Date: **Task No: 1**

Design a multi-dimensional data model schema namely Star, Snowflake and Fact Constellations for a Categorical data using SQL Server Management Studio (SSMS). (Perform the above for Banking, Healthcare, Manufacturing, Sales and Automobile)

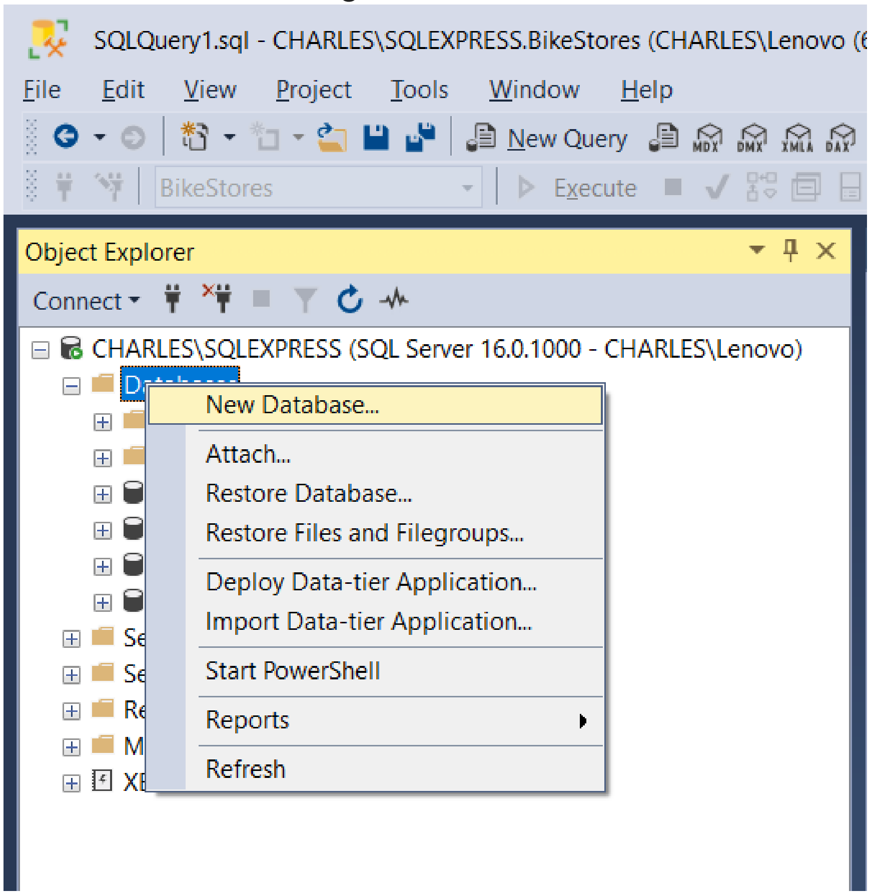
AIM:

PROCEDURE:

To practice creating a star schema data model from scratch, install SQL Server Management Studio, Microsoft SQL Server

Step 1: Install Diagram Support.

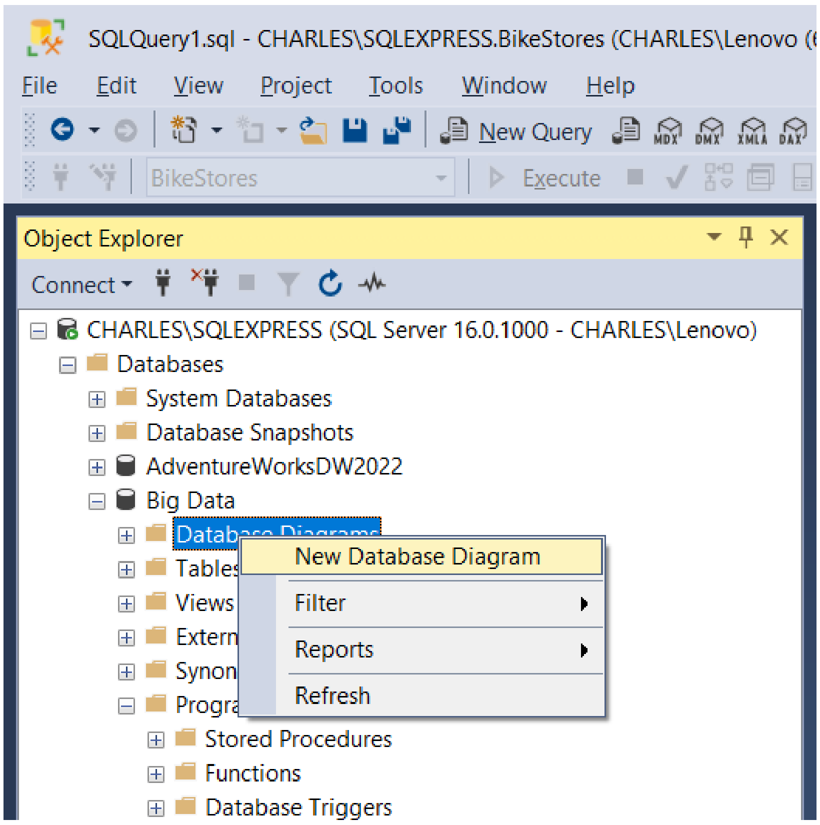
- Create a new database "BigData"


