla", "salary" : "58000", "bonus" : "14500" }
{ "dept_id" : "10", "last_name" : "Major", "salary" : "34000", "bonus" : "8500" }
{ "dept_id" : "10", "last_name" : "Mason", "salary" : "39000", "bonus" : "9750" }
{ "dept_id" : "10", "last_name" : "Mason", "salary" : "39000", "bonus" : "9750" }
{ "dept_id" : "20", "last_name" : "Smith", "salary" : "45000", "bonus" : "23000" }
{ "dept_id" : "20", "last_name" : "<NULL>", "salary" : "65000", "bonus" : "29000" }
{ "dept_id" : "20", "last_name" : "Major", "salary" : "78000", "bonus" : "" }
{ "dept_id" : "20", "last_name" : "Smith", "salary" : "75000", "bonus" : "18750" }
{ "dept_id" : "20", "last_name" : "Jefferson", "salary" : "90000", "bonus" : "22500" }
{ "dept_id" : "20", "last_name" : "Smith", "salary" : "45000", "bonus" : "23000" }
{ "dept_id" : "20", "last_name" : "<NULL>", "salary" : "65000", "bonus" : "29000" }
{ "dept_id" : "20", "last_name" : "Major", "salary" : "78000", "bonus" : "" }
{ "dept_id" : "20", "last_name" : "Smith", "salary" : "75000", "bonus" : "18750" }
{ "dept_id" : "20", "last_name" : "Jefferson", "salary" : "90000", "bonus" : "22500" }
{ "dept_id" : "30", "last_name" : "Sandy Jackson", "salary" : "38000", "bonus" : "19000" }
….
```

Vertical Data Output

Use “set vout on” command to output data in vertical format, where each column is printed on its own line. Vertical output format is helpful when outputting data from a tables with many columns

Sample SQL script for CSV output:

```
0:sa@192.168.1.160> set vout on
```

Vertical output:

```
0:sa@192.168.1.160> select name,crdate from sys.sysobjects;

name  | sysrowsetcolumns
crdate| 2005-10-14 01:36:15.923

name  | sysrowsets
crdate| 2005-10-14 01:36:15.910

name  | sysallocunits
crdate| 2005-10-14 01:36:15.910

name  | sysfiles1
crdate| 2003-04-08 09:13:38.093
…
```

Column Autoformatting

Use “set autoformat <table>” command to automatically format table columns to optimally size column sizes for character and numeric fields

set autoformat supports 2 parameters:

maxsize – defined maximum size for long character columns, default is 40 characters
sample – defined sample size for table data selection to identify optimal columns sizes, default is 5%

Sometime default sample is not enough and for small table recommendation is to set sample to 50%-100%

Sample table columns autoformatting

```
0:sa@192.168.1.160\SQLSERVER2008> set autoformat SalesLT.Customer

Unable to create automatic column formatting. Please increase sample size and retry

0:sa@192.168.1.160\SQLSERVER2008> set autoformat SalesLT.Customer sample 50
0:sa@192.168.1.160\SQLSERVER2008> col
COLUMN   MIDDLENAME
FORMAT   A10

COLUMN   CUSTOMERID
FORMAT   9999999999

COLUMN   PASSWORDHASH
FORMAT   A40

COLUMN   SALESPERSON
FORMAT   A24

COLUMN   COMPANYNAME
FORMAT   A36

COLUMN   PASSWORDSALT
FORMAT   A12

COLUMN   TITLE
FORMAT   A5

COLUMN   LASTNAME
FORMAT   A22

COLUMN   FIRSTNAME
FORMAT   A15

COLUMN   SUFFIX
FORMAT   A6

COLUMN   EMAILADDRESS
FORMAT   A34

COLUMN   PHONE
FORMAT   A19
```

Chapter 3

Passing parameters as script arguments

You can bypass the prompts for values associated with substitution variables by passing values to parameters in a script through the START / @ command.

Placing an ampersand (&) followed by a numeral in the script in place of a substitution variable. Each time script is executed, value of “&<N>” is replaced with the corresponding command line argument after @*filename*

Use of variables

& and && indicate substitution variables in SQLS*Plus scripts or commands

When SQLS*Plus encounters a variable defined with &&, it prompts you for the value and then uses this value for every subsequent occurrence of that variable it encounters. The variable and its value are stored.

When you define a variable with &, however, SQLS*Plus discards the variable and its value immediately after use, so that repeated use of &<variablename> results in repeated prompts for the value of <variablename>.

Bind variables

Bind variables are variables created in SQLS*Plus and then used in T-SQL or SQL.

Bind variables can be displayed in SQLS*Plus or referenced in T-SQL subprograms that run in SQLS*Plus.

Creating bind variables

Bind variables created in SQLS*Plus with the VARIABLE command. For example

