

Documentation for db2-hash-routines

Helmut K. C. Tessarek

14th June, 2015

Abstract

db2-hash-routines is a package which provides User Defined Functions and Stored Procedures for IBM[®] DB2[®] to generate and validate hashes.

<http://tessus.github.io/db2-hash-routines>

Date: 2015-06-14 00:51:26 -0400 Id: 2e393fd

Contents

1	db2-hash-routines	1
1.1	Building the library and registering the UDFs and SPs	1
1.2	Description of the UDFs and SPs	2
A	UDF and SP reference	3
A.1	bcrypt	3
A.2	php_md5	4
A.3	apr_md5	5
A.4	apr_crypt	6
A.5	apr_sha1	7
A.6	apr_sha256	8
A.7	validate_pw	9

1 db2-hash-routines

1.1 Building the library and registering the UDFs and SPs

Login as the instance user. Change the DB2PATH variable in the `makertn` script for your environment.

```
DB2PATH=/home/db2inst1/sqllib
```

Set DB2PATH to the directory where DB2 is accessed. This is usually the instance home directory.

After changing the above setting, start the script

Linux and AIX	<code>./makertn</code>
Win32	<code>makertn.bat</code>

The UDFs and SPs are written in ANSI C and should compile on all platforms. You can use the `bldrtn` script in your `sqllib/samples/c` directory as a good start. The only thing that you have to do is to install APR and APR-util. You can get APR and APR-util at <http://apr.apache.org/>. Furthermore you need to add the compiler and linker flags for APR (see `makertn`).

To register the UDFs and SPs, connect to your database and run the script:

```
db2 -tvf register.ddl
```

1.2 Description of the UDFs and SPs

This library delivers the following routines¹:

```
bcrypt  
php_md5  
apr_md5  
apr_crypt  
apr_sha1  
apr_sha256  
validate_pw
```

The `php_md5` routine is compatible to the PHP `md5` function.

The `apr_md5`, `apr_crypt`, `apr_sha1` and `bcrypt` routines are compatible to the functions used in Apache's `htpasswd` utility.

The `apr_sha256` routine returns the identifier `{SHA256}` plus the base64 encoded sha256 hash.

`validate_pw` can be used to validate a password against a hash.

Note: In win32 environments `apr_crypt` returns the output of `bcrypt`, if available. If `bcrypt` is not available, the output of `apr_md5` is returned.

¹see Appendix A for a reference of the UDFs and SPs

A UDF and SP reference

A.1 bcrypt

```
>>-BCRYPT--(--expression--)-----><
```

```
>>-BCRYPT--(--expression--,--hash--)-----><
```

bcrypt algorithm. The `bcrypt` routine is compatible to the function used in Apache's `htpasswd` utility.

The argument can be a character string that is either a `CHAR` or `VARCHAR` not exceeding 4096 bytes.

The result of the function is `CHAR(60)`. The result can be null; if the argument is null, the result is the null value.

Examples:

1)

```
INSERT INTO USERS (username, password)
VALUES ('test', bcrypt('testpwd'))
```

2)

```
SELECT bcrypt( 'testpwd' ) FROM SYSIBM.SYSDUMMY1
```

```
1
```

```
-----
$2y$05$2jb66aPElSkNLT1t8e6dQepuCY2BP3JnYUh0xeV9r1PEo0Gy0Lkym
```

```
1 record(s) selected.
```

3)

```
CALL bcrypt('testpwd', ?)
```

```
Value of output parameters
```

```
-----
```

```
Parameter Name   : HASH
```

```
Parameter Value  : $2y$05$WYSu1X6PVA0Ra.aPSjrdv.S6hOp.AYSnNRT521rmLRjD4Mj9UY6ve
```

```
Return Status = 0
```

A.2 php_md5

```
>>-PHP_MD5--(--expression--)-----><
```

```
>>-PHP_MD5--(--expression--,--hash--)-----><
```

MD5 hash. The `php_md5` routine is compatible to the PHP `md5` function.

The argument can be a character string that is either a `CHAR` or `VARCHAR` not exceeding 4096 bytes.

The result of the function is `CHAR(32)`. The result can be null; if the argument is null, the result is the null value.

Examples:

1)

```
INSERT INTO USERS (username, password)
VALUES ('test', php_md5('testpwd'))
```

2)

```
SELECT php_md5( 'testpwd' ) FROM SYSIBM.SYSDUMMY1
```

```
1
-----
342df5b036b2f28184536820af6d1caf
```

```
1 record(s) selected.
```

3)

```
CALL php_md5('testpwd', ?)
```

```
Value of output parameters
```

```
-----
```

```
Parameter Name  : HASH
```

```
Parameter Value : 342df5b036b2f28184536820af6d1caf
```

```
Return Status = 0
```

A.3 apr_md5

```
>>-APR_MD5--(--expression--)-----><
```

```
>>-APR_MD5--(--expression--,--hash--)-----><
```

Seeded MD5 hash. The `apr_md5` routine is compatible to the function used in Apache's `htpasswd` utility.

The argument can be a character string that is either a CHAR or VARCHAR not exceeding 4096 bytes.

The result of the function is CHAR(37). The result can be null; if the argument is null, the result is the null value.

Examples:

1)

```
INSERT INTO USERS (username, password)
VALUES ('test', apr_md5('testpwd'))
```

2)

```
SELECT apr_md5( 'testpwd' ) FROM SYSIBM.SYSDUMMY1
```

```
1
-----
$apr1$GfVm0TyJ$n7F1Vkw1/kX8MLgTJq1lp1
```

```
1 record(s) selected.
```

3)

```
CALL apr_md5('testpwd', ?)
```

```
Value of output parameters
```

```
-----
```

```
Parameter Name   : HASH
```

```
Parameter Value  : $apr1$GfVm0TyJ$n7F1Vkw1/kX8MLgTJq1lp1
```

```
Return Status = 0
```

A.4 apr_crypt

```
>>-APR_CRYPT--(--expression--)-----><
```

```
>>-APR_CRYPT--(--expression--,--hash--)-----><
```

Unix crypt. The `apr_crypt` routine is compatible to the function used in Apache's `htpasswd` utility.

The argument can be a character string that is either a CHAR or VARCHAR not exceeding 4096 bytes.

The result of the function is CHAR(13). The result can be null; if the argument is null, the result is the null value.

Examples:

1)

```
INSERT INTO USERS (username, password)
VALUES ('test', apr_crypt('testpwd'))
```

2)

```
SELECT apr_crypt( 'testpwd' ) FROM SYSIBM.SYSDUMMY1
```

```
1
-----
cqs7u0vz8KBlk
```

```
1 record(s) selected.
```

3)

```
CALL apr_crypt('testpwd', ?)
```

```
Value of output parameters
```

```
-----
```

```
Parameter Name  : HASH
```

```
Parameter Value : cqs7u0vz8KBlk
```

```
Return Status = 0
```


A.5 apr_sha1

```
>>-APR_SHA1--(--expression--)-----><
```

```
>>-APR_SHA1--(--expression--,--hash--)-----><
```

SHA1 algorithm. The `apr_sha1` routine is compatible to the function used in Apache's `htpasswd` utility.

The argument can be a character string that is either a CHAR or VARCHAR not exceeding 4096 bytes.

The result of the function is CHAR(33). The result can be null; if the argument is null, the result is the null value.

Examples:

1)

```
INSERT INTO USERS (username, password)
VALUES ('test', apr_sha1('testpwd'))
```

2)

```
SELECT apr_sha1( 'testpwd' ) FROM SYSIBM.SYSDUMMY1
```

```
1
-----
{SHA}m08HW0aqxvmp4Rl1SMgZC3LJWB0=

1 record(s) selected.
```

3)

```
CALL apr_sha1('testpwd', ?)
```

```
Value of output parameters
```

```
-----
```

```
Parameter Name   : HASH
```

```
Parameter Value  : {SHA}m08HW0aqxvmp4Rl1SMgZC3LJWB0=
```

```
Return Status = 0
```

A.6 apr_sha256

```
>>-APR_SHA256--(--expression--)-----><
```

```
>>-APR_SHA256--(--expression--,--hash--)-----><
```

SHA256 algorithm. The `apr_sha256` routine returns the identifier {SHA256} plus the base64 encoded sha256 hash.

The argument can be a character string that is either a CHAR or VARCHAR not exceeding 4096 bytes.

The result of the function is CHAR(52). The result can be null; if the argument is null, the result is the null value.

Examples:

1)

```
INSERT INTO USERS (username, password)
VALUES ('test', apr_sha256('testpwd'))
```

2)

```
SELECT apr_sha256( 'testpwd' ) FROM SYSIBM.SYSDUMMY1
```

```
1
```

```
-----
{SHA256}qFtqIIE8Maixs/NhjaeWJxyaopOz+AmHMF0yGuxQEIc=
```

```
1 record(s) selected.
```

3)

```
CALL apr_sha256('testpwd', ?)
```

```
Value of output parameters
```

```
-----
```

```
Parameter Name : HASH
```

```
Parameter Value : {SHA256}qFtqIIE8Maixs/NhjaeWJxyaopOz+AmHMF0yGuxQEIc=
```

```
Return Status = 0
```

A.7 validate_pw

```
>>-VALIDATE_PW--(--password--,--hash--)-----><
```

```
>>-VALIDATE_PW--(--password--,--hash--,--is_valid--)-----><
```

This routine can be used to validate a password against a hash.

The two input arguments can be character strings that are either a CHAR or VARCHAR not exceeding 4096 bytes (password) and 120 bytes (hash). The second parameter (hash) must not be empty, otherwise an `SQLSTATE 39701` is returned.

The result of the routine is an INTEGER. If the password is valid, 1 is returned. If the password is not valid, 0 is returned. The result can be null; if the argument is null, the result is the null value.

Examples:

1)

```
SELECT validate_pw('testpwd', 'cqs7u0vz8KBlk') FROM SYSIBM.SYSDUMMY1"
```

```
1
-----
1
```

1 record(s) selected.

2)

```
CALL validate_pw('testpwd', 'cqs7u0vz8KBlk', ?)
```

Value of output parameters

```
-----
Parameter Name  : IS_VALID
Parameter Value : 1
```

Return Status = 0

3)

```
CALL validate_pw('testpwd', '0123456789abcdef', ?)
```

Value of output parameters

Parameter Name : IS_VALID
Parameter Value : 0

Return Status = 0

Date: 2015-06-14 00:51:26 -0400 Id: 2e393fd125ee209525212a7f89c407dd600c0e3c