SQLQuery1.sql - CHARLES\SQLEXPRESS.BikeStores (CHARLES\Lenovo (
File
Edit
View
Project
Tools
Window
Help
New Query
BikeStores
Execute
Object Explorer
Connect
CHARLES\SQLEXPRESS (SQL Server 16.0.1000 - CHARLES\Lenovo)
New Database...
Attach...
Restore Database...
Restore Files and Filegroups...
Deploy Data-tier Application...
Import Data-tier Application...
Start PowerShell
Reports
Refresh

- Databases -> BigData -> Database Diagrams > Install Diagram Support
- Select the "Yes" in the pop-up window to close the window to one or more create support objects

Step 2: Create New Database Diagram.

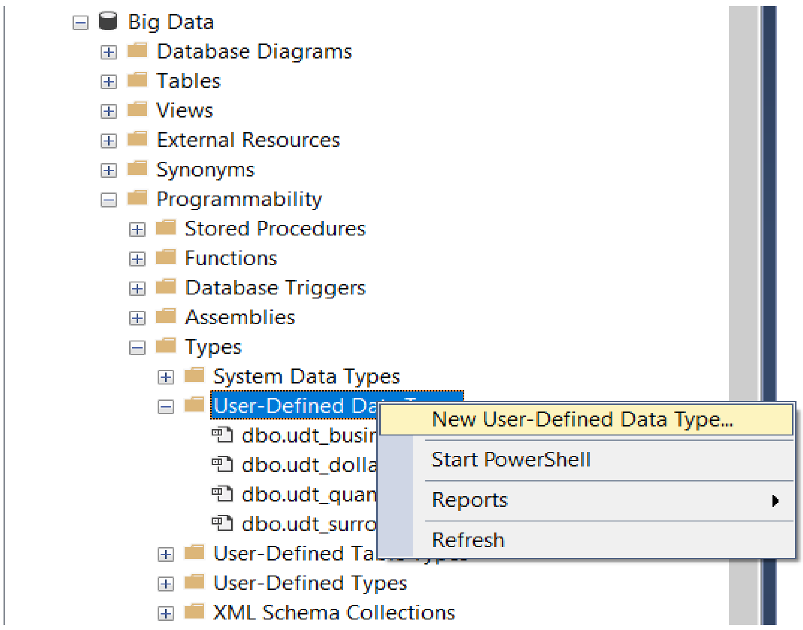
- Right-click on the menu item "Database Diagrams" and select the "New Database Diagram" item.



