Final Assignment - Database Administration - Part 1

Estimated time needed: **45** minutes.

About This SN Labs Cloud IDE

This Skills Network Labs Cloud IDE provides a hands-on environment for course and project related labs. It utilizes Theia, an open-source IDE (Integrated Development Environment) platform, that can be run on desktop or on the cloud. To complete this lab, we will be using the Cloud IDE based on Theia and PostgreSQL database running in a Docker container.

Important Notice about this lab environment

Please be aware that sessions for this lab environment are not persisted. Every time you connect to this lab, a new environment is created for you. Any data you may have saved in the earlier session would get lost. Plan to complete these labs in a single session, to avoid losing your data.

Scenario

You have assumed the role of database administrator for the PostgreSQL server and you will perform the User Management tasks and handle the backup of the databases.

Objectives

In Part 1 of this assignment you will be working on the following aspects of Database Administration.

- Installation/Provisioning
- Configuration
- User Management
- Backup

Note - Screenshots

Throughout this lab you will be prompted to take screenshots and save them on your own device. These screenshots will need to be uploaded for peer review in the next section of the course. You can use various free screengrabbing tools to do this or use your operating system's shortcut keys to do this (for example *Alt+PrintScreen* in Windows).

Exercise 1.1 - Set up the lab environment

Before you proceed with the assignment

- 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

Task 1.1 - Find the settings in PostgreSQL

What is the maximum number of connections allowed for the postgres server on theia lab?

Hint: */home/project/postgres/data/postgresql.conf* is the configuration file for PostgreSQL.

Take a screenshot of the config file that clearly shows this information. Name the screenshot as **max-connections.jpg**. (images can be saved with either .jpg or .png extension)

Exercise 1.2 - User Management

Perform these user management tasks on your PostgreSQL server. Perform the tasks 1.2 to 1.5 using the PostgreSQL CLI. DO NOT USE THE PGADMIN GUI.

Task 1.2 - Create a User

Create a user named `backup_operator`.

Take a screenshot of the command you used and the output. Name the screenshot as **create-user.jpg**. (images can be saved with either .jpg or .png extension)

Task 1.3 - Create a Role

Create a role named `backup`.

Take a screenshot of the command you used and the output. Name the screenshot as **create-role.jpg**. (images can be saved with either .jpg or .png extension)

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.

Take a screenshot of the command you used and the output. Name the screenshot as **grant-privs-to-role.jpg**. (images can be saved with either .jpg or .png extension)

Task 1.5 - Grant role to an user

Grant the role `backup` to `backup_operator`

Take a screenshot of the command you used and the output. Name the screenshot as **grant-role.jpg**. (images can be saved with either .jpg or .png extension)

Exercise 1.3 - Backup

Task 1.6 - Backup a database on PostgreSQL server

Backup the database `tolldata` using PGADMIN GUI.

Backup the database `tolldata` into a file named `tolldatabackup.tar`, select the backup format as `Tar`

Take a screenshot of the window that shows the filename and format you have specified. Name the screenshot as **backup-database.jpg**. (images can be saved with either .jpg or .png extension)

End of assignment - Part 1.

Authors

Ramesh Sannareddy

Other Contributors

Rav Ahuja

Final Assignment - Database Administration - Part 2

Estimated time needed: **45** minutes.

About This SN Labs Cloud IDE

This Skills Network Labs Cloud IDE provides a hands-on environment for course and project related labs. It utilizes Theia, an open-source IDE (Integrated Development Environment) platform, that can be run on desktop or on the cloud. To complete this lab, we will be using the Cloud IDE based on Theia and MySQL database running in a Docker container.

Important Notice about this lab environment

Please be aware that sessions for this lab environment are not persisted. Every time you connect to this lab, a new environment is created for you. Any data you may have saved in the earlier session would get lost. Plan to complete these labs in a single session, to avoid losing your data.

Scenario

You have assumed the role of database administrator for the MySQL server and will perform the tasks like configuration check, recovery of data. You will use indexing to improve the database performance. You will identify which storage engines are supported by the server and which table uses which storage engine. Optionally You will also automate backup tasks.

Objectives

In part 2 of this assignment you will be working on the following aspects of Database Administration.

