

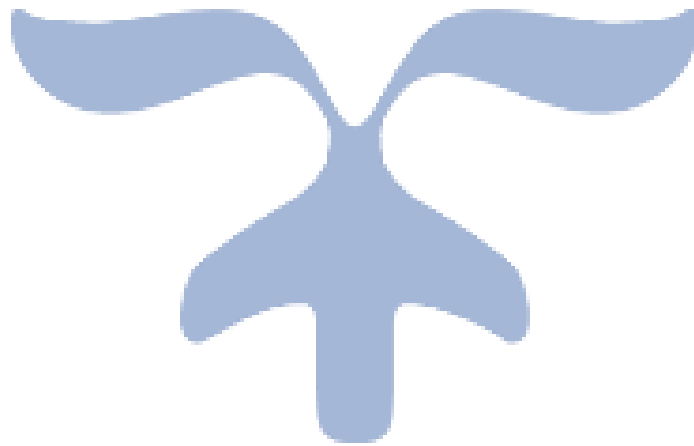


NETWORK SECURITY

Assignment 3

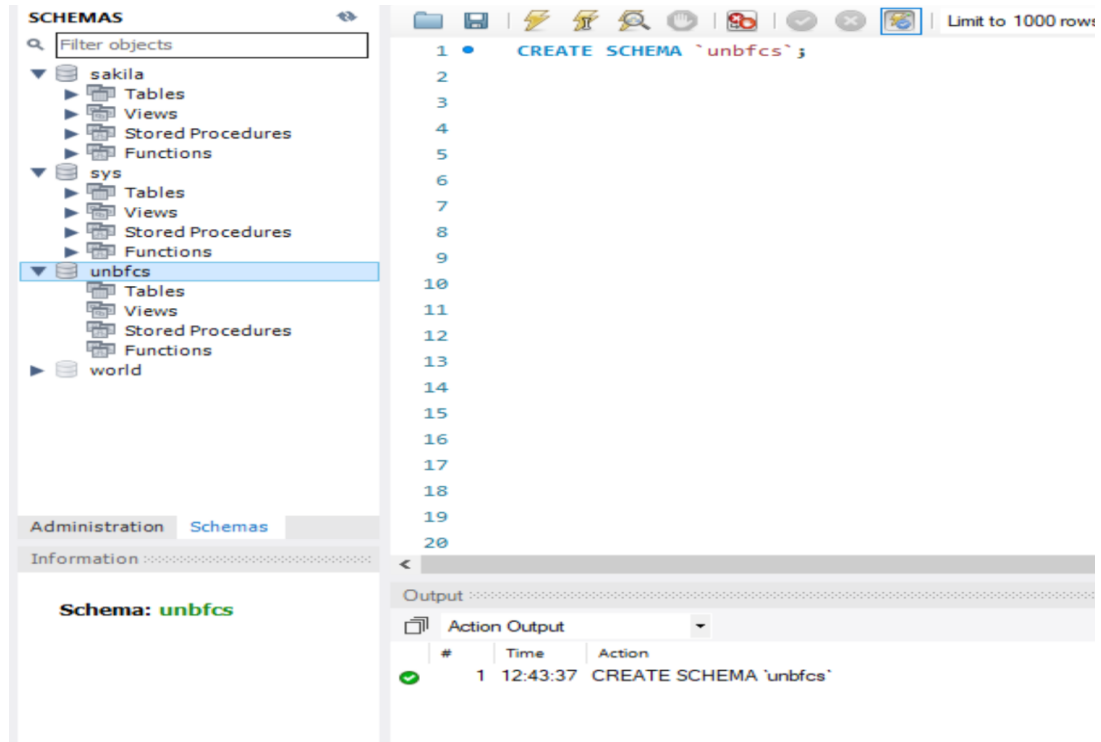
Soheil Shirvani

3720505

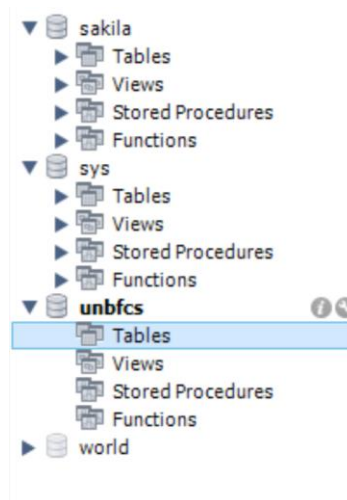


A. Data Integrity

1. Create Schema for UNBFCS



2. Expand Database and Select Tables



3. Create Table CS4415 with Specified Fields and Insert Values:

The screenshot displays a database management interface. On the left, a 'SCHEMAS' pane shows a tree view with 'sakila', 'sys', 'unbfc', and 'world' schemas. Under 'unbfc', there is a 'Tables' folder containing 'cs4415'. The main editor shows the following SQL script:

```
1 CREATE TABLE `unbfc`.`cs4415` (  
2   `StuID` INT NOT NULL,  
3   `StuFName` VARCHAR(45) NULL,  
4   `StuLName` VARCHAR(45) NULL,  
5   `StuGrade` FLOAT NULL,  
6   PRIMARY KEY (`StuID`));  
7  
8 INSERT INTO `unbfc`.`cs4415`  
9   (`StuID`, `StuFName`, `StuLName`, `StuGrade`) VALUES (1000, 'Bob', 'Bobby', 9.6);  
10 INSERT INTO `unbfc`.`cs4415`  
11   (`StuID`, `StuFName`, `StuLName`, `StuGrade`) VALUES (1011, 'John', 'Johnny',  
12     18.82);  
13 INSERT INTO `unbfc`.`cs4415`  
14   (`StuID`, `StuFName`, `StuLName`, `StuGrade`) VALUES (1023, 'Rose', 'Rosey',  
15     18.82);
```

Below the editor, the 'Output' pane shows the execution results:

#	Time	Action	Message
1	09:16:11	Apply changes to unbfc	Changes applied
2	09:16:57	CREATE TABLE `unbfc`.`cs4415` (`StuID` INT NOT NULL, `StuFName` VARCHAR(45) NULL, `StuLName` V...	0 row(s) affected
3	09:17:15	INSERT INTO `unbfc`.`cs4415` (`StuID`, `StuFName`, `StuLName`, `StuGrade`) VALUES (1000, 'Bob', 'Bobby', 9.6)	1 row(s) affected
