DATABASE & SYSTEM SECURITY

Lab-10

Q1. You are a database security consultant. You want to store some confidential information in your database. Do this using AES encryption.

First, create a html page with the username and secret message fields. Once the inputs are given and submit button is pressed, the javascript program in the background will encrypt the message using AES and store it in the database along with the username.

Create a table called 'dbusers2' to store the output.

AES Java Code – (use this to create a package)

```
-----------------------------------------------
import javax.crypto.Cipher; import
javax.crypto.spec.SecretKeySpec;

public class AdvancedEncryption
{
    private byte[] key;

    private static final String ALGORITHM = "AES";

    public AdvancedEncryption(byte[] key)
    {
this.key = key;
    }
    /**
* Encrypts the given plain text
     *
* @param plainTextThe plain text to encrypt
     */
    public byte[] encrypt(byte[] plainText) throws Exception
    {
SecretKeySpecsecretKey = new SecretKeySpec(key, ALGORITHM);
Cipher cipher = Cipher.getInstance(ALGORITHM);
cipher.init(Cipher.ENCRYPT_MODE, secretKey);

        return cipher.doFinal(plainText);
    }

    /**
* Decrypts the given byte array
     *
* @param cipherTextThe data to decrypt
     */
    public byte[] decrypt(byte[] cipherText) throws Exception
    {
SecretKeySpecsecretKey = new SecretKeySpec(key, ALGORITHM);
Cipher cipher = Cipher.getInstance(ALGORITHM);
cipher.init(Cipher.DECRYPT_MODE, secretKey);

        return cipher.doFinal(cipherText);
    }
}
------------------------------------------------
```

Usage example:

```
byte[] encryptionKey = "MyKeyValue16char".getBytes(StandardCharsets.UTF_8);
byte[] plainText = "(MySecretMessage)".getBytes(StandardCharsets.UTF_8);
AdvancedEncryptionStandardadvancedEncryptionStandard =
newAdvancedEncryptionStandard(encryptionKey);
byte[] cipherText = advancedEncryptionStandard.encrypt(plainText); byte[]
decryptedCipherText = advancedEncryptionStandard.decrypt(cipherText);
```

- -

-- Sol)

1. Connect as system to sqlplus and create the ‘dbusers’ table. create table dbusers2(username varchar2(20), password varchar2(20), epwd blob);
2. Switch on the Eclipse IDE.
   Go to ‘File’ ∟ New ∟ Other ∟ Scroll down to ‘Web’ ∟ Expand the folder and choose ‘Dynamic Web Project’.
3. Set project name as something of your choice.
4. Choose ‘Dynamic web module version’ as 2.5.
5. Click Finish.
6. Then expand the project file in ‘Project Explorer’. Expand ‘WebContent’ ∟ WEB-INF ∟ Right click on ‘lib’ ∟ Build Path… ∟ Configure Build Path… ∟ Click on ‘Classpath’ ∟ Click ‘Add External JARs…’ ∟ Navigate to the ‘ojdbc14.jar’ and add it.
7. Create a package called ‘p1’ with the java AES code.
   Expand your project in Project Explorer ∟ Java Resources ∟ Right click on ‘src’ ∟ New ∟ Package ∟ Type Name as ‘p1’.
   Right click on ‘p1’ ∟ New ∟ Class ∟ Name it AdvancedEncryption∟ Press Finish.
   You can use the AES functions in your jsp file by utilising the package.
8. Then right click on ‘WebContent’. This is where you make your html and jsp files.
   Right click on ‘WebContent’ ∟ New ∟ HTML File.
   Give a filename and end it with ‘.html’ extension. Press Finish.
   secretdataenc.html

```
<html>
<head>
<metacharset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<formaction="aesencryption.jsp"method="post">

Username<inputtype="text"name="uname">

<br>
My secret message <inputtype="text"name="secret">

<br>
<inputtype="submit">



</form>
</body>
</html>
```

Save the file.

