Lab 06 – Transactions and Security

# **Objectives**

The purpose of this lab is to introduce the student to both transactions and security. In the real-world, databases tasks often involve multiple steps and if any step in the middle fails, the procedure is a failure. This lap walks the student through a couple transactions and lets them learn how various steps have varying consequences that they need to be aware of.

By the end of this lab, the student will be able to:

* Describe the steps of a transaction, how a transaction begins and ends and walk through live scenarios of a variety of transactions
* Understand and act appropriately on what needs to be done in the case of transaction failure
* Grant and revoke permissions to and from other users and public users from the database

**Submission**

***Your submission will be a single SQL file with the solutions provided. (with a .sql file extension)***

DBS211\_L06\_LastName.sql

Your submission needs to include a comment header block and be commented top clearly indicate the answers to each question. Make sure every SQL statement terminates with a semicolon.

Example Submission

|  |
| --- |
| -- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  -- Name: Your Name  -- ID: #########  -- Date: The current date  -- Purpose: Lab 06 DBS211  -- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  -- Q1 SOLUTION --  SELECT \* FROM TABLE;  -- Q2 Solution –  SELECT \* FROM TABLE; |

Locate, select, and submit the file to the Lab 04 link.

## **Setup**

Create a new worksheet in SQL developer and add an appropriate comment header that includes your name, student id, the date and the purpose of the file (i.e. DBS211 – Lab 04).

## **Style Guide**

Your SQL should be written using the standard coding style:

* all keywords are to be upper case,
* all user-defined names are to be lower case, (example: table and field names)
* there should be a carriage return before each major part of the SQL statements (i.e. before SELECT, FROM, WHERE and ORDER BY)

See the following sample:

SELECT columns

FROM tables

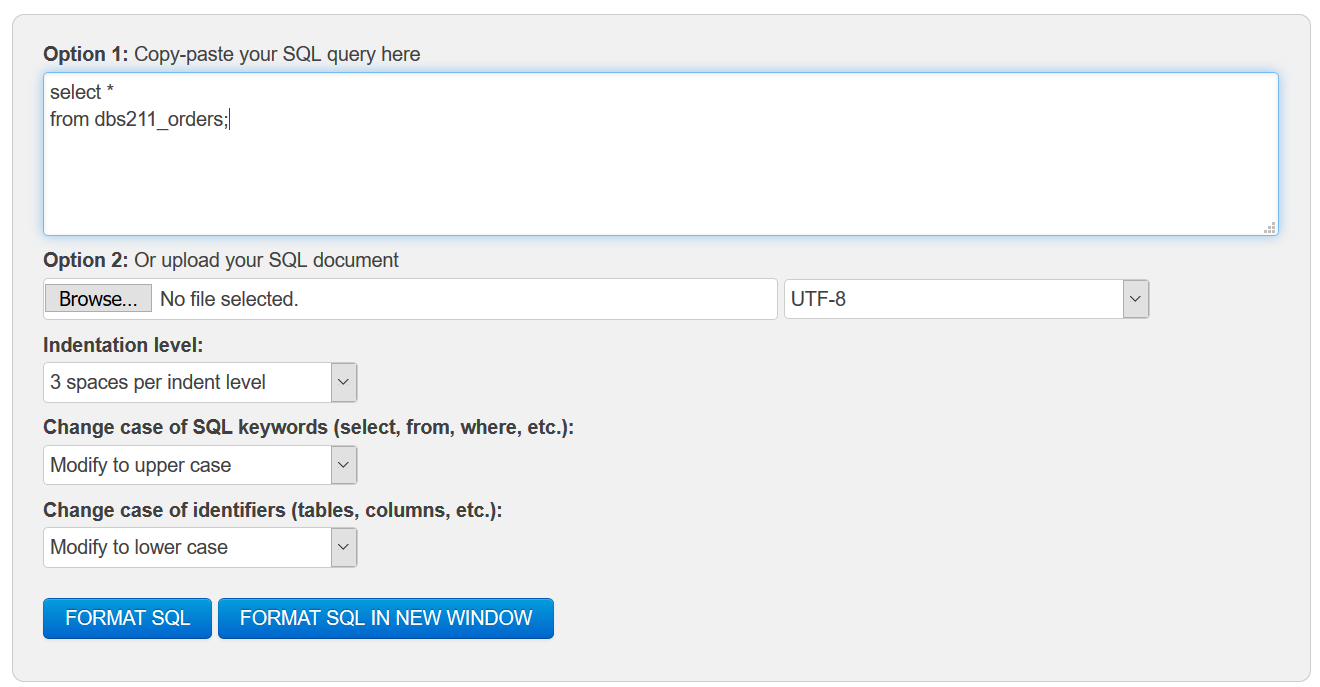
WHERE conditions

ORDER BY column1, column2;

To save time, you can write the SQL statements in your SQL developer. To make sure that your SQL statements style follows the standard SQL style guideline, copy and paste your SQL statements onto the following website and click on “FORMAT SQL” or “FORMAT SQL IN NEW WINDOW”.

<https://www.freeformatter.com/sql-formatter.html#ad-output>

You can also upload your SQL file. See the setting in the following image. Have SQL keywords (SELECT, INSERT, UPDATE, etc) uppercase and user defined objects and identifiers (tables, columns, etc.) lowercase.



**Marking Scheme**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Points** | **Question** | **Points** |
| **1** | 1 | **6** | 1 |
| **2** | 1 | **7** | 1 |
| **3** | 1 | **8** | 1 |
| **4** | 1 | **9** | 1 |
| **5** | 1 | **10** | 1 |

Total: 10

You will use following data to complete the given tasks:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **employeeNumber** | **lastname** | **firstname** | **extension** | **email** | **OfficeCode** | **reportsTo** | **jobTitle** |
| 1001 | Patel | Ralph | 22333 | rpatel@mail.com | 1 | NULL | Sales Rep |
| 1002 | Denis | Betty | 33444 | [bdenis@mail.com](mailto:bdenis@mail.com) | 4 | NULL | Sales Rep |
| 1003 | Biri | Ben | 44555 | [bbirir@mail.com](mailto:bbirir@mail.com) | 2 | NULL | Sales Rep |
| 1004 | Newman | Chad | 66777 | [cnewman@mail.com](mailto:cnewman@mail.com) | 3 | NULL | Sales Rep |
| 1005 | Ropeburn | Audrey | 77888 | aropebur@mail.com | 1 | NULL | Sales Rep |

* ***SET TRANSACTION READ WRITE*** starts a new transaction.
* ***COMMIT*** commits the current transaction, making its changes permanent.
* ***SAVEPOINT <name>***  sets a pointer to a location that can be rolled back to.
* ***ROLLBACK*** rolls back the current transaction, canceling its changes.
* ***SET autocommit*** disables or enables the default ***autocommit*** mode for the current session.

**Tasks:**

1. List the 4 ways that we know that a transaction can be started.

1. Using SQL, create an **empty** table, that is the same as the dbs211\_employees table, and name it *staffs*.

Execute the following commands.

SET AUTCOMMIT OFF;  
SET TRANSACTION READ WRITE;

1. Write an INSERT statement to populate the *staffs* table with the rows of the sample data. Insert the NULL value for the reportsTo column. (Write a single INSERT statement to insert all the rows)
2. Create a query that shows all the inserted rows from the *staffs* table. (show all the columns) How many rows are selected?
3. Execute the rollback command. Display all rows and columns from the *staffs* table. How many rows are selected?
4. Repeat Task 4. Make the insertion permanent to the table *staffs*. Display all rows and columns from the ***staffs*** table. How many rows are selected?
5. Do the following tasks:
   1. Write an update statement to update the value of column *repotsto* to 1004 for all the employees except the employee with ID 1004 in the *staffs* table.
   2. Write an update statement to update the jobTitle of the employee 1004 to ‘Sales Manager’ and make your changes permanent.
6. Create a query that shows all employees. (show all columns)
7. Execute the rollback command.
   1. Display all employees from the *staff* table whose manager is 1004. How many rows are still updated?
   2. Was the rollback command effective?
   3. What was the difference between the result of the rollback execution from Task 6 and the result of the rollback execution of this task?
8. Write statements to permanently remove the *staffs* table created for this lab.

Good luck.