4	09:17:15	INSERT INTO `unbfc`.`cs4415` (`StuID`, `StuFName`, `StuLName`, `StuGrade`) VALUES (1011, 'John', 'Johnny', 18.82)	1 row(s) affected
5	09:17:15	INSERT INTO `unbfc`.`cs4415` (`StuID`, `StuFName`, `StuLName`, `StuGrade`) VALUES (1023, 'Rose', 'Rosey', 18.82)	1 row(s) affected

Below the output, a new SQL query is entered:

```
17 SELECT * FROM unbfc.cs4415
```

The 'Result Grid' pane shows the data returned by the query:

StuID	StuFName	StuLName	StuGrade
1000	Bob	Bobby	9.6
1011	John	Johnny	18.82
1023	Rose	Rosey	18.82
NULL	NULL	NULL	NULL

4. Create CS4415Hash Table with Addition column and Read and Insert Hash value to the new column:

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'unbfcscs' schema with a table named 'cs4415Hash'. The right pane shows the SQL script for creating the table and inserting data.

```

21 CREATE TABLE 'unbfcscs'.cs4415Hash (
22     'StuID' INT NOT NULL,
23     'StuFName' VARCHAR(45) NULL,
24     'StuLName' VARCHAR(45) NULL,
25     'StuGrade' FLOAT NULL,
26     'StuHashValue' VARCHAR(32) NULL,
27     PRIMARY KEY ('StuID'));
28
29 INSERT INTO unbfcscs.cs4415Hash (StuID,StuFName,StuLName,StuGrade, StuHashValue)
30 SELECT StuID,StuFName,StuLName,StuGrade, MD5(concat(StuID,StuFName,StuLName,StuGrade)) FROM unbfcscs.cs4415;
31
32 SELECT * FROM unbfcscs.cs4415Hash;
33

```

The 'Result Grid' shows the following data:

StuID	StuFName	StuLName	StuGrade	StuHashValue
1000	Bob	Bobby	9.6	85f2aab9e692aac0fdd5237fc5813e5
1011	John	Johnny	18.82	4271288e82c10d6427e2d0de876dc99e
1023	Rose	Rosey	18.82	19cad736ef6fec8e833e066287bd99d1

The 'Action Output' pane shows the execution results:

#	Time	Action	Message
4	14:06:48	INSERT INTO unbfcscs.cs4415 ('StuID','StuFName','StuLName','StuGrade') VALUES (1011,'John','Johnny',...	1 row(s) affected
5	14:06:48	INSERT INTO unbfcscs.cs4415 ('StuID','StuFName','StuLName','StuGrade') VALUES (1023,'Rose','Rosey',...	1 row(s) affected
6	14:06:48	SELECT * FROM unbfcscs.cs4415 LIMIT 0, 1000	3 row(s) returned
7	14:06:54	CREATE TABLE unbfcscs.cs4415Hash ('StuID' INT NOT NULL, 'StuFName' VARCHAR(45) NULL, 'StuLName'...	0 row(s) affected
8	14:07:02	INSERT INTO unbfcscs.cs4415Hash (StuID,StuFName,StuLName,StuGrade, StuHashValue) SELECT StuID,StuFName,StuLName,StuGrade, MD5(concat(StuID,StuFName,StuLName,StuGrade)) FROM unbfcscs.cs4415;	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
9	14:07:02	SELECT * FROM unbfcscs.cs4415Hash LIMIT 0, 1000	3 row(s) returned

Question 1: Insert (Soheil Shirvani 3720505 20) into New Table:

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'unbfcscs' schema with a table named 'cs4415Hash'. The right pane shows the SQL script for inserting data into the table.

```

29 INSERT INTO unbfcscs.cs4415Hash (StuID,StuFName,StuLName,StuGrade, StuHashValue)
30 SELECT StuID,StuFName,StuLName,StuGrade, MD5(concat(StuID,StuFName,StuLName,StuGrade)) FROM unbfcscs.cs4415;
31
32 SELECT * FROM unbfcscs.cs4415Hash;
33
34 SELECT MD5(concat(StuID,StuFName,StuLName,StuGrade)) FROM unbfcscs.cs4415 WHERE StuID=1023; # '19cad736ef6fec8e833e066287bd99d1'
35 SELECT MD5(concat(StuID,StuFName,StuLName,20)) FROM unbfcscs.cs4415 WHERE StuID=1023; # 'f1f514e47eeb358db00465b3bca1856a'
36
37 INSERT INTO unbfcscs.cs4415Hash (StuID,StuFName,StuLName,StuGrade, StuHashValue)
38 VALUES (3720505,'Soheil','Shirvani',20,MD5(concat(3720505,'Soheil','Shirvani',20))) ;
39
40 SELECT * FROM unbfcscs.cs4415Hash;
41

```

The 'Result Grid' shows the following data:

StuID	StuFName	StuLName	StuGrade	StuHashValue
1000	Bob	Bobby	9.6	85f2aab9e692aac0fdd5237fc5813e5
1011	John	Johnny	18.82	4271288e82c10d6427e2d0de876dc99e
1023	Rose	Rosey	18.82	19cad736ef6fec8e833e066287bd99d1
3720505	Soheil	Shirvani	20	d4482d2427630492141432af14fe93d6

The 'Action Output' pane shows the execution results:

#	Time	Action	Message
11	14:19:31	SELECT * FROM unbfcscs.cs4415Hash LIMIT 0, 1000	3 row(s) returned
12	14:19:45	INSERT INTO unbfcscs.cs4415Hash (StuID,StuFName,StuLName,StuGrade, StuHashValue) VALUES (3720505,...	Error Code: 1054. Unknown column 'Soheil' in field list
13	14:20:02	SELECT * FROM unbfcscs.cs4415Hash LIMIT 0, 1000	3 row(s) returned
14	14:21:32	INSERT INTO unbfcscs.cs4415Hash (StuID,StuFName,StuLName,StuGrade, StuHashValue) VALUES (3720505,...	Error Code: 1054. Unknown column 'Soheil' in field list
15	14:22:06	INSERT INTO unbfcscs.cs4415Hash (StuID,StuFName,StuLName,StuGrade, StuHashValue) VALUES (3720505,...	1 row(s) affected
16	14:22:11	SELECT * FROM unbfcscs.cs4415Hash LIMIT 0, 1000	4 row(s) returned

Question 2: Modify Grade in new Inserted Value:

The screenshot displays the SQL Server Enterprise Manager interface. The SQL Commands window shows the following query:

```

29 INSERT INTO unbfcs.cs4415Hashed (StuID,StuFName,StuLName,StuGrade, StuHashValue)
30 SELECT StuID,StuFName,StuLName,StuGrade, MD5(concat(StuID,StuFName,StuLName,StuGrade)) FROM unbfcs.cs4415;
31
32 SELECT * FROM unbfcs.cs4415Hashed;
33
34 SELECT MD5(concat(StuID,StuFName,StuLName,StuGrade)) FROM unbfcs.cs4415 WHERE StuID=1023; # '19cad736ef6fec8e833e066287bd99d1'
35 SELECT MD5(concat(StuID,StuFName,StuLName,20)) FROM unbfcs.cs4415 WHERE StuID=1023; # 'f1f514e47eeb358db00465b3bca1856a'
36
37 INSERT INTO unbfcs.cs4415Hashed (StuID,StuFName,StuLName,StuGrade, StuHashValue)
38 VALUES (3720505,'Soheil','Shirvani',20,MD5(concat(3720505,'Soheil','Shirvani',20))) ;
39
40 SELECT * FROM unbfcs.cs4415Hashed;
41

```

The Results Grid shows the following data:

StuID	StuFName	StuLName	StuGrade	StuHashValue
1000	Bob	Bobby	9.6	85f2aab9e692aac0fdd5237ffc5813e5
1011	John	Johnny	18.82	4271288e82c10d6427e2d0de876dc99e
1023	Rose	Rosey	18.82	19cad736ef6fec8e833e066287bd99d1
3720505	Soheil	Shirvani	20	d4482d2427630492141432af14fe93d6

The Action Output window shows the following messages:

#	Time	Action	Message
11	14:19:31	SELECT * FROM unbfcs.cs4415Hashed LIMIT 0, 1000	3 row(s) returned
12	14:19:45	INSERT INTO unbfcs.cs4415Hashed (StuID,StuFName,StuLName,StuGrade, StuHashValue) VALUES (3720505,...	Error Code: 1054. Unknown column 'Soheil' in field list
13	14:20:02	SELECT * FROM unbfcs.cs4415Hashed LIMIT 0, 1000	3 row(s) returned
14	14:21:32	INSERT INTO unbfcs.cs4415Hashed (StuID,StuFName,StuLName,StuGrade, StuHashValue) VALUES (3720505,...	Error Code: 1054. Unknown column 'Soheil' in field list
15	14:22:06	INSERT INTO unbfcs.cs4415Hashed (StuID,StuFName,StuLName,StuGrade, StuHashValue) VALUES (3720505,...	1 row(s) affected
16	14:22:11	SELECT * FROM unbfcs.cs4415Hashed LIMIT 0, 1000	4 row(s) returned

Question 3:

In this type of attack, an intruder can change each of the records and validate his changes just by updating the hash value using new values. To protect the table against such attacks, 3 important defense practices can be performed:

- 1) We can save the hash values on a separate table and ensure integrity by checking that table. This way the one who modifies a value can't change the hash value.
- 2) We should check if the modify value is legal and then update the hash value. This means we can't simply update the hash field of the table
- 3) We should consider access control. Accessing to database must be protected and given to specific people who we have acknowledged. This way if something changed in the database, digital forensics could reveal the thief identity though the logs.

B. Data Confidentiality

1. Create Encrypted Grade Table:

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Schemas' tree with 'unbfcs' selected. The right pane shows the 'SQL Commands' window with the following SQL script:

```

43
44 SELECT * FROM unbfcs.cs4415Hashed;
45
46 CREATE TABLE 'unbfcs'.cs4415enc (
47     'StuID' blob NOT NULL,
48     'StuFName' blob,
49     'StuLName' blob,
50     'StuGrade' blob
51 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
52
53 SELECT * FROM 'unbfcs'.cs4415enc;
54
55 INSERT INTO unbfcs.cs4415enc (StuID, StuFName, StuLName, StuGrade)

```

The 'Result Grid' at the bottom shows the output of the SQL commands:

Time	Action	Message
14:27:23	SELECT StuHashValue FROM unbfcs.cs4415Hashed WHERE StuID=3720505 LIMIT 0, 1000	1 row(s) returned
14:28:13	UPDATE unbfcs.cs4415Hashed SET StuGrade=10, StuHashValue=MD5(concat(StuID,StuFName,StuLName,10)) ...	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
14:28:13	SELECT * FROM unbfcs.cs4415Hashed LIMIT 0, 1000	4 row(s) returned
14:34:53	CREATE TABLE 'unbfcs'.cs4415enc ('StuID' blob NOT NULL, 'StuFName' blob, 'StuLName' blob, 'StuGrade' ...	0 row(s) affected
14:34:53	SELECT * FROM 'unbfcs'.cs4415enc LIMIT 0, 1000	0 row(s) returned

2. Read and Encrypt the Previous values into the new one:

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Schemas' tree with 'unbfcs' selected. The right pane shows the 'SQL Commands' window with the following SQL script:

```

51 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
52
53 SELECT * FROM 'unbfcs'.cs4415enc;
54
55 INSERT INTO unbfcs.cs4415enc (StuID, StuFName, StuLName, StuGrade)
56 SELECT aes_encrypt(StuID,'qazWSX123!@#'),
57        aes_encrypt(StuFName,'qazWSX123!@#'),
58        aes_encrypt(StuLName,'qazWSX123!@#'),
59        aes_encrypt(StuGrade,'qazWSX123!@#') FROM unbfcs.cs4415;
60
61 SELECT * FROM 'unbfcs'.cs4415enc;
62
63

```

The 'Result Grid' at the bottom shows the output of the SQL commands:

Time	Action	Message
14:28:13	UPDATE unbfcs.cs4415Hashed SET StuGrade=10, StuHashValue=MD5(concat(StuID,StuFName,StuLName,10)) ...	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
14:28:13	SELECT * FROM unbfcs.cs4415Hashed LIMIT 0, 1000	4 row(s) returned
14:34:53	CREATE TABLE 'unbfcs'.cs4415enc ('StuID' blob NOT NULL, 'StuFName' blob, 'StuLName' blob, 'StuGrade' ...	0 row(s) affected
14:34:53	SELECT * FROM 'unbfcs'.cs4415enc LIMIT 0, 1000	0 row(s) returned
14:36:22	INSERT INTO unbfcs.cs4415enc (StuID, StuFName, StuLName, StuGrade) SELECT aes_encrypt(StuID,'qazWSX1...	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
14:36:22	SELECT * FROM 'unbfcs'.cs4415enc LIMIT 0, 1000	3 row(s) returned

Question 4: Insert My Encrypted Value (3720505,Soheil,Shirvani,20)

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the database schema, including the 'unbfc' database and the 'cs4415enc' table. The central pane shows the SQL commands being executed:

```

56 SELECT aes_encrypt(StuID, 'qazWSX123!@#'),
57 aes_encrypt(StuFName, 'qazWSX123!@#'),
58 aes_encrypt(StuLName, 'qazWSX123!@#'),
59 aes_encrypt(StuGrade, 'qazWSX123!@#') FROM unbfc.cs4415;
60
61 SELECT * FROM `unbfc`.`cs4415enc`;
62
63 INSERT INTO unbfc.cs4415enc (StuID, StuFName, StuLName, StuGrade)
64 VALUES (aes_encrypt(3720505, 'qazWSX123!@#'), aes_encrypt('Soheil', 'qazWSX123!@#'),
65 aes_encrypt('Shirvani', 'qazWSX123!@#'), aes_encrypt(20, 'qazWSX123!@#'));
66
67 SELECT * FROM `unbfc`.`cs4415enc`;
68

```

The bottom pane shows the 'Result Grid' with the following data:

StuID	StuFName	StuLName	StuGrade
01.00	01.00	01.00	01.00
01.00	01.00	01.00	01.00
01.00	01.00	01.00	01.00
01.00	01.00	01.00	01.00

The 'Output' pane shows the execution results of the commands:

#	Time	Action	Message
65	14:54:39	CREATE TABLE `unbfc`.`cs4415enc` (`StuID` blob NOT NULL, `StuFName` blob, `StuLName` blob, `StuGrade` ...	0 row(s) affected
66	14:54:40	SELECT * FROM `unbfc`.`cs4415enc` LIMIT 0, 1000	0 row(s) returned
67	14:54:47	INSERT INTO unbfc.cs4415enc (StuID, StuFName, StuLName, StuGrade) SELECT aes_encrypt(StuID, 'qazWSX123!@#'),	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
68	14:54:47	SELECT * FROM `unbfc`.`cs4415enc` LIMIT 0, 1000	3 row(s) returned
69	14:54:54	INSERT INTO unbfc.cs4415enc (StuID, StuFName, StuLName, StuGrade) VALUES (aes_encrypt(3720505, 'qazWSX123!@#'),	1 row(s) affected
70	14:54:54	SELECT * FROM `unbfc`.`cs4415enc` LIMIT 0, 1000	4 row(s) returned

Question 5: Update my grade with encrypted value for 18.82

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the database schema, including the 'unbfc' database and the 'cs4415enc' table. The central pane shows the SQL commands being executed:

```

71 UPDATE unbfc.cs4415enc SET StuGrade = aes_encrypt(18.82, 'qazWSX123!@#') WHERE StuID = aes_encrypt('3720505', 'qazWSX123!@#');
72
73
74 SELECT * FROM `unbfc`.`cs4415enc`;
75
76 SELECT CAST(aes_decrypt(StuID, 'qazWSX123!@#') as CHAR(200)),
77 CAST(aes_decrypt(StuFName, 'qazWSX123!@#') as CHAR(200)),
78 CAST(aes_decrypt(StuLName, 'qazWSX123!@#') as CHAR(200)),

```

The bottom pane shows the 'Result Grid' with the following data:

StuID	StuFName	StuLName	StuGrade
BLOB	BLOB	BLOB	BLOB
BLOB	BLOB	BLOB	BLOB
BLOB	BLOB	BLOB	BLOB
BLOB	BLOB	BLOB	BLOB

The 'Output' pane shows the execution results of the commands:

#	Time	Action	Message
---	------	--------	---------

Here we can't see the result. Question 7 Shows the result after casting values to Characters.

Question 6:

The problem is if someone gets the key he/she can access the values and modify them by encrypting the value using the same key. There are 2 defense strategies we can do.

- 1) We should make sure we protect the key in a safe place outside this database. This makes sure that the thief can simply access and gets the key.
- 2) One thing is we can use AES CBC mode using a random vector and concatenating it with the encrypted data. So here we are enforcing the AES algorithm making sure an attacker can't break the algorithm.
- 3) Another problem is when an attacker access encrypted values he/she can find similar encrypted values and get to know which users have same fields. This way attacker can compromise the privacy of the users. One solution to that is we can have different keys for each row. For this we can have a base key and add some pre-defined salt to it for each row like using users ID with the encryption key. With this solution we can make sure that we don't have similar values and if an attacker get to break the algorithm he/she has only identified one row and can't modify other rows.
- 4) Another solution is to define access control so if a DB administrator changed a value we can go through the logs can reveal the identity of the attacker

Question 7: Showing the plain text of the table

```
76 • SELECT CAST(aes_decrypt(StuID,'qazWSX123!@#') as CHAR(200)),
77 CAST(aes_decrypt(StuFName,'qazWSX123!@#') as CHAR(200)),
78 CAST(aes_decrypt(StuLName,'qazWSX123!@#') as CHAR(200)),
79 CAST(aes_decrypt(StuGrade,'qazWSX123!@#') as CHAR(200))
80 FROM unbfcs.cs4415enc;
```

Result Grid

CAST(aes_decrypt(StuID,'qazWSX123!@#') as CHAR(200))	CAST(aes_decrypt(StuFName,'qazWSX123!@#') as CHAR(200))	CAST(aes_decrypt(StuLName,'qazWSX123!@#') as CHAR(200))	CAST(aes_decrypt(StuGrade,'qazWSX123!@#') as CHAR(200))
1000	Bob	Bobby	9.6
1011	John	Johnny	18.82
1023	Rose	Rosey	18.82
3720505	Soheil	Shirvani	18.82

Result 3 x

Output

Action Output

#	Time	Action	Message
1	12:02:06	SELECT CAST(aes_decrypt(StuID,'qazWSX123!@#') as CHAR(200)), CAST(aes_decrypt(StuFName,'qazWSX123!@#') as CHAR(200)), CAST(aes_decrypt(StuLName,'qazWSX123!@#') as CHAR(200)), CAST(aes_decrypt(StuGrade,'qazWSX123!@#') as CHAR(200)) FROM unbfcs.cs4415enc;	4 row(s) returned

So we can see the update as well as all the insertions worked well.