9. Right click on ‘WebContent’ ⎿ New ⎿ JSP File.

   Give a filename and end it with ‘.jsp’ extension. Press Finish.

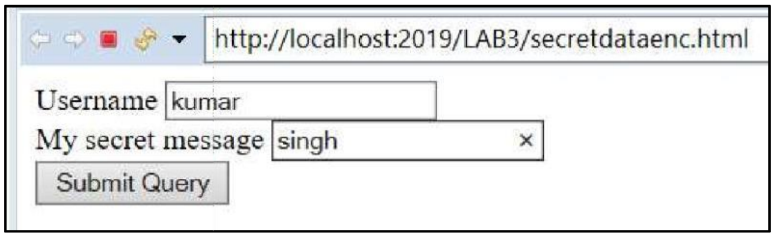
   aesencryption.jsp

```
<%@ pagelanguage="java"contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>

<%@pageimport="java.sql.*"%>
<%@pageimport="p1.*"%>
<%@pageimport="java.lang.Object"%>
<%@pageimport="java.nio.charset.StandardCharsets"%>



<!DOCTYPEhtmlPUBLIC"-//W3C//DTD HTML 4.01
Transitional//EN""http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<metahttp-equiv="Content-Type"content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
```

```
<h1>  Welcome   </h1>

<%


String u=request.getParameter("uname");

String p=request.getParameter("secret");

byte[] encryptionKey = "MyKeyValue16char".getBytes(StandardCharsets.UTF_8); //give 16 chars as 16x8=128
bits

byte[] plainText = p.getBytes(StandardCharsets.UTF_8);

AdvancedEncryptionaes = newAdvancedEncryption(encryptionKey); byte[] cipherText =
aes.encrypt(plainText);



out.println(" Input:\n"+u +"---"+p); try {

                        Class.forName("oracle.jdbc.driver.OracleDriver");


                Connection
con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","system","sy  stem");

            PreparedStatementps=con.prepareStatement("insert into dbusers2 values(?,?,?)");
            ps.setString(1,u);  ps.setString(2,p);
ps.setBytes(3,cipherText);
System.out.println(cipherText);

int a=ps.executeUpdate();

out.println(a+ "  Record Inserted Successfully");
}
catch(Exception e)
{
                out.println(e);

}

%>
```

Save the file.

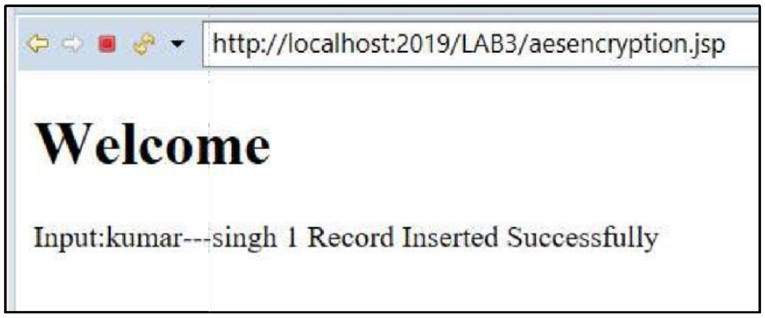
10. Right click on the html file and press ‘Run As’ ∟ 1 Run on Server ∟ (The Tomcat server should already be selected) ∟ Press Finish.



Username kumar
My secret message singh

11. Give your username and message.

    The inputs I am giving are ‘kumar’ as username and ‘singh’ as message.