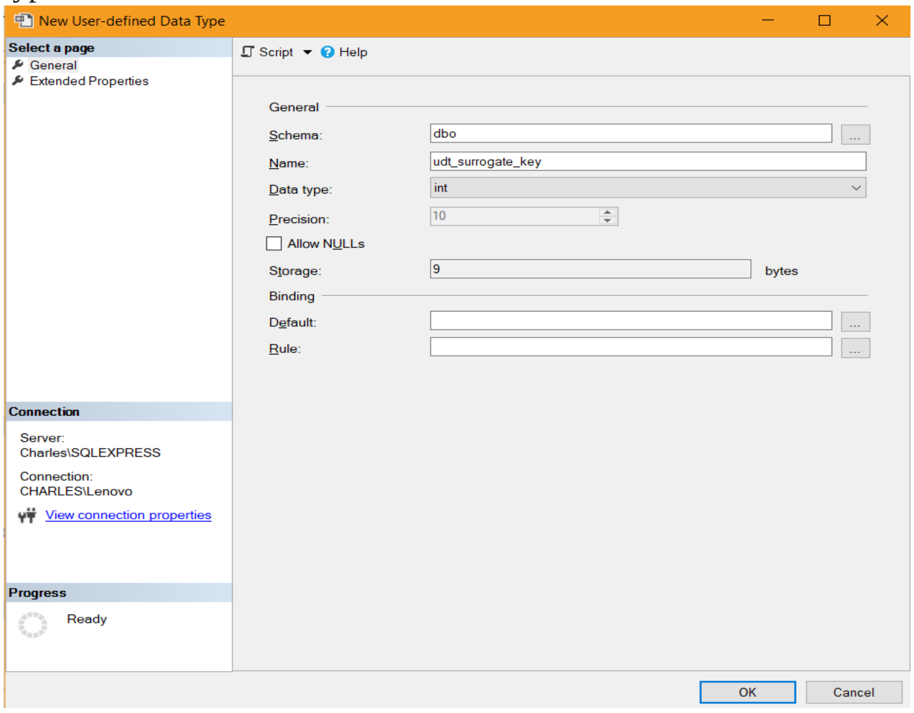
- A pop-up window with a title "Add Table" shows up. Click on the "Close" button at the bottom of the new window to close the window

Step 3: Create User-Defined Data Types

- Right-click on the item "Databases -> data_modeling -> Programmability -> Types -> User-Defined Data Types" in the Object Explore panel and select the menu item "New User-Defined Data Type" in the context menu.



- A new window appears. Enter a value "udt_surrogate_key" in the name field and select the data type with "int". Then, click on the "OK" button to create a new data type.



General

Schema: dbo

Name: udt_surrogate_key

Data type: int

Precision: 10

Allow NULLs [ ]

Storage: 9 bytes

Binding

Default:

Rule:

Connection

Server:
Charles\SQLEXPRESS

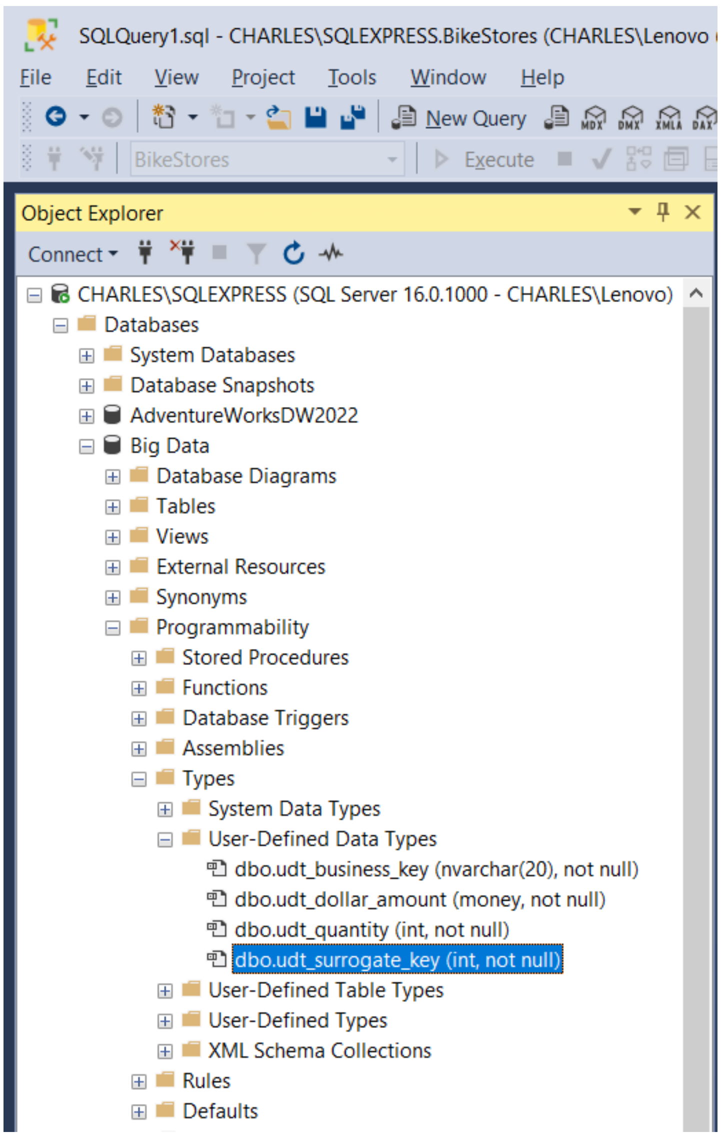
Connection:
CHARLES\Lenovo

View connection properties

Progress

Ready

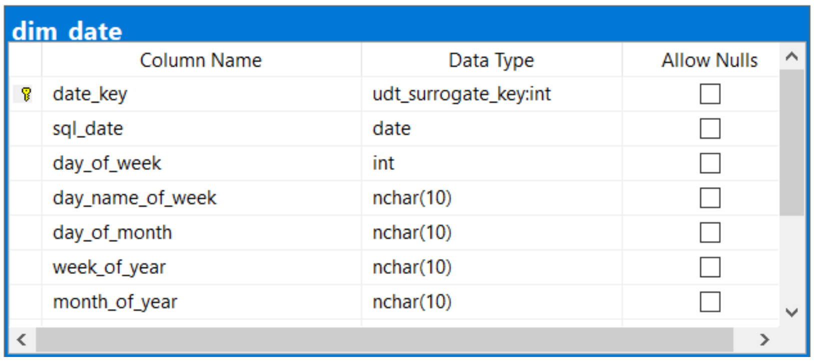
- Follow the preceding procedure described in this step, create other user-defined data types.


SQLQuery1.sql - CHARLES\SQLEXPRESS.BikeStores (CHARLES\Lenovo

- CHARLES\SQLEXPRESS (SQL Server 16.0.1000 - CHARLES\Lenovo)
  - Databases
    - System Databases
    - Database Snapshots
    - AdventureWorksDW2022
    - Big Data
      - Database Diagrams
      - Tables
      - Views
      - External Resources
      - Synonyms
      - Programmability
        - Stored Procedures
        - Functions
        - Database Triggers
        - Assemblies
        - Types
          - System Data Types
          - User-Defined Data Types
            - dbo.udt_business_key (nvarchar(20), not null)
            - dbo.udt_dollar_amount (money, not null)
            - dbo.udt_quantity (int, not null)
            - dbo.udt_surrogate_key (int, not null)
          - User-Defined Table Types
          - User-Defined Types
          - XML Schema Collections
        - Rules
        - Defaults