- Installing/Provisioning
- Configuration
- Recovery
- Indexing
- Storage Engines
- Automation of routine tasks

Note - Screenshots

Throughout this lab you will be prompted to take screenshots and save them on your own device. These screenshots need to be uploaded for peer review in the next section of the course. You can use various free screengrabbing tools to do this or use your operating system's shortcut keys to do this (for example *Alt+PrintScreen* in Windows).

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.

Take a screenshot of the list of tables. Name the screenshot as **database-restore.jpg**. (images can be saved with either .jpg or .png extension)

Task 2.2 - Find the table data size

Find the data size of the table billdata.

Take a screenshot of the command you used and the output.

Name the screenshot as **table-data-size.jpg**. (images can be saved with either .jpg or .png extension)

Exercise 2.3 - Indexing

Task 2.3 - Baseline query performance

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

Take a screenshot of the command you used and the output along with the query time. Name the screenshot as **query-base-line.jpg**. (images can be saved with either .jpg or .png extension)

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.

Take a screenshot of the command you used and the output. Name the screenshot as **index-creation.jpg**. (images can be saved with either .jpg or .png extension)

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.

Take a screenshot of the command you used and the output along with the query time. Name the screenshot as **query-indexed.jpg**. (images can be saved with either .jpg or .png extension)4

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.

Take a screenshot of the command you used and the output. Name the screenshot as **storage-engines.jpg**. (images can be saved with either .jpg or .png extension)

Task 2.7 - Find the storage engine of a table

Find the storage engine of the table `billdata`.

Take a screenshot of the command you used and the output. Name the screenshot as **storage-engine-type.jpg**. (images can be saved with either .jpg or .png extension)

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.sh`that 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

Take a screenshot of the bash script with the entire code clearly visible. Name the screenshot as **bash-script.jpg**. (images can be saved with either .jpg or .png extension)

End of assignment - Part 2.

Authors

Ramesh Sannareddy

Other Contributors

Rav Ahuja

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2021-08-30	0.1	Ramesh Sannareddy	Created initial version
2021-10-19	0.2	Alison Woolford	Updates after review
2021-10-22	0.3	Steve Hord	QA pass
2022-01-03	0.4	Lakshmi Holla	Made changes in Task 2.5

Final Assignment - Database Administration - Part 3

Estimated time needed: **45** minutes.

About This SN Labs Cloud IDE

This Skills Network Labs Cloud IDE provides a hands-on environment for course and project related labs. It utilizes Theia, an open-source IDE (Integrated Development Environment) platform, that can be run on desktop or on the cloud. To complete this lab, we will be using the Cloud IDE based on Theia running in a Docker container. You will also need an instance of DB2 running in IBM Cloud.

Important Notice about this lab environment

Please be aware that sessions for this lab environment are not persisted. Every time you connect to this lab, a new environment is created for you. Any data you may have saved in the earlier session would get lost. Plan to complete these labs in a single session, to avoid losing your data.

Scenario

You have been assigned the work to provison a cloud instance of IBM DB2 server and perform the tasks like restoration of data, index creation to improve the query performance. You will create views to make queries easier to write. Optionally You will also connect to the cloud instance of IBM DB2 server and from command line.

Objectives

In part 3 of this assignment you will be working on the following aspects of Database Administration.

- Restore data
- Indexing
- View creation
- Connecting from command line

Note - Screenshots

Throughout this lab you will be prompted to take screenshots and save them on your own device. These screenshots will be needed to be uploaded for peer review in the next section of the course. You can use various free screengrabbing tools to do this or use your operating system's shortcut keys to do this (for example *Alt+PrintScreen* in Windows).

Exercise 3.1 - Prepare the lab environment

Before you proceed with the assignment, you need to have access to a cloud instance of IBM DB2 database. If you do not have access, use the instructions in this lab Hands-on Lab: Sign up for IBM Cloud and Create a Db2 service instance to create a instance for yourself.

Download the file billing.csv

Exercise 3.2 - Restore data

Task 3.1 - Restore the table billing

Use the **billing.csv** you have downloaded earlier, restore the csv file into a table named billing.

Note: You will see that each column has data type and column width auto generated based on the content. Edit column attributes by clicking on the pencil icon next to the respective attributes to change the width of country column to varchar of 30 and month column to varchar of 7.

Take a screenshot of the import status clearly showing the number of rows imported.