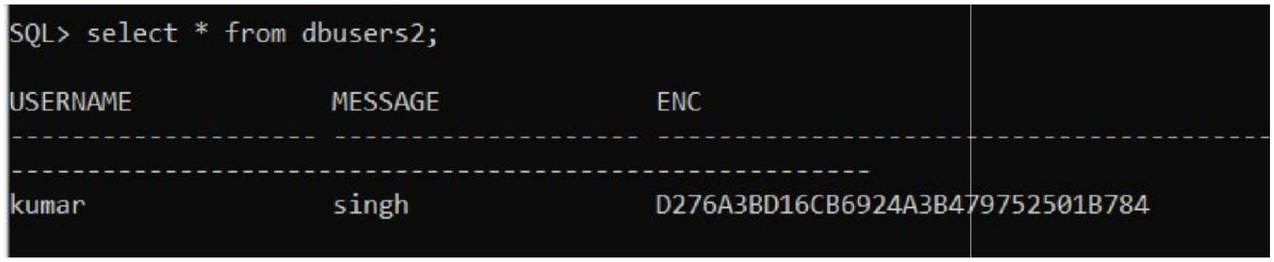


Welcome

Input:kumar---singh 1 Record Inserted Successfully

12. The hashed password must be stored in the table ‘dbusers2’ now.

    Display the table using ‘select * from dbusers2;’.



```
SQL> select * from dbusers2;

USERNAME             MESSAGE              ENC
-------------------- -------------------- ----------------------------------------------------------------------------
kumar                singh                D276A3BD16CB6924A3B479752501B784
```

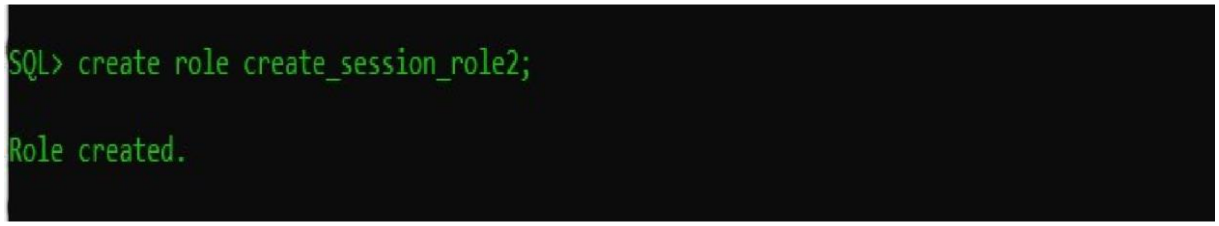
<u>Post-Lab:</u>

Q1.Create an Application Security User

Sol)

Step1:create a role

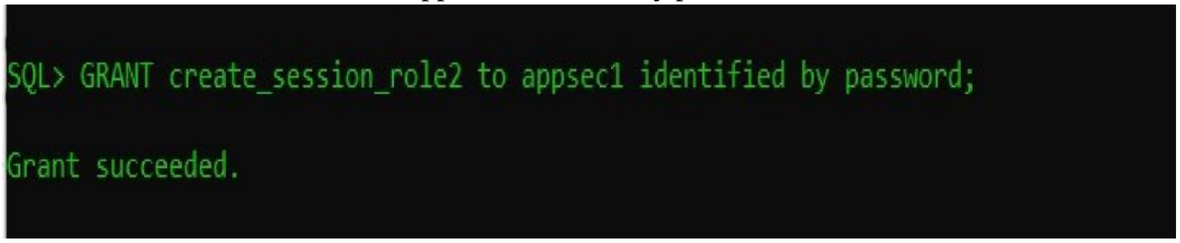
Create role create_session_role2;



```
SQL> create role create_session_role2;

Role created.
```

Step2: Grant the role to appsec1

GRANT create_session_role2 to appsec1 identified by password;