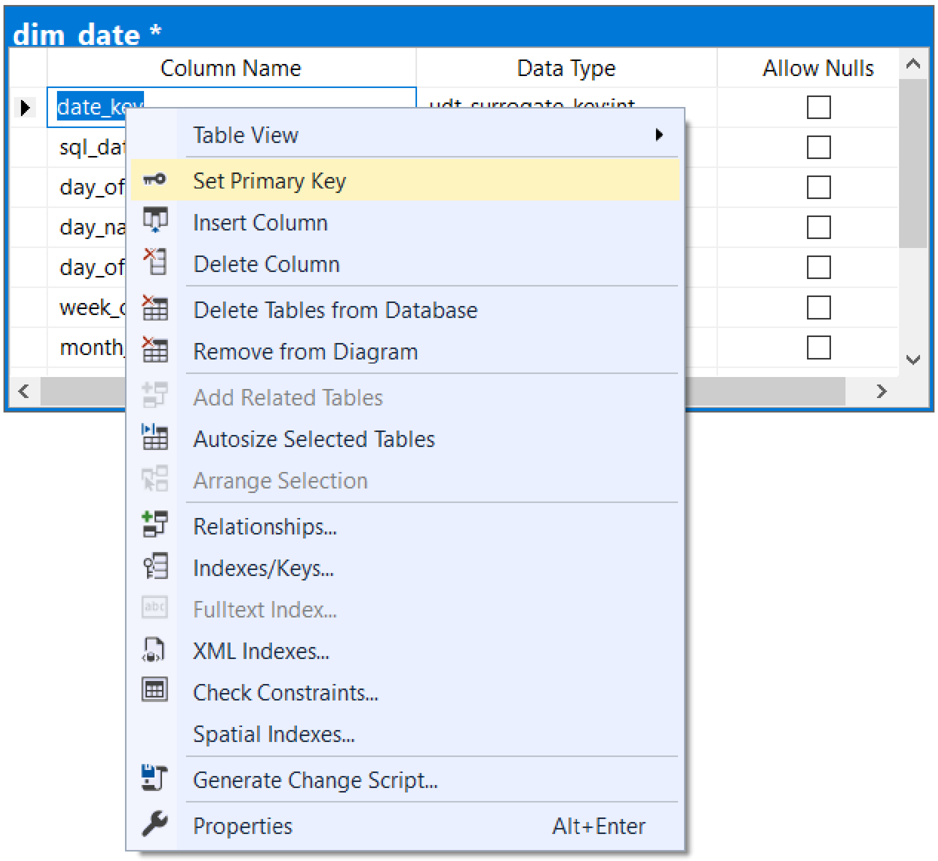
Step 4: Create a Dimension Table in SSMS.

- Right-click on an empty space in the middle panel, select the “New Table…” menu item in the context menu
- A small pop-up window shows up and enter a value “dim_date” as the table name. Click the “OK” button to close this window and start a new pop-up window.
- In the new window, enter a value “date_key” as a column name and select the user-defined data type “udt_surrogate_key” as the data type. Then uncheck the “Allow Nulls” checkbox.


dim_date

Column Name	Data Type	Allow Nulls
date_key	udt_surrogate_key:int	☐
sql_date	date	☐
day_of_week	int	☐
day_name_of_week	nchar(10)	☐
day_of_month	nchar(10)	☐
week_of_year	nchar(10)	☐
month_of_year	nchar(10)	☐

- Right-click on trigonal icon beside the text “date_key”, select the “Set Primary Key” menu item in the context menu.

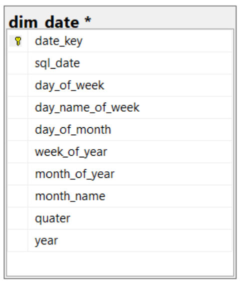


Step 5: Save the New Diagram.

- Click on the “Save” button on the window’s toolbar.
- A pop-up window shows up to ask a name of the diagram. Enter a value “fact_sale_order” as the name for the diagram then click on the “OK” button.

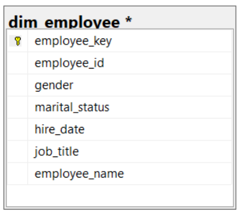
Step 6: Create All Dimension Tables.

- Repeat Step 5 to create other dimension tables.



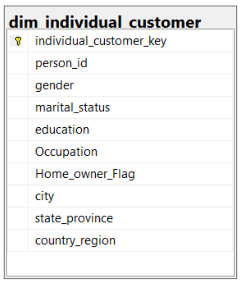
dim_date *

- date_key
- sql_date
- day_of_week
- day_name_of_week
- day_of_month
- week_of_year
- month_of_year
- month_name
- quater
- year



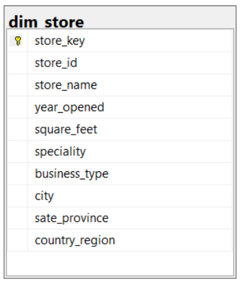
dim_employee *

- employee_key
- employee_id
- gender
- marital_status
- hire_date
- job_title
- employee_name



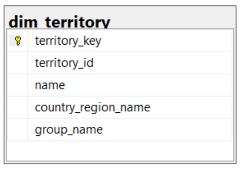
dim_individual_customer

- individual_customer_key
- person_id
- gender
- marital_status
- education
- Occupation
- Home_owner_Flag
- city
- state_province
- country_region



dim_store

- store_key
- store_id
- store_name
- year_opened
- square_feet
- speciality
- business_type
- city
- sate_province
- country_region