```
VARIABLE v_table_name VARCHAR(50) –s “MY_TABLE”
```

This command creates a bind variable named v_table_name with a datatype of VARCHAR and initial value of “MY_TABLE”.

For more information, see the VARIABLE command. (To list session bind variables, type VARIABLE without arguments.)

Referencing bind variables

Bind variables in T-SQL referenced by typing a colon (:) followed immediately by the name of the variable. For example

```
SET @Table_Name = :v_table_name;
```

```
0:sa@192.168.1.160\SQLSERVER2008> var name varchar(20) -s "This is a variable"

0:sa@192.168.1.160\SQLSERVER2008> /
begin
DECLARE @Name VARCHAR(20)
SET @Name = :name
print @Name
end

This is a variable
```

Displaying bind variables

To display the value of a bind variable in SQLS*Plus, use the SQLS*Plus PRINTVAR command. For example:

```
PRINTVAR name

0:sa@192.168.1.160\SQLSERVER2008> PRINTVAR name

:name
------------------
This is a variable
```

Setting bind variables values directly

To set the value of a bind variable directly in SQLS*Plus, use the SQLS*Plus SETVAR command. For example:

```
0:sa@192.168.1.160\SQLSERVER2008> SETVAR name "NEW_ORDERS"
0:sa@192.168.1.160\SQLSERVER2008> PRINTVAR name

:name
----------
NEW_ORDERS
```

Using bind variables values in non-SQL/TSQL report elements

Bind variable can be used on TTITLE and BTITLE.

For example:

```
0:sa@192.168.1.160\SQLSERVER2008> setvar v3 @@servername;
0:sa@192.168.1.160\SQLSERVER2008>
0:sa@192.168.1.160\SQLSERVER2008> var v2 varchar(10) -s "Title Header"
0:sa@192.168.1.160\SQLSERVER2008>
0:sa@192.168.1.160\SQLSERVER2008> ttitle ':v2 :v3'

0:sa@192.168.1.160\SQLSERVER2008> select top 5 name from sysobjects;


                    'Title Header ADMIN-PC\SQLSERVER2008
name
--------------------------------------------------------------------------------
sysrscols
sysrowsets
sysallocunits
sysfiles1
syspriorities
0:sa@192.168.1.160\SQLSERVER2008> var

Currently defined bind variables:

var         length   value
-------------------------------------------------
:v2         10      Title Header
:v3         22      ADMIN-PC\SQLSERVER2008
```

Assigning SQL Server global variables to bind variables

SQL Server global variable value can be assigned to bind variable duing the time of creation of later using VARIABLE and SETVAR commands

For Example:

```
VARIABLE v3 varchar(40) –s @@ servername
```

or

```
SETVAR v3 @@servername;
```

Define Variables

Define variables contain either pre-defined value, such as database use or connection string or can be set by user manually or programmatically using COLUMNS and NEW_VALUE option of the columns

Defining and manually assigning values to define variables

```
DEFINE Variable = 'value’
```

Example:

```
DEFINE LastName = 'Jackson’
```

Programmatically assigning values to define variables

1) Define variable <define>

2) Define column with new_value <define>

3) Select data into column from table

Example:

```
DEFINE LName = 'Jackson’

COLUMN LastName new_value LName

select ‘Olson’ LastName;
```

Pre-defined variables

_CONNECT_IDENTIFIER
Connection identifier used to make connection.

_CONNECT_DATABASE
Database used to make connection, where available.

_DATE
Current date in default system format

_EDITOR
Editor used by the EDIT command.

_LANGUAGE
Language set in database (as “select @@language”)

_LOGON
Database or OS logon user name used to make connection.

_PRIVILEGE
Privilege level of the current connection (SYSADMIN or not)

_S_EDITION
Database edition of the connected SQL Server Database

_S_VERSION
Version of the connected SQL Server Database.

_S_LEVEL
Level of the connected SQL Server Database.

_USER
Database schema name used to make connection.

Use of define variables in SQLS*Plus command prompt

Define variables can be used to customize SQLS*Plus command prompt

Example:

```
0:sa@192.168.1.160\SQLSERVER2008> set sqlprompt
"_USER@_CONNECT_DATABASE>"

sa@AdventureWorksLT2008>set sqlprompt reset

0:sa@192.168.1.160\SQLSERVER2008>
```

Chapter 4

List of SQLS*Plus Commands

Command	Usage	Description
&	&<variable name>	Use substitution variable
&&	&&<variable name>	Use substitution variable
/		Executes the SQL command or batch currently stored in the SQL buffer
ACCEPT|ACC	ACC[EPT] variable [DEF[AULT] default] [PROMPT text| NOPR[OMPT]] [HIDE]	Reads keyboard input and stores it into the SQLS*Plus variable (listed by DEFINE command) *variable* Name of the variable If variable does not exist, SQLS*Plus creates it. *DEF[AULT]* Use set default value if a reply is not provided. *PROMPT text* Skip a line and display text before accepting the variable value *NOPR[OMPT]* Skip a line and wait for an input without prompt. *HIDE* Suppress the display of the typed characters
AGAIN |!!	Again | !!<number>	Rerun latest matching entry or command from history
BREAK|BRE	BREAK [ON <COLUMN|REPORT|ROW>] <DUP|NODUP> <SKI[P] N|PAGE>	ON specifies on what column to track value changes that occur in the report and the formatting action to perform ON REPORT specifies that break related action (SKIP

		lines or calculate COMP values) will be done at the end of the report ON ROW executes SKIP action (if specified) after every row DUP|NODUP NODUP prints blanks for duplicate values in break column DUP prints the value of a break column in every row SKIP skips specified number of lines or whole page on break BREAK is needed for COMP statements to work. COMP expects to have BREAK columns defined for ON part of the COMP statement
BTITLE	**Syntax** BTI[TLE] [*printspec* [*text* | *variable*] ...] | [ON | OFF] where *printspec* represents one or more of the following clauses used to place and format the text: CE[NTER] LE[FT] R[IGHT] S[KIP] [*n*] COL [*n*] TAB [*n*] Enter BTITLE with no clauses to list the current BTITLE definition. **Options** See the **TTITLE** command for information on terms and	Places and formats a specified title at the bottom of each report page, or lists the current BTITLE definition. **See the TTITLE command syntax for information on terms and clauses in the BTITLE command syntax.**

<table>
<tr><td></td><td>clauses in the BTITLE command syntax.

<b>Examples</b>

<code>BTITLE LEFT 'REPORT ' RIGHT ‘PAGE:’ SQL.PNO</code></td><td></td></tr>
<tr><td><b>CAT</b></td><td>cat &lt;table_name&gt; -a</td><td>Synonym for “select * from <table/view name”

Cat <table/view name>

Selects first 3000 rows

Options: -a – select all rows</td></tr>
<tr><td><b>CD</b></td><td>cd &lt;directory_name&gt;</td><td>Change current directory location</td></tr>
<tr><td><b>CLEAR</b></td><td>clear &lt;BREAKS|COMPUTES|SCREEN&gt;</td><td>BRE[AKS] – clears all defined breaks

COMP[UTES] – clears all defined computes

SCR[EEN] – clear screen</td></tr>
<tr><td><b>COLUMN | COL</b></td><td>col &lt;name> heading <name> format <format> <ON|OFF> <PRINT|NOPRINT>

<b><i>Supported Character Formats</i></b>

To change the width of a character field to <i>n</i>, use FORMAT A<i>n</i>. (“A” for alphabetic.)

<b><i>Supported Number Formats</i></b>

<table>
<tr><th>Element</th><th>Examples</th><th>Description</th></tr>
<tr><td>,
(comma)</td><td><code>9,999</code></td><td>Displays a comma</td></tr>
<tr><td>.
(period)</td><td><code>99.99</code></td><td>Displays a period (decimal point) to separate the integral and fraction of a number.</td></tr>
</table>
</td><td>Set column format

ON|OFF – turn off/on column format attributes

PRINT|NOPRINT – show or hide column from the query output</td></tr>
</table>

<table>
<tr>
<td></td>
<td>
<table>
<tr><td>$</td><td><code>$9999</code></td><td>Displays a leading dollar sign.</td></tr>
<tr><td>0</td><td><code>0999 9990</code></td><td>Displays leading zeros and trailing zeros.</td></tr>
<tr><td>9</td><td><code>9999</code></td><td>Displays a value with the number of digits specified by the number of 9s.</td></tr>
</table>
</td>
<td></td>
</tr>
<tr>
<td><b>COMPUTE|COMP</b></td>
<td>
COMP[UTE] [function [LAB[EL] text] ... OF {column } ... ON {column}

Supported functions:

<table>
<tr><td>Function</td><td>Description</td><td>Datatypes</td></tr>
<tr><td>AVG</td><td>Non-null values average</td><td>Numeric</td></tr>
<tr><td>COUNT|COU</td><td>Non-null values count</td><td>All</td></tr>
<tr><td>MINIMUM|MIN</td><td>Minimum value</td><td>Numeric & character</td></tr>
<tr><td>MAXIMUM|MAX</td><td>Maximum value</td><td>Numeric & character</td></tr>
<tr><td>NUMBER|NUM</td><td>Row count</td><td>All</td></tr>
<tr><td>SUM</td><td>Non-null values sum</td><td>Numeric</td></tr>
<tr><td>STD</td><td>Non-null values standard deviation (for population)</td><td>Numeric</td></tr>
<tr><td>VARIANCE|VAR</td><td>Non-null values variance (for population)</td><td>Numeric</td></tr>
</table>
</td>
<td>
COMP in combination with the BREAK, calculates and prints summary function values, based on groups defined by BREAK.

COMP with no parameters lists all current COMPUTE definitions

LABEL Defines the label for the computed function value

OF {column}

OF defines columns that is used to calculate value summary of function

ON {column}

ON defines columns that is used to group values summary functions around.

Corresponding BREAK ON columns must exists for ON clause in COMP to work

If multiple COMPUTE commands use the same ON column, only the last ON column would be used.
</td>
</tr>
<tr>
<td><b>CONNECT | CONN</b></td>
<td>connect user/password@ip[\\instance_name][:db</td>
<td>Create a new session</td>
</tr>
</table>

<table>
<tr><td></td><td>_name]
<table><tr><td>-a</td><td>create additional session, don't disconnect</td></tr><tr><td>-l</td><td>List all sessions, don't disconnect</td></tr><tr><td>-s <sess_num></td><td>switch to session (as reported by -l)</td></tr></table></td><td>Connect creates a database session. Multiple sessions can exists at the same time
Instance name and database names are optional
Instead of db_name it is possible to use below environmental variables to set up default database for connection:
<b>a) SQLSDBNAME</b>
