# DATABASE ADMINISTRATION FINAL PROJECT

## Exercise 1.1 - Set up the lab environment

- Start the PostgreSQL Server
- Download the lab setup bash file from https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0231EN-SkillsNetwork/labs/Final%20Assignment/postgres-setup.sh
- Run the bash file

TERMINAL:

- wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0231EN-SkillsNetwork/labs/Final%20Assignment/postgres-setup.sh
- $ sudo chmod +x postgres-setup.sh
- ls -l postgres-setup.sh
- ./postgres-setup.sh

## Exercise 1.2 - User Management

### Task 1.2 - Create a User

- Create a user named backup_operator.

POSTGRE CLI:

- CREATE USER backup_operator WITH PASSWORD 'backup_operator_password';

### Task 1.3 - Create a Role

- Create a role named backup.

POSTGRE CLI:

- CREATE ROLE backup;

### Task 1.4 - Grant privileges to the role

- Grant the following privileges to the backup role.
    - CONNECT ON tolldata DATABASE.
    - SELECT ON ALL TABLES IN SCHEMA toll.

POSTGRE CLI:

- GRANT CONNECT ON DATABASE tolldata TO backup;
- \connect tolldata;
- GRANT SELECT ON ALL TABLES IN SCHEMA toll TO backup;

### Task 1.5 - Grant role to an user

- Grant the role backup to backup_operator

POSTGRE CLI:

- GRANT backup TO backup_operator;

# Exercise 2.1 - Set up the lab environment

- Before you proceed with the assignment, start the MySQL Server.

# Exercise 2.2 – Recovery

## Task 2.1 - Restore MySQL server using a previous backup

- Download the backup file https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0231EN-SkillsNetwork/labs/Final%20Assignment/billingdata.sql.
- Restore this file onto MySQL server.
- List the tables in the billing database.

TERMINAL:

- wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0231EN-SkillsNetwork/labs/Final%20Assignment/billingdata.sql

MySQL CLI:

- CREATE DATABASE billing;
- USE billing;
- SOURCE billingdata.sql;
- SHOW TABLES;

## Task 2.2 - Find the table data size

- Find the data size of the table billdata.

MySQL CLI:

- SELECT table_name, (data_length + index_length)/1024 AS DataSize_kB FROM INFORMATION_SCHEMA.TABLES WHERE table_name = 'billdata';

# Exercise 2.3 – Indexing

## Task 2.3 - Baseline query performance

- Write a query to select all rows with a billedamount > 19999 in table billdata.

MYSQL CLI:

- SELECT * FROM billdata WHERE billedamount>19999;
- EXPLAIN SELECT * FROM billdata WHERE billedamount>19999;

## Task 2.4 - Create an index

- Your customer wants to improve the execution time of the query you wrote in Task 2.3.
- Create an appropriate index to make it run faster.

MYSQL CLI:

- CREATE INDEX billedamount_index ON billdata(billedamount);

### Task 2.5 - Document the improvement in query performance

- Find out if the index has any impact on query performance.
- Re-run the baseline query of Task 2.3 after creating the index.

MYSQLCLI:

- SELECT * FROM billdata WHERE billedamount>19999;
- EXPLAIN SELECT * FROM billdata WHERE billedamount>19999;

## Exercise 2.4 - Storage Engines

### Task 2.6 - Find supported storage engines

- Run a command to find out if your MySQL server supports the MyISAM storage engine.

MYSQL CLI:

- SHOW ENGINES;

### Task 2.7 - Find the storage engine of a table

- Find the storage engine of the table billdata.

MYSQL CLI:

- SELECT table_name, engine FROM INFORMATION_SCHEMA.TABLES WHERE table_name = 'billdata';

## Exercise 2.5 - OPTIONAL Exercise (Non-graded) Automation of routine tasks

### Bonus Task 2.8 - Write a bash script that performs a backup of all the databases

- mysqldump is a command line tool that performs logical backups of a database.
- Its generic syntax is mysqldump db_name > backup-file.sql
- Its extended syntax is mysqldump --all-databases --user=root --password=NzA4Ny1y > backup-file.sql
- Write a bash script named mybackup.shthat performs the following tasks.
    - Perform the backup of all databases using the mysqldump
    - Store the output in the file all-databases-backup.sql
    - In the /tmp directory, create a directory named after current date like YYYYMMDD. For example, 20210830
    - Move the file all-databases-backup.sql to /tmp/mysqldumps/<current date>/ directory

MYBACKUP.SH

```
#!/bin/sh
# The above line tells the interpreter this code needs to be run as a shell script.

# Set the database name to a variable.
DATABASE2='billing'

# This will be printed on to the screen. In the case of cron job, it will be printed to the logs.
echo "Pulling Database: This may take a few minutes"

# Set the folder where the database backup will be stored
backupfolder=/home/theia/backups

# Number of days to store the backup
keep_day=30

sqlfile=$backupfolder/all-databases-backup.sql
zipfile=$backupfolder/all-database-$(date +%d-%m-%Y_%H-%M-%S).gz

# Create a backup

if mysqldump  $DATABASE2 > $sqlfile ; then
   echo 'Sql dump created'
    # Compress backup
    if gzip -c $sqlfile > $zipfile; then
        echo 'The backup was successfully compressed'
    else
        echo 'Error compressing backupBackup was not created!'
        exit
    fi
    rm $sqlfile
else
   echo 'pg_dump return non-zero code No backup was created!'
   exit
fi

# Delete old backups
find $backupfolder -mtime +$keep_day -delete

```

# Exercise 3.1 - Restore data

### Task 3.1 - Restore the table billing

- Use the billing.csv and restore the CSV file into a table named billing. Write a query to display the first five rows of the table along with the number of rows imported.

QUERY:

- SELECT * FROM billing LIMIT 5;

# Exercise 3.2 - Create a view

### Task 3.2 - Create a view named basicbilldetails with the columns customerid, month, billedamount

QUERY:

- CREATE VIEW basicbilldetails AS SELECT customerid, month, billedamount FROM billing;

# Exercise 3.3 - Indexing

## Task 3.3 - Baseline query performance

- Write a query to find out all the rows with a billing amount of 19929.
- Hint: Use the command SELECT strftime("%Y-%m-%d %H:%M:%f", "now"); before and after your query to display the run time.

QUERY:

```
SELECT strftime('%Y-%m-%d %H:%M:%S:%f','now') AS start;
SELECT * FROM billing WHERE billedamount=19929;
SELECT strftime('%Y-%m-%d %H:%M:%S:%f','now') AS end;
```

## Task 3.4 - Create an index

Create an index that can make the query in the previous task faster. Name the index as billingamount.

QUERY:

```
CREATE INDEX billingamount ON billing(billedamount);
```

## Task 3.5 - Document the improvement in query performance

- Find out if the index has any impact on query performance.
- Re-run the query to find out all the rows with a billing amount of 19929.

QUERY:

```
SELECT strftime('%Y-%m-%d %H:%M:%S:%f','now') AS start;
SELECT * FROM billing WHERE billedamount=19929;
SELECT strftime('%Y-%m-%d %H:%M:%S:%f','now') AS end;
```