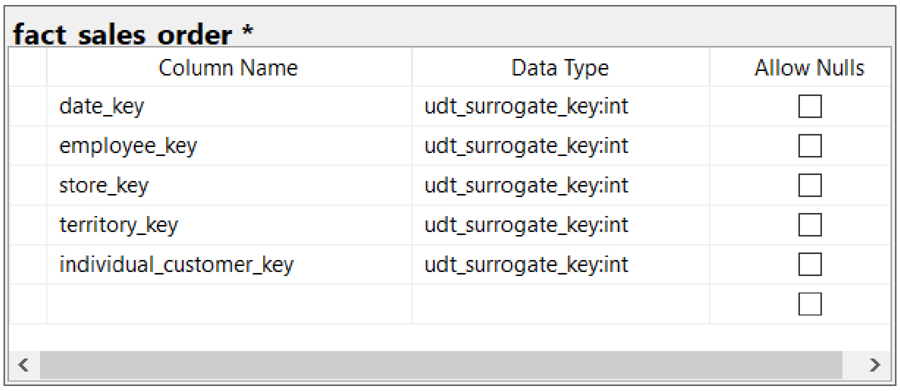


dim_territory

- territory_key
- territory_id
- name
- country_region_name
- group_name

Step 7: Create a Fact Table.

- Right-click on an empty space in the diagram panel, select the "New Table…" menu item from the context menu
- enter a value "fact_sales_order" as the fact table name in the new pop-up window.
- Click the window by clicking on the "OK" button.
- The second pop-up window appears. Enter all dimension table keys and select the user-defined data type "udt_surrogate_key" as their data types. Uncheck all "Allow Nulls" checkboxes.

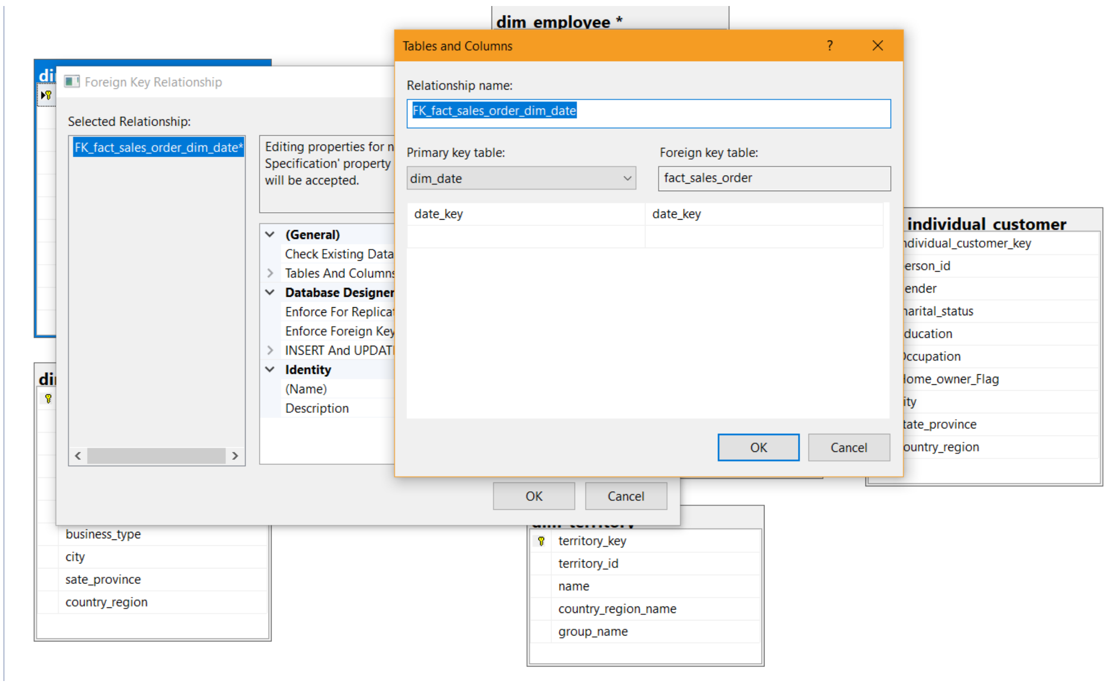


fact_sales_order *

Column Name	Data Type	Allow Nulls
date_key	udt_surrogate_key:int	☐
employee_key	udt_surrogate_key:int	☐
store_key	udt_surrogate_key:int	☐
territory_key	udt_surrogate_key:int	☐
individual_customer_key	udt_surrogate_key:int	☐
		☐