b) <b>SQLCMDDBNAME</b> (standard sqlcmd variable)</td></tr>
<tr><td><b>COUNT</b></td><td></td><td>Count rows in the tables</td></tr>
<tr><td><b>DEFINE</b></td><td>DEF[INE] [<i>variable</i>]|[<i>variable</i> = <i>text</i>]
-l list defines
Below are the pre-defined variables
_CONNECT_IDENTIFIER
Connection identifier used to make connection.
_CONNECT_DATABASE
Database used to make connection, where available.
_EDITOR
Editor used by the EDIT command.
_S_VERSION
Version of the connected SQL Server Database.
_S_LEVEL
Level of the connected SQL Server Database.
_USER
User name used to make connection.</td><td>Define a variable and assigns a value to it, or lists the value and variable type of a single variable or all variables</td></tr>
</table>

CTAS	CTAS source_table destination_table empty_flag	Create destination table as select from the source table
DEPS	DEPS [NAME] -r list dependencies recursively	Object dependencies and references
DESCRIBE | DESC | DE	DESCRIBE [OBJECT_NAME|SCHEMA.OBJECT_NAME] [detail]	Describe a table, view or a stored procedure “detail” option provides detailed describe information
DIR		List file directory
DISCONNECT		Disconnect session
EDIT | ED	ED | ED <sql_file_name>	Edit current statement or sql script file Default editor is “notepad” Environment variable SQLSEDIT can be used to set custom editor
EXEC		Execute T-SQL procedure
FIND		Find a line in T-SQL procedure source
GREP	grep <pattern table [extra clause]> -v show rows that don't match pattern -I ignore case Search pattern across all table columns	Show rows that match pattern
HEAD		Show first rows of table
HELP		Provide help for a command
HISTORY | HIST | HI		Show history items matching pattern (or all)
HOST | HOS | HO		Execute host OS command
ID		Display current user and login
LIST		List last sql statement
LS		List all objects matching pattern

PAUSE	SET PAUSE <TEXT>
	SET PAUSE [ON|OFF]
		Enables to control scrolling of terminal when executing reports. First step is to "SET PAUSE text", and then "SET PAUSE ON" to make text to appear each time SQLS*Plus pauses.

PRINTVAR		Print bind variables
PROMPT		Sends the specified message

		or a blank line to the user's screen
PURGE	purge <table where ...> -c print table count at the end -n 1000 chunk size -i 1000 max iterations -q be quiet You can specify an additional “where” clause: **purge Table where id=23**	Delete from (large) table in chunks Purge executes a series (-i) of delete statements, where each statement deletes (-n) rows at a time and commits.
PWD		Show current directory
QUIT		Leave SQLS*Plus.
RECOMPILE		Recompile objects
REFS		Display referential integrity dependencies
REM		
RERUN | !!	rerun <history_number>	Run history item number
SET AUTOCOMMIT	set auto[commit] <on|imm[ediate]|off>	Controls when SQL Server commits changes to the database after SQL commands or T-SQL call. ON commits changes to the database after SQL Server executes successful DML or T-SQL call. OFF turns off automatic committing so commit changes has to be done explicitly. IMMEDIATE is a synonym of ON.
SET AUTOFORMAT	set autoformat <table_name> maxsize <N> sample <N> maxsize default is 40 characters sample default is 10% of table size	Automatically generates optimal format definitions for table %char% and %int% columns based of sampling of table data Maxsize defines maximum column size for long character columns Sample defines what percent of the table data to scan to create optimal format definitions
SET COLSEP		Set column separator character

SET FEEDBACK	set feedback <on|off|N>	Display number of records returned by a query when a query selects at least n records **N** – when number of selected records is over N, number of records returned will be shown
SET HEADING		Set heading value
SET HEADSEP		Set heading separator
SET LINESIZE | LINES	set linesize <size>	Set Table of Contents **Type chapter title (level 1)1** Type chapter title (level 2) ....2 Type chapter title (level 3)..................................3 **Type chapter title (level 1)4** Type chapter title (level 2) ....5 Type chapter title (level 3)..................................6 output line size
SET NEWPAGE|NEWP	set newpage <0|n|none>	Sets number of blank lines to print from the page top to the top title If newpage is set to 0, form feed character is printed at the beginning of each page
SET MARKUP HTML | SET MARK HTML	SET MARK[UP] HTML [ON | OFF] [HEAD text] [BODY text] [TABLE text] [ENTMAP {ON | OFF}] [SPOOL {ON | OFF}] [PRE[FORMAT] {ON | OFF}]	Set output to HTML **HTML [ON|OFF]** HTML is a mandatory argument which specifies that HTML output is to be generated. HTML arguments, ON and OFF, specify whether or not to generate HTML output. The default is OFF. **HEAD text**

		The HEAD text option enables to specify content for the <HEAD> tag. By default, text includes a default in-line CSS and title. If text includes spaces, it must be enclosed in quotes. **BODY text** The BODY text option enables to specify attributes for the <BODY> tag. By default, there are no attributes. If text includes spaces, it must be enclosed in quotes. **TABLE text** The TABLE text option enables to enter attributes for the <TABLE> tag. By default, the <TABLE> WIDTH attribute is set to 90% and the BORDER attribute is set to 1. If text includes spaces, it must be enclosed in quotes. **ENTMAP {ON|OFF}** ENTMAP ON or OFF specifies whether or not SQL*Plus replaces special characters <, >, " and & with the HTML entities &lt;, &gt;, &quot; and &amp; respectively. ENTMAP is set ON by default. **SPOOL {ON|OFF}** SPOOL ON or OFF specifies whether or not SQLS*Plus writes the HTML opening tags, <HTML> and <BODY>, and the closing tags, </BODY> and </HTML>, to the start and end of each file created by

		the SQLS*Plus SPOOL filename command. The default is OFF. Header and footer tags enabled by the SET MARKUP HTML SPOOL ON option are not written to the spool file until "SPOOL filename" command is not issued **PRE[FORMAT] {ON|OFF}** PREFORMAT ON or OFF specifies whether or not SQLS*Plus writes output to the <PRE> tag or to an HTML table. The default is OFF, so output is written to a HTML table by default.