Name the screenshot as **restore-table.jpg**. (images can be saved with either .jpg or .png extension)

Exercise 3.3 - Create a view

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

Take a screenshot of the sql statement used to create the view.

Name the screenshot as **create-view.jpg**. (images can be saved with either .jpg or .png extension)

Exercise 3.4 - Indexing

Task 3.3 - Baseline query performance

Write a query to find out all the rows with a billing amount of 19929.

Take a screenshot of the command you used along with the query run time.

Name the screenshot as **query-base-line-db2.jpg**. (images can be saved with either .jpg or .png extension)

Task 3.4 - Create an index

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

Take a screenshot of the sql statement you used and the output.

Name the screenshot as **index-creation-db2.jpg**. (images can be saved with either .jpg or .png extension)

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.

Take a screenshot of the command you used and the output along with the query time.

Note: Sometimes, the query time after index creation may increase. This can happen due to various factors like

- bandwidth at the time of firing the query
- the load on free cloud tier that your IBM DB2 instance uses is dynamic and other load may impact your query time

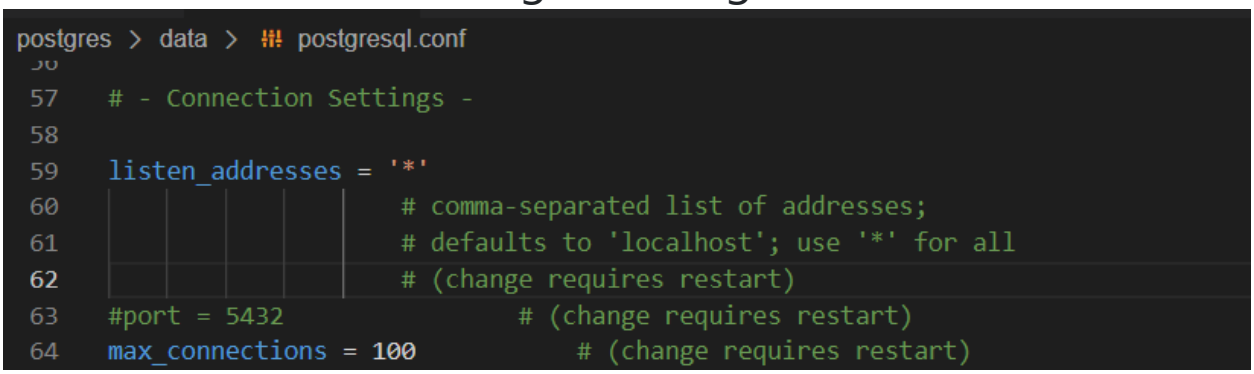
You will NOT be evaluated on the query run time. However, you are encouraged to run the query multiple times and pick the result with lowest query run time.

Name the screenshot as **query-after-index.jpg**. (images can be saved with either .jpg or .png extension)

OPTIONAL Exercise (Non-graded) - Connecting to IBM DB2 from command line

Bonus Task 3.6 - Connect to the clould instance of IBM DB2 using the db2cli command line tool

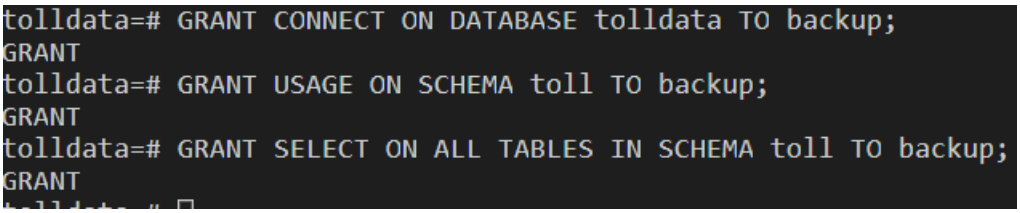
Task 1.1 - Find the settings in PostgreSQL



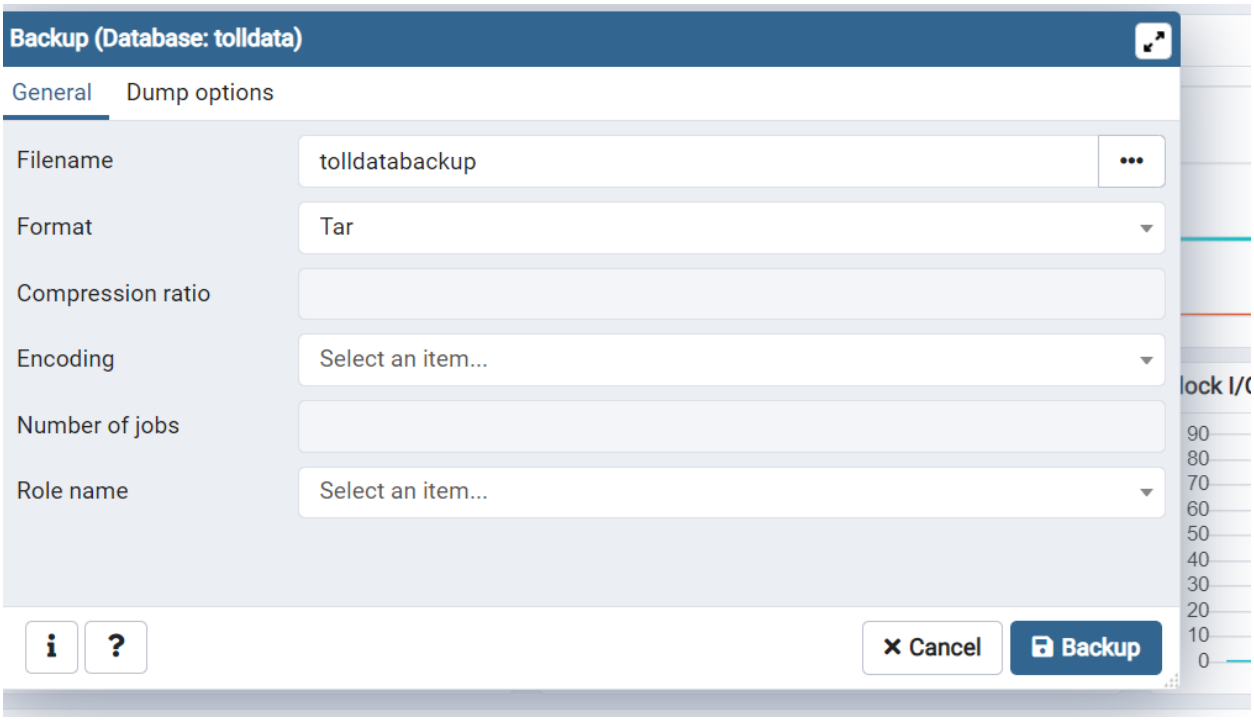
Task 1.2 - Create a User

Task 1.3 - Create a Role

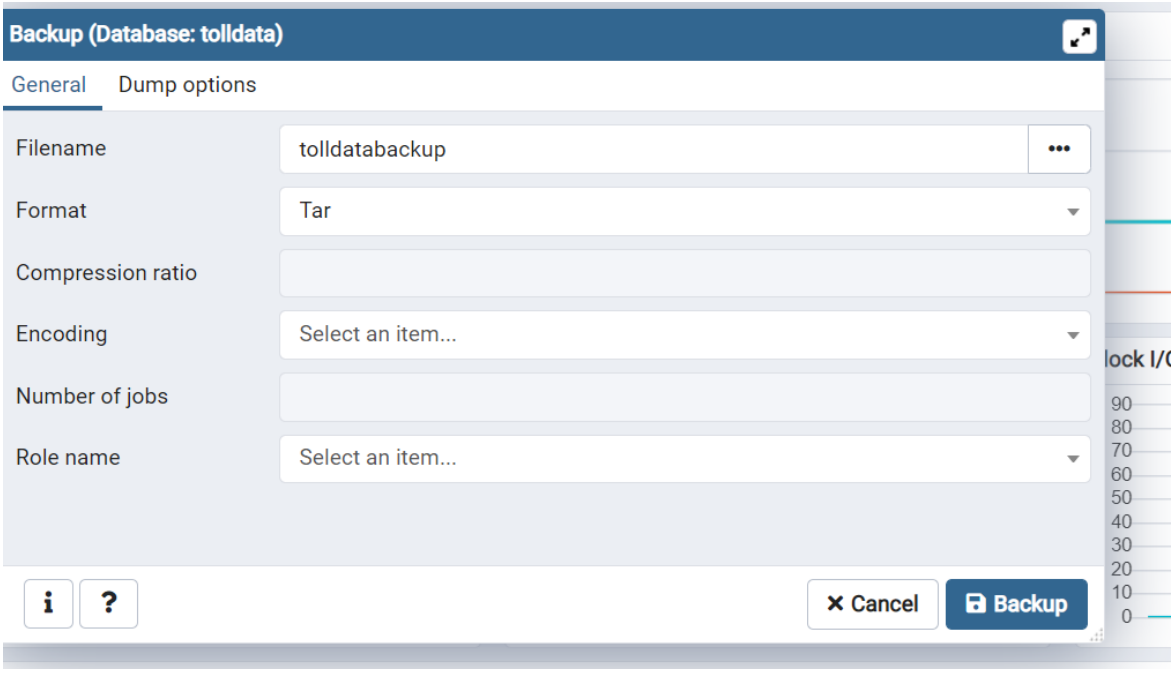
Task 1.4 - Grant privileges to the role



Task 1.5 - Grant role to an user



Task 1.6 - Backup a database on PostgreSQL server



Task 2.1 - Restore MySQL server using a previous backup

```
mysql> SOURCE billingdata.sql;
ERROR 1007 (HY000): Can't create database 'billing'; database exists
Database changed
Query OK, 0 rows affected (0.02 sec)

Query OK, 132000 rows affected (1.61 sec)
Records: 132000  Duplicates: 0  Warnings: 0

mysql> SHOW TABLES;
+------------------+
| Tables_in_billing |
+------------------+
| billdata          |
+------------------+
1 row in set (0.00 sec)
```

Task 2.2 - Find the table data size

```
mysql> SELECT TABLE_NAME AS `Table`,
    -> ROUND(((DATA_LENGTH + INDEX_LENGTH) / 1024 / 1024), 2) AS `Size (MB)`
    -> FROM information_schema.TABLES
    -> WHERE table_schema = "billing"
    -> ORDER BY (data_length + index_length);
+----------+-----------+
| Table    | Size (MB) |
+----------+-----------+
| billdata |      6.52 |
+----------+-----------+
1 row in set (0.00 sec)

mysql>
```

Task 2.3 - Baseline query performance

```
mysql> SELECT * FROM billdata WHERE billedamount > 19999;
+--------+------------+--------------+---------+
| billid | customerid | billedamount | monthid |
+--------+------------+--------------+---------+
|   8509 |        285 |        20000 |   20096 |
|  68268 |        559 |        20000 |   20146 |
|  81622 |        643 |        20000 |   20157 |
|  84858 |        317 |        20000 |   20161 |
|  89353 |        871 |        20000 |   20163 |
| 102682 |        937 |        20000 |   20174 |
| 109574 |        386 |        20000 |  201810 |
| 121844 |        777 |        20000 |  201910 |
+--------+------------+--------------+---------+
8 rows in set (0.07 sec)
```

Task 2.4 - Create an index

```
mysql> CREATE INDEX amount ON billdata (billedamount);
Query OK, 0 rows affected (0.49 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Task 2.5 - Document the improvement in query performance

```
mysql> SELECT * FROM billdata WHERE billedamount > 19999;
+--------+------------+--------------+---------+
| billid | customerid | billedamount | monthid |
+--------+------------+--------------+---------+
|   8509 |        285 |        20000 |   20096 |
|  68268 |        559 |        20000 |   20146 |
|  81622 |        643 |        20000 |   20157 |
|  84858 |        317 |        20000 |   20161 |
|  89353 |        871 |        20000 |   20163 |
| 102682 |        937 |        20000 |   20174 |
| 109574 |        386 |        20000 |  201810 |
| 121844 |        777 |        20000 |  201910 |
+--------+------------+--------------+---------+
8 rows in set (0.00 sec)
```

Task 2.6 - Find supported storage engines

```
mysql> SHOW ENGINES
    -> ;
+--------------------+---------+----------------------------------------------------------------+--------------+------+------------+
| Engine             | Support | Comment                                                        | Transactions | XA   | Savepoints |
+--------------------+---------+----------------------------------------------------------------+--------------+------+------------+
| FEDERATED          | NO      | Federated MySQL storage engine                                 | NULL         | NULL | NULL       |
| MEMORY             | YES     | Hash based, stored in memory, useful for temporary tables      | NO           | NO   | NO         |
| InnoDB             | DEFAULT | Supports transactions, row-level locking, and foreign keys     | YES          | YES  | YES        |
| PERFORMANCE_SCHEMA | YES     | Performance Schema                                             | NO           | NO   | NO         |
| MyISAM             | YES     | MyISAM storage engine                                          | NO           | NO   | NO         |
| MRG_MYISAM         | YES     | Collection of identical MyISAM tables                          | NO           | NO   | NO         |
| BLACKHOLE          | YES     | /dev/null storage engine (anything you write to it disappears) | NO           | NO   | NO         |
| CSV                | YES     | CSV storage engine                                             | NO           | NO   | NO         |
| ARCHIVE            | YES     | Archive storage engine                                         | NO           | NO   | NO         |
+--------------------+---------+----------------------------------------------------------------+--------------+------+------------+
9 rows in set (0.00 sec)
```

Task 2.7 - Find the storage engine of a table

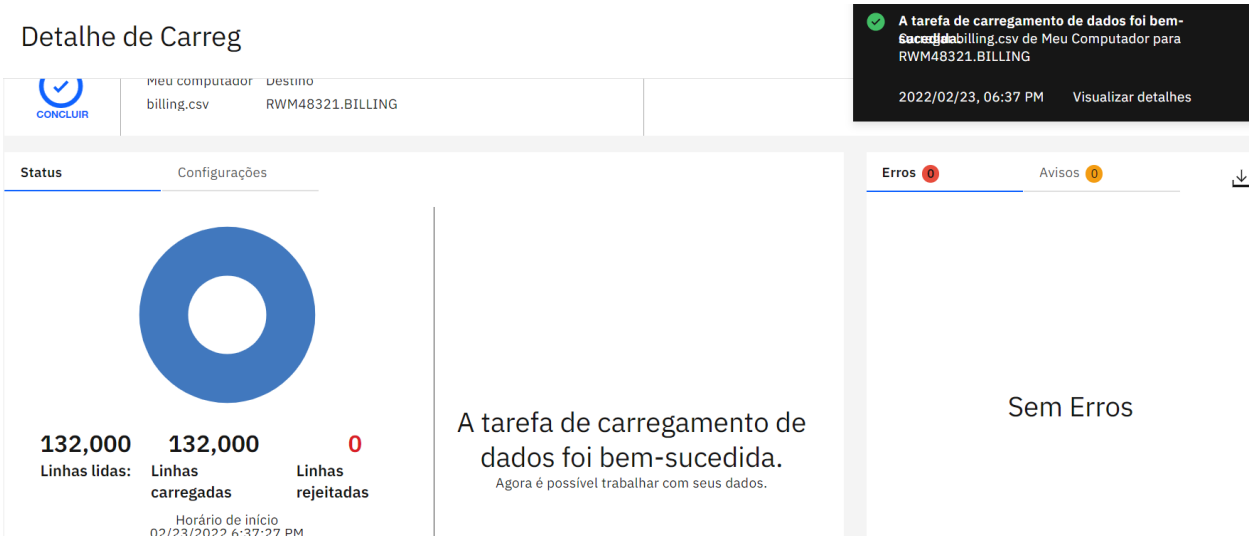
b