Step 8: Establish Relationship Between Dimension tables and Fact table

- In the diagram panel, drag the key icon in the table "dim_table" over the "date_key" in the table "fact_sales_order". Two pop-up windows appear immediately.
- In the first pop-up window presents the relationship name, "FK_fact_sales_order_dim_date". All the foreign names will use this format.
- verify if the keys are linked correctly and click OK button



Tables and Columns

Relationship name:

FK_fact_sales_order_dim_date

Primary key table:	Foreign key table:
dim_date	fact_sales_order
date_key	date_key

OK Cancel

Foreign Key Relationship

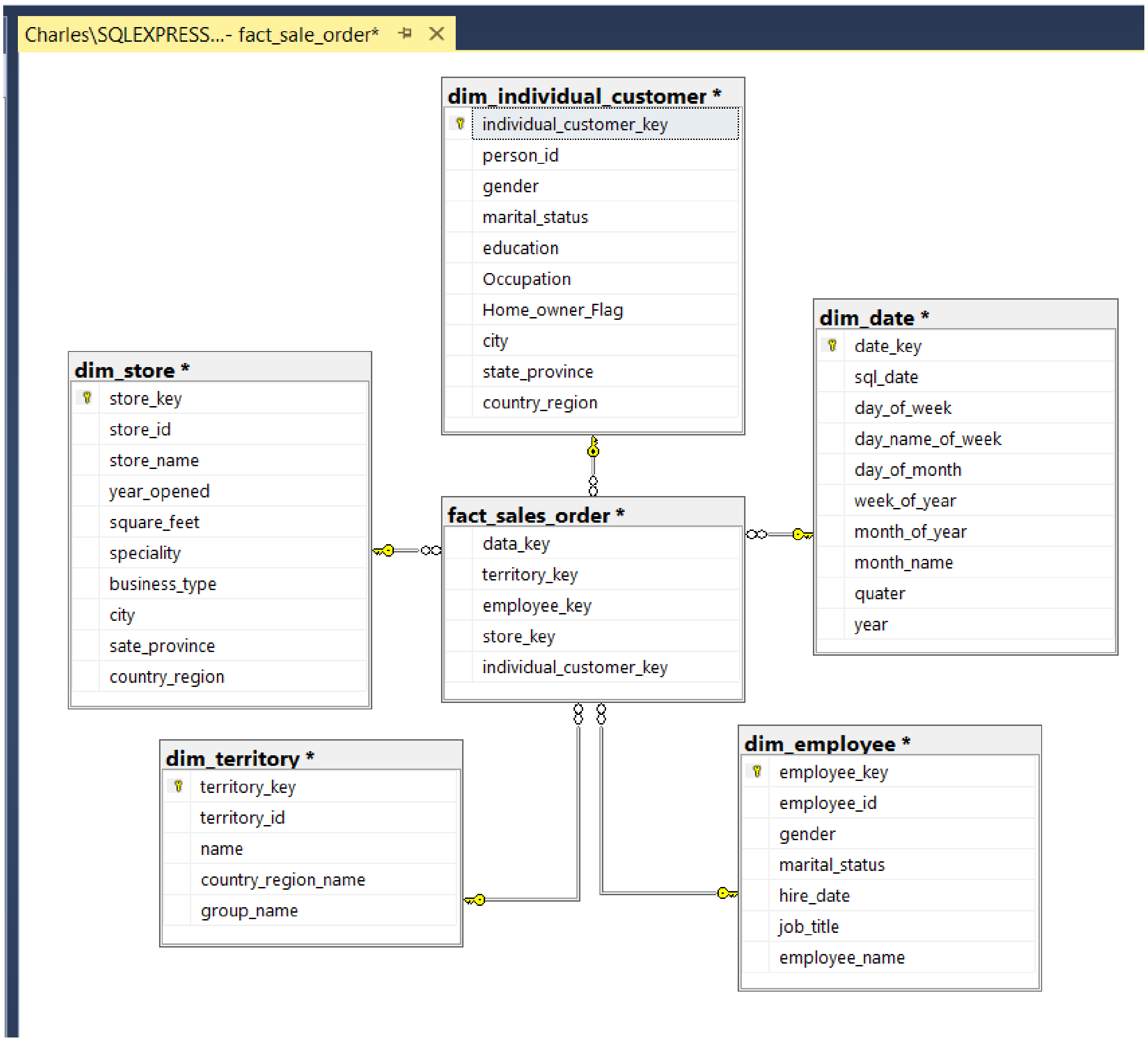
Selected Relationship:

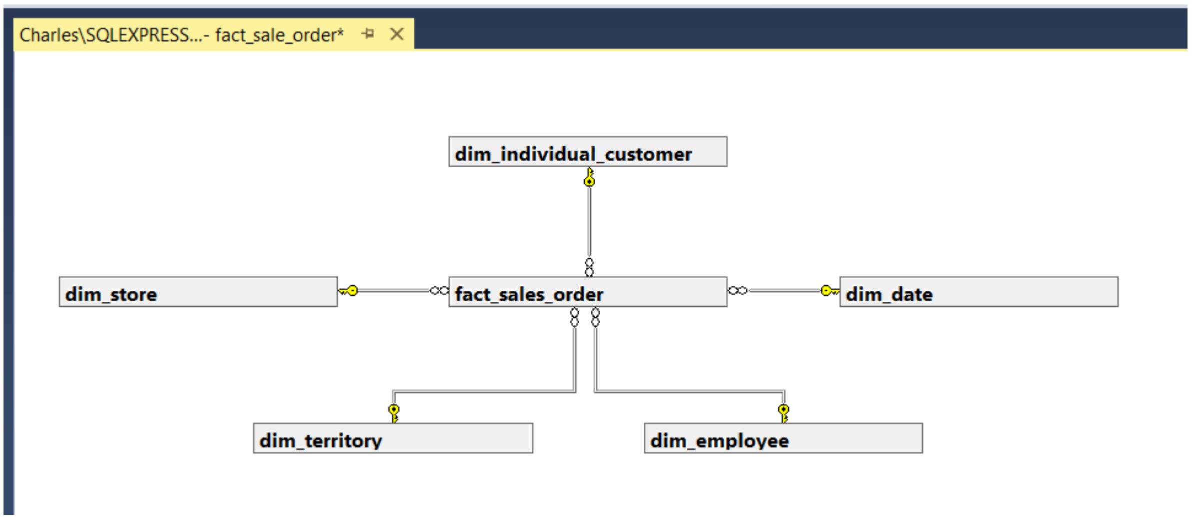
FK_fact_sales_order_dim_date*

Editing properties for n Specification' property will be accepted.

(General)
Check Existing Data
Tables And Columns
Database Designer
Enforce For Replicat
Enforce Foreign Key
INSERT And UPDAT
Identity
(Name)
Description

OK Cancel




Charles\SQLEXPRESS...- fact_sale_order*
dim_individual_customer
dim_store
fact_sales_order
dim_date
dim_territory
dim_employee

RESULT:

Date:

Task No: 2

To configure, monitor, and administer a Data warehouse and perform basic Query operations on the DW.

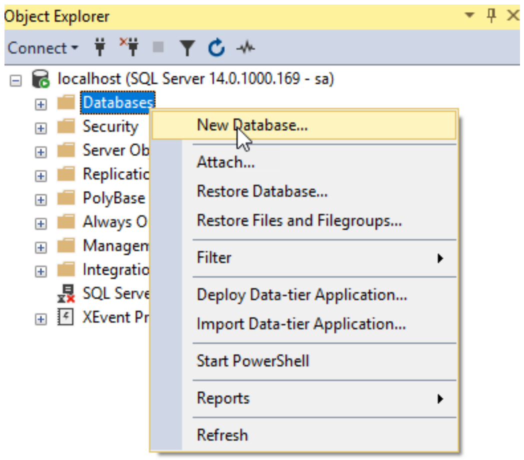
AIM:

TASK 2a:

PROCEDURE:

