Assignment 3 – Security and Data Governance

Submission

Your submission should include three items:

- 5 .sql files as outlined later in this assignment.
- 5 .txt files as outlined later in this assignment
- DO NOT submit a .zip file. 10 separate files is what is required.
- MAKE SURE in all files it is clear which question – and which part of which question – is being answered.

Reminders

This will form the basis of future assignments. We will continue to grow and enhance this database. Make sure you do a complete job of these tasks to help ensure a smooth process for future assignments.

Tasks

1 – (20%) – Using your music database assume there is a user with userid **dba625a**. Give this user access to the music database.

- Give this user **read** access to the **entire database**
- Give this user **write** access only to the **bandrecognition table**
- Give this user the ability **create views**
- This user should **not** have the ability to create any other objects

What to hand in.

- Connect to the database using this new userid.
- Demonstrate that the user can read data from the database

- Demonstrate that the user can update data in the bandrecognition table
- Demonstrate that the user cannot update data in other tables
- Demonstrate that the user can create a view
- Demonstrate that the user cannot create an index

2 – (20%) – Create an audit policy which tracks the following:

- Any update to the **bandrecognition** table. This would include an INSERT, UPDATE and DELETE command.
- Any change to **genre** column of the **bandinfo** table

What to hand in.

- Connect to the database using the new userid **dba625a**
- Have the user run an INSERT command against the bandrecognition table
- Have the user run an UPDATE command against the bandrecognition table
- Have the user run a DELETE command against the bandrecognition table
- Have the user run an UPDATE command against the bandinfo table which updates the genre column
- Show the created records in the audit table which reflects the above audited activities

3 – (20%) – Create a trigger which performs the exact same operation and as part of the trigger firing updates a bandaudit table.

What to hand in.

- Create a bandaudit table with the following columns
    - Tblnm (which will contain the table name)
    - Tblops (which contains an “I” or “U” or “D”)
    - Tbldttm (which contains the current date/time stamp)
- Turn off the audit policy from Step 2
- Re-run the same operations as in Step 2
- Provide the output of a SELECT * FROM BANDAUDIT

4 – (20%) – Update the bandinfo table so that it is encrypted.

NOTE: Depending on the database system you are using you may need to completely recreate the database. If you have to do this, do the following:

- EXPORT all the tables
- DROP the database
- CREATE the database WITH encryption turned on
- IMPORT all the tables

What to hand in:

- Show the commands executed to accomplish the above
- Show the results of a SELECT * FROM BANDINFO without the decryption key
- Show the results of a SELECT * FROM BANDINFO with the decryption key

5 – (20%) – This will cover roles and row/column filtering

- Re-create the userid called **dba625b**
- Create a new userid called **dba625c**

- Create a ROLE called BANDCAN and add dba625b to this role.
- Create a ROLE called BANDANA and add dba625c to this role
- BANDCAN role should have read access to bandinfo
- BANDCAN role should only be allowed to see information from bandino table if the BASECOUNTRY column is ‘Canada’
- BANDANA role should have read access to bandinfo table
- BANDANA role should only see the first letter of GENRE – the rest of GENRE should be masked with an ‘*’ (for example: ROCK would show up as R***)

What to hand in:

- All of the security commands required to create the two roles and provide the appropriate authorizations
- All of the security commands required to add the respective userids to their respective roles
- All of the security commands required to define the row filters and column filters for the respective roles
- The output from you (database owner) running a SELECT * FROM BANDINFO
- The output from dba625b running a SELECT * FROM BANDINFO
- The output from dba625c running a SELECT * FROM BANDINFO

What to hand in:

Testing –

The specific instructions for each question are provided in the above steps. Follow those instructions. The bottom line is that you must demonstrate that each of the steps above have functioned correctly. If you feel you need to run some additional commands to ensure the fully working functionality is demonstrated, please feel free to do so and document what you are doing clearly.

Filenames –

For each question above you would hand in two files:

(i) An **A3Qn.sql** file which shows all the commands you are running

(ii) An **A3Qn.txt** output file which shows the results of all your tests and command executions

In the end, you should hand in 5 .sql files called A3Q1.sql, A3Q2.sql, …. A3Q5.sql – and – 5 .txt (output) files called A3Q1.txt, A3Q2.txt, …. A3Q5.txt.

Marks will be deducted if you do not follow these naming conventions. It needs to be clear what question – and what part of what question – I am marking.

Make sure you include comments in your .sql files to make it clear what you are running.

Make sure you include comments in your output file (you can add these manually) to ensure it is clear what output is being addressed.

Marking Scheme –

There are 5 questions worth 20% each.

Q1 – New userid and GRANT commands – 20%

Q2 – Audit controls – 20%

Q3 – Trigger for auditing – 20%

Q4 – Encryption – 20%

Q5 – Roles and Row/Column Filters – 20%

To receive the marks you must have the appropriate code in the .sql file and tests in the .txt file showing the code works. Marks will be deducted if you fail to follow programming best practices (such as good comments) in your .sql file.