```
mysql> SELECT TABLE_NAME, ENGINE FROM information_schema.TABLES where TABLE_SCHEMA = 'billing';
+------------+--------+
| TABLE_NAME | ENGINE |
+------------+--------+
| billdata   | InnoDB |
+------------+--------+
1 row in set (0.01 sec)
```

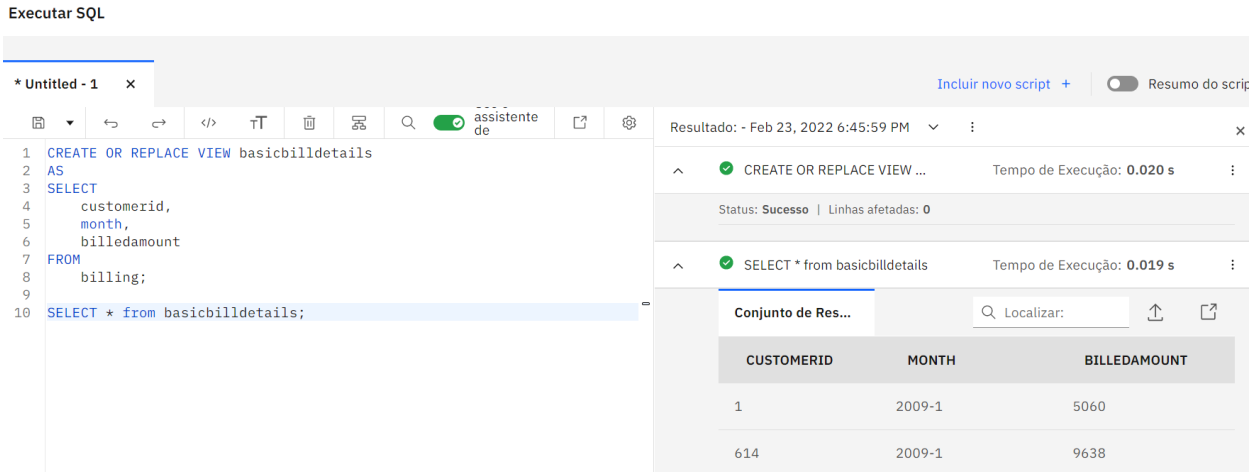
Bonus Task 2.8 - Write a bash script that performs a backup of all the Databases

```
#!/bin/sh

# Set the database name to a variable.

DATABASE='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=$backupfolder/all-databases-backup-$(date +%d-%m-%Y_%H-%M-%S).sql

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

# Create a backup

if mysqldump $DATABASE > $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
```

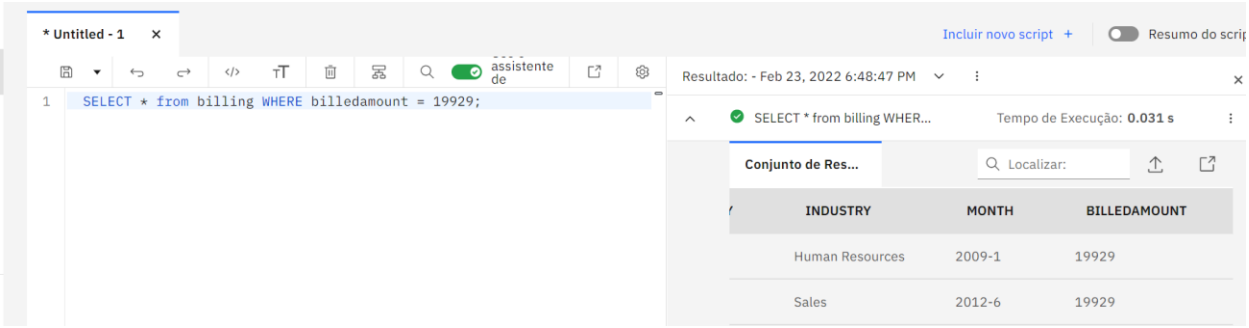
Task 3.1 - Restore the table billing



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

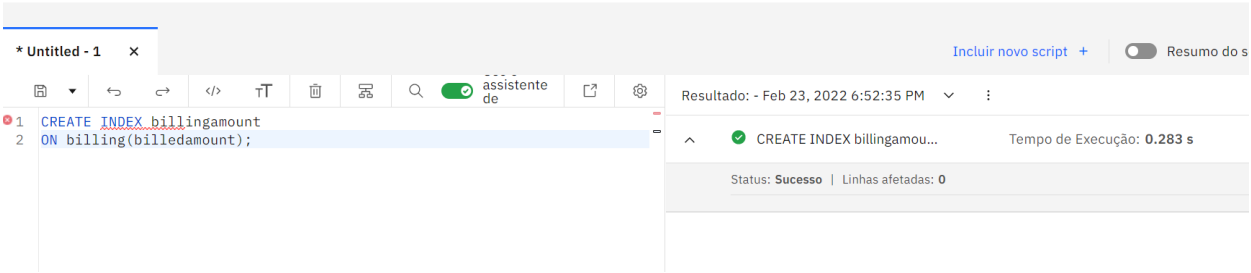


Task 3.3 - Baseline query performance



Task 3.4 - Create an index

Executar SQL



Task 3.5 - Document the improvement in query performance