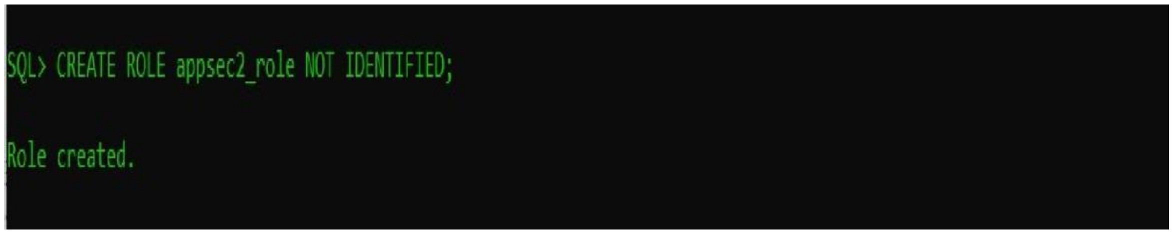
```
SQL> GRANT create_session_role2 to appsec1 identified by password;

Grant succeeded.
```

Q2. Our application security user needs to create procedures, functions, Java stored procedures, tables, and views. When creating those items, appsec requires the CREATE PROCEDURE, CREATE TABLE, and CREATE VIEW system privileges. We will grant those privileges to a role named appsec_role, and grant that role to the appsec user.

Sol)

Step1:create a role

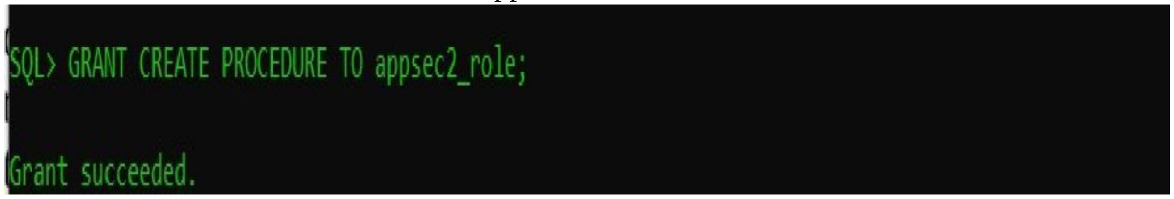
CREATE ROLE appsec2_role NOT IDENTIFIED;



```
SQL> CREATE ROLE appsec2_role NOT IDENTIFIED;

Role created.
```

Step2: Grant create procedure to the role

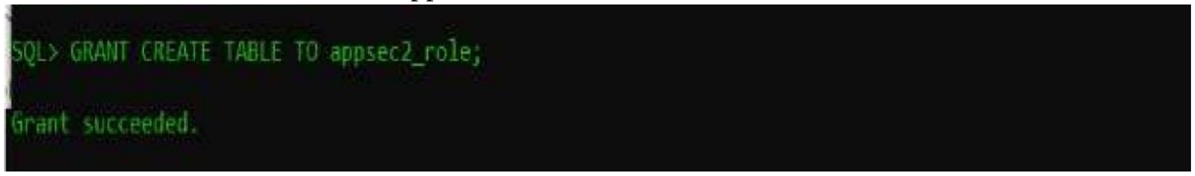
GRANT CREATE PROCEDURE TO appsec2_role;



```
SQL> GRANT CREATE PROCEDURE TO appsec2_role;

Grant succeeded.
```

Step3:Grant create table to the role

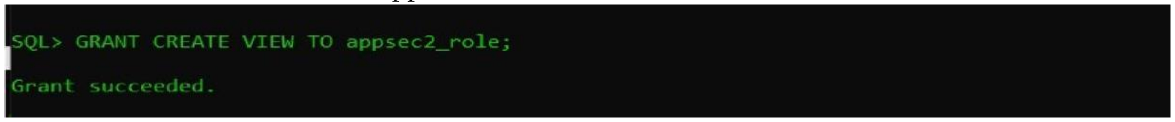
GRANT CREATE TABLE TO appsec2_role;



```
SQL> GRANT CREATE TABLE TO appsec2_role;

Grant succeeded.
```