SET OUTPUT	set output <csv | json | default>	Set output to CSV (commas separated values), to JSON or to default output. Supported CSV and JSON format outputs all fields surrounded by double quotes
SET PAGESIZE | PAGES	set pagesize <size>	Set output page size
SET SQLPROMPT| SQLP	set sqlprompt <message> | reset	Sets the SQLS*Plus command prompt. SET SQLPROMPT can use define variables in the message
SET TERMOUT|TERM	set termout on|off	Controls the display of output generated by commands in a script that is executed with @, @@ or START. OFF stops output to screen to enable output to a file without displaying it on a screen. ON displays the output on screen.

		TERMOUT OFF does not affect output from commands entered interactively or directed to SQLS*Plus from the OS.
SET UNDERLINE	SET UND[ERLINE] {‘-‘ | c | ON | OFF}	Set the character to underline column headings. The underline character cannot be an alphanumeric or a white space. ON or OFF turns underlining on or off.
SET VERIFY		Print ampersand replacing
SET VOUT	set vout on|off	Set vertical output mode
SETVAR	SETVAR <variable> <value>	Set value of bind variable
SHO[W]	SHOW <set parameter name>	Show value of the named SET parameter
SHOW DB|DATABASE		Show current database
SHOW DBS|DATABASES		Show available databases
SHOW ERRORS		Show SQL Server error log
SHOW LICENSE		Show license information and license days to expiration
SHOW PARAMETER|PARM	show parameter <pattern>	Show database parameters
SHOW SERVERS		Show names of SQL Server instances located on a servers that broadcast on local domain network
SHOW TABLES|TAB		Show database tables
SHOW USER		Show the current username
SPOOL	SPO[OL] [file_name[.ext] [CRE[ATE] | REP[LACE] | APP[END]] | OFF |]	Write output to file APPEND – add output to the file CREATE – would not allow to overwrite existing file REPLACE (default) – replace existing file OFF – stop spooling
START | @		Execute sql script

STARTREL | @@		Execute sql script relative / nested to a running script
TTITLE	**Syntax** TTI[TLE] [*printspec* [*text* | *variable*] ...] [ON | OFF] where *printspec* represents one or more of the following clauses used to place and format the text: CE[NTER] LE[FT] R[IGHT] S[KIP] [*n*] COL [n] TAB [n] **Options:** These options also apply to the BTITLE command. *text* The ttitle text. Enter text in single quotes if you want to place more than one word on a single line. *variable* A substitution variable or any of the following system-maintained values, **SQL.LNO** (the current report line number), **SQL.PNO** (the current report page number) , **SQL.SYSDATE** (the current report timestamp), **SQL.USER** (the current connected user) To print one of these values, reference the appropriate variable in the title. You can format *variable* with the FORMAT clause. SQLS*Plus substitution variables (& variables) are expanded before TTITLE is executed. The resulting string is stored as the TTITLE text. **OFF** Turns the ttitle off (suppresses its display) without affecting its definition. **ON** Turns the title on (restores its display). When you define a top title, SQLS*Plus automatically sets TTITLE to ON. **S[KIP] [*n*], n>=1**	Places and formats a specified title at the top of each report page. Enter TTITLE with no clauses to list its current definition.

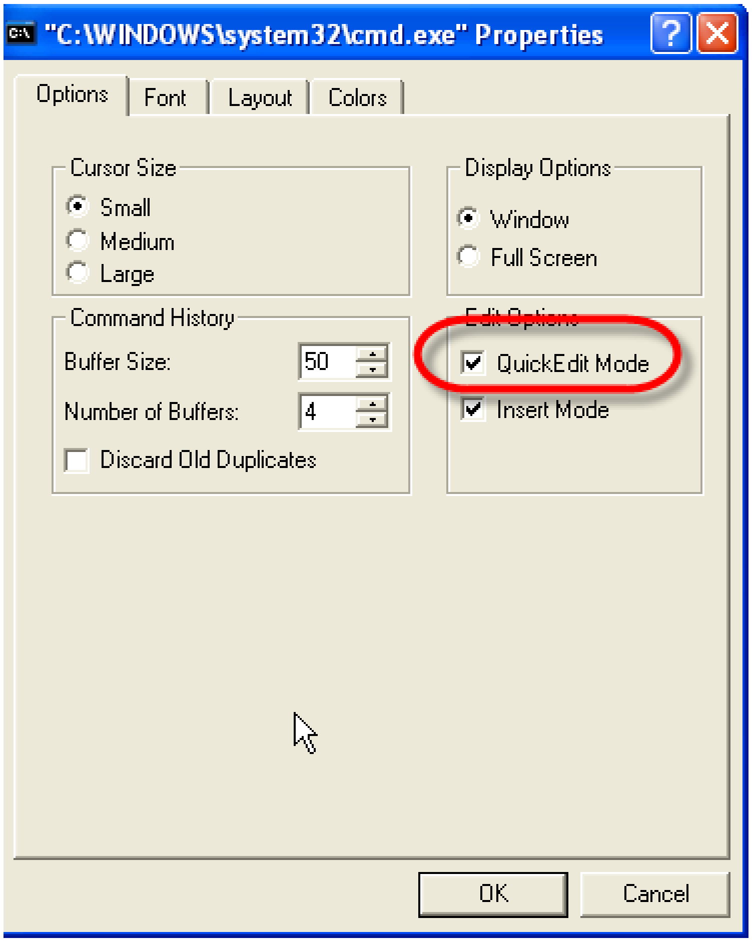
	Skips to the start of a new line *n* times; if you omit *n*, one time;
	COL [N]
	Moves to the line column n. N can be negative.
	TAB [n]
	Moves forward n columns (line columns, not database table columns) or backwards if n is a negative number
	LE[FT] | CE[NTER] | R[IGHT]
	Left-align, center, and right-align data on the current line respectively. CENTER and RIGHT use current LINESIZE value to calculate the relative position of the data items
	TTITLE with no clauses lists current TTITLE definition.
	Examples
	To define "Monthly Report" as the top title and to left-align it, to center the current date, to right-align the page number, and to display "Data in Millions" in the center of the next line, enter
	`TTITLE LEFT 'Monthly Report' CENTER SQL.SYSDATE`
	`RIGHT 'Page:' SQL.PNO SKIP CENTER 'Data in Millions'`	

TSQL		Display t-sql procedure code
VARIABLE	VARIABLE <name> <type> -s <value>	declare a bind variable

Chapter 5

Using Command Window as a suitable work environment

1) Activate Quick Edit Mode



"C:\WINDOWS\system32\cmd.exe" Properties

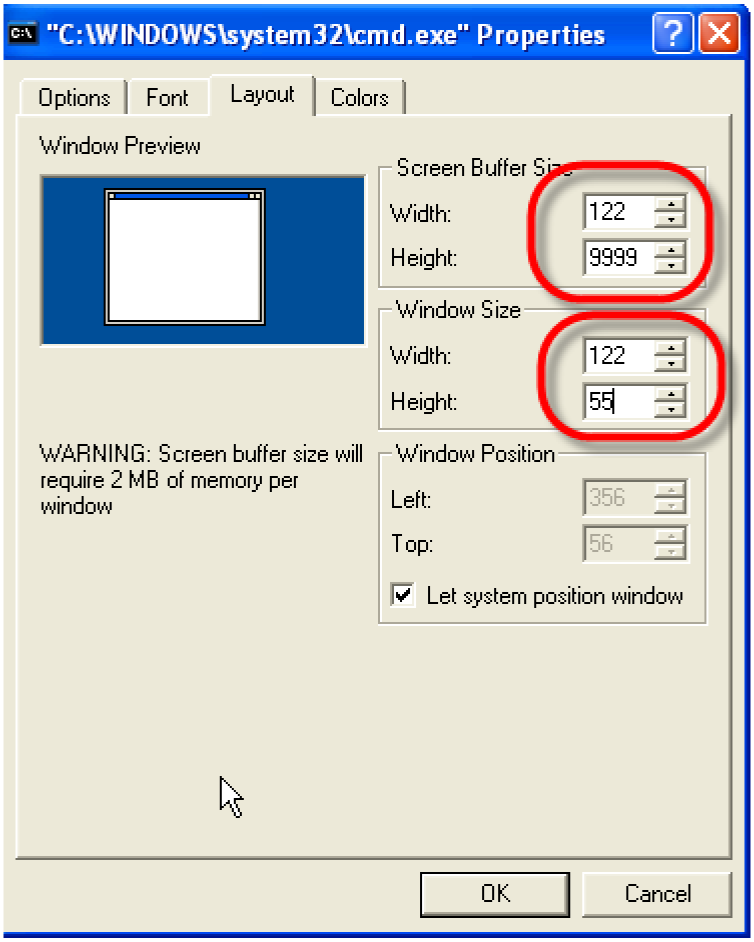
Cursor Size
● Small
○ Medium
○ Large

Display Options
● Window
○ Full Screen

Command History
Buffer Size: 50
Number of Buffers: 4
□ Discard Old Duplicates

Edit Options
☑ QuickEdit Mode
☑ Insert Mode

2) Set proper layout attributes



"C:\WINDOWS\system32\cmd.exe" Properties

Window Preview

Screen Buffer Size
Width: 122
Height: 9999

Window Size
Width: 122
Height: 55

Window Position
Left: 356
Top: 56

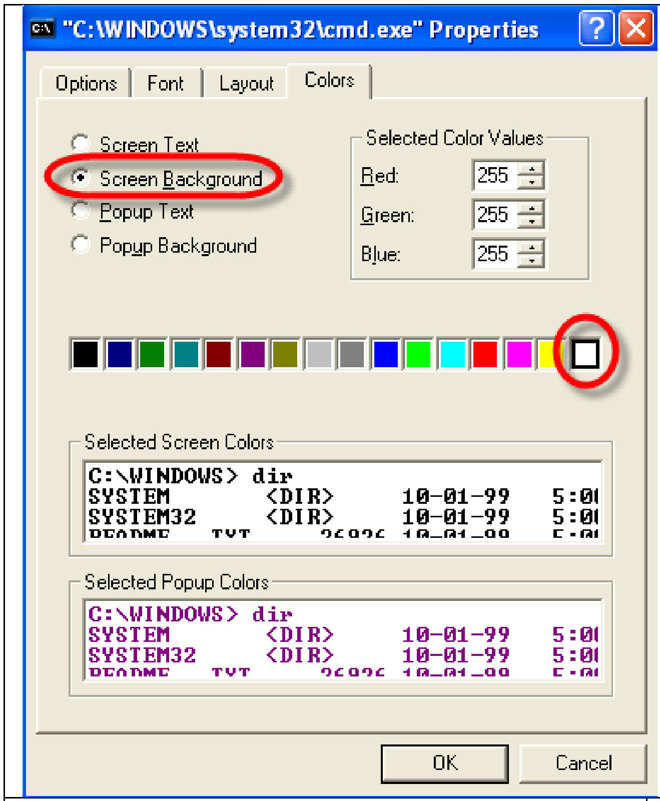
☑ Let system position window

WARNING: Screen buffer size will require 2 MB of memory per window

a) Set Screen Buffer Size to 122 and 9999 correspondingly

b) Set Window Buffer Size to 122 and 55 correspondingly

3) Set “easy to work with” Colors



"C:\WINDOWS\system32\cmd.exe" Properties

○ Screen Text
● Screen Background
○ Popup Text
○ Popup Background

Selected Color Values

Red: 255
Green: 255
Blue: 255

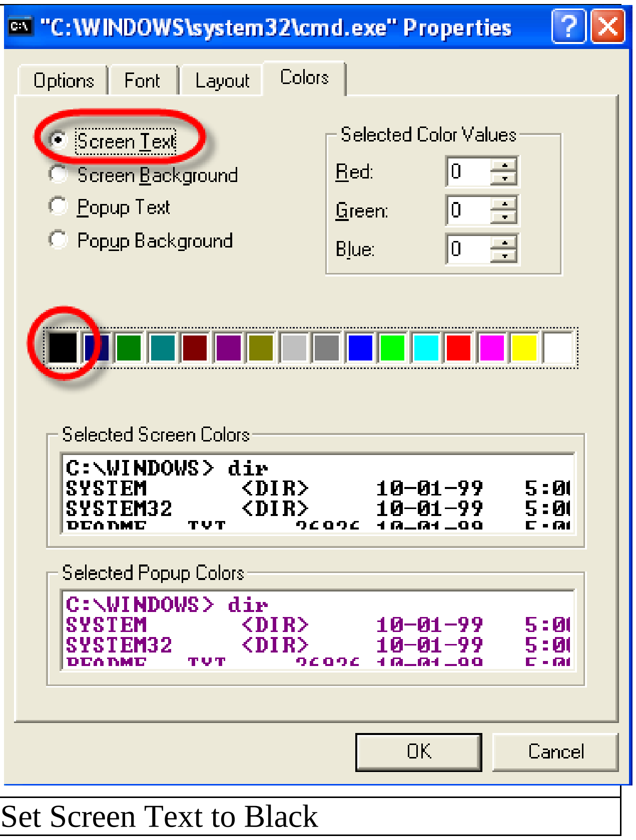
Selected Screen Colors

```
C:\WINDOWS> dir
SYSTEM        <DIR>      10-01-99     5:0
SYSTEM32      <DIR>      10-01-99     5:0
```

Selected Popup Colors

```
C:\WINDOWS> dir
SYSTEM        <DIR>      10-01-99     5:0
SYSTEM32      <DIR>      10-01-99     5:0
```

Set Screen Background to White



"C:\WINDOWS\system32\cmd.exe" Properties

● Screen Text
○ Screen Background
○ Popup Text
○ Popup Background

Selected Color Values

Red: 0
Green: 0
Blue: 0

Selected Screen Colors

```
C:\WINDOWS> dir
SYSTEM        <DIR>      10-01-99     5:0
SYSTEM32      <DIR>      10-01-99     5:0
```

Selected Popup Colors

```
C:\WINDOWS> dir
SYSTEM        <DIR>      10-01-99     5:0
SYSTEM32      <DIR>      10-01-99     5:0
```

Set Screen Text to Black