Step4:Grant create view to the role

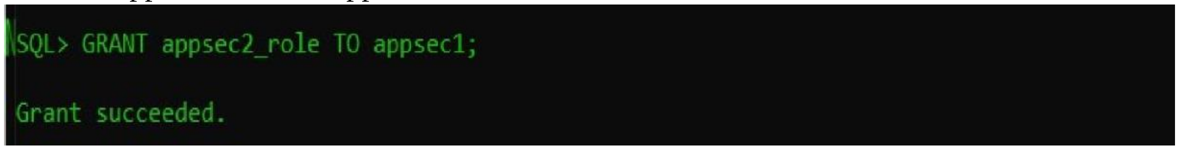
GRANT CREATE VIEW TO appsec2_role;



```
SQL> GRANT CREATE VIEW TO appsec2_role;

Grant succeeded.
```

Step5:Grant role to the user

GRANT appsec2_role TO appsec1;



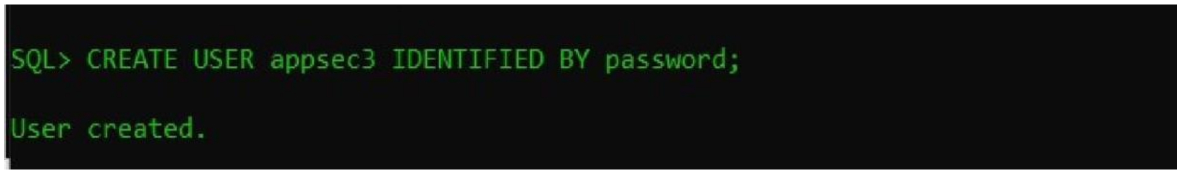
```
SQL> GRANT appsec2_role TO appsec1;

Grant succeeded.
```

Q3. Write a query to specify how much space appsec may use, a quota. We'll start out permitting two megabytes of space

Sol)

Step1:First we need to create a user

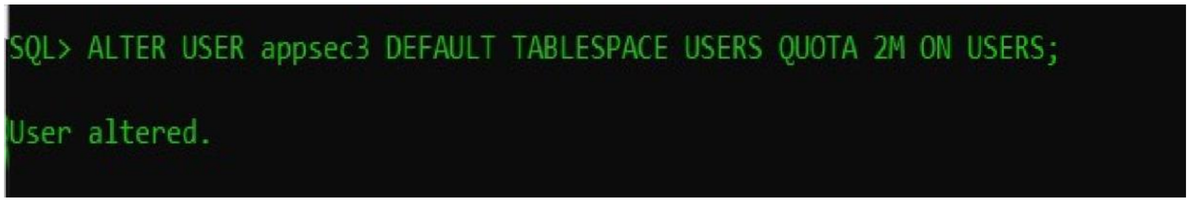
CREATE USER appsec3 IDENTIFIED BY password;



```
SQL> CREATE USER appsec3 IDENTIFIED BY password;

User created.
```

Step2:Alter the user with tablespace of 2 megabytes

ALTER USER appsec3 DEFAULT TABLESPACE USERS QUOTA 2M ON USERS;



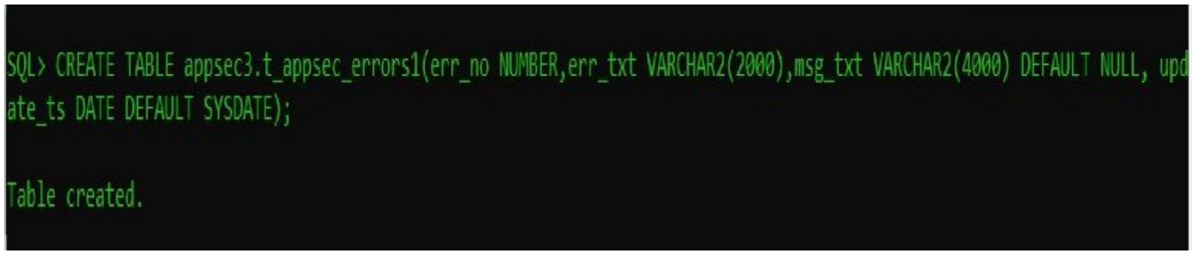
```
SQL> ALTER USER appsec3 DEFAULT TABLESPACE USERS QUOTA 2M ON USERS;

User altered.
```

Q4.Write a query to Create a table in the appsec schema for logging errors and create a trigger associated with that table.

Sol)

Step1:Create a table

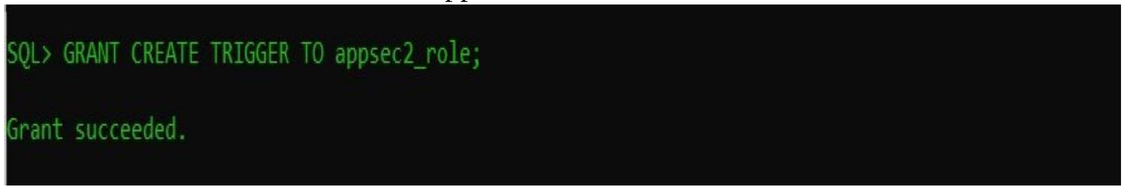
CREATE TABLE appsec.t_appsec_errors (err_no NUMBER, err_txt VARCHAR2(2000), msg_txt VARCHAR2(4000) DEFAULT NULL, update_ts DATE DEFAULT SYSDATE);



```
SQL> CREATE TABLE appsec3.t_appsec_errors1(err_no NUMBER,err_txt VARCHAR2(2000),msg_txt VARCHAR2(4000) DEFAULT NULL, upd
ate_ts DATE DEFAULT SYSDATE);

Table created.
```

Step2:Grant the trigger to the role

GRANT CREATE TRIGGER TO appsec2_role;



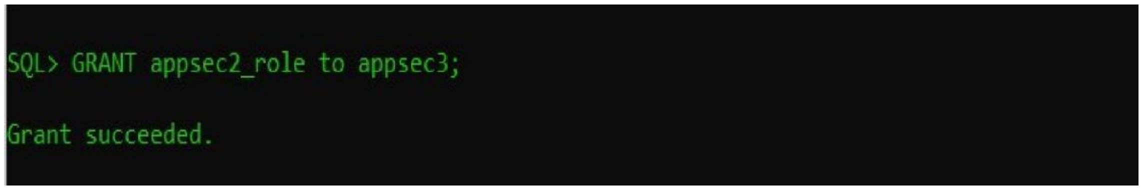
```
SQL> GRANT CREATE TRIGGER TO appsec2_role;

Grant succeeded.
```

Q5. Create a trigger, to grant the privilege of non-default role

Sol)

Step1:Grant the role to user

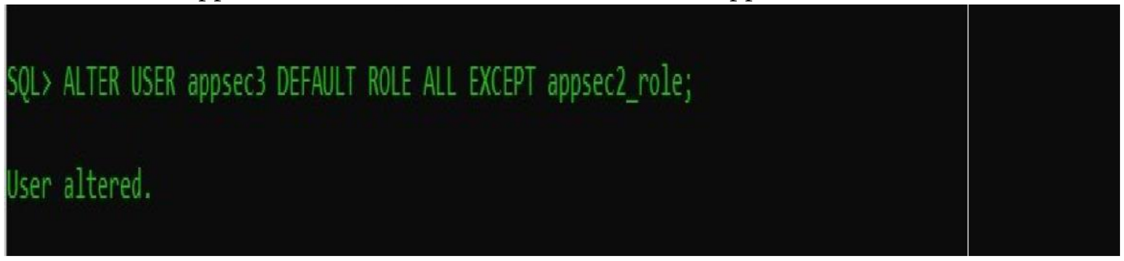
GRANT appsec2_role to appsec3;



```
SQL> GRANT appsec2_role to appsec3;

Grant succeeded.
```

Step2:Alter the user

AlTER USER appsec3 DEFAULT ROLE ALL EXCEPT appsec2_role;



```
SQL> ALTER USER appsec3 DEFAULT ROLE ALL EXCEPT appsec2_role;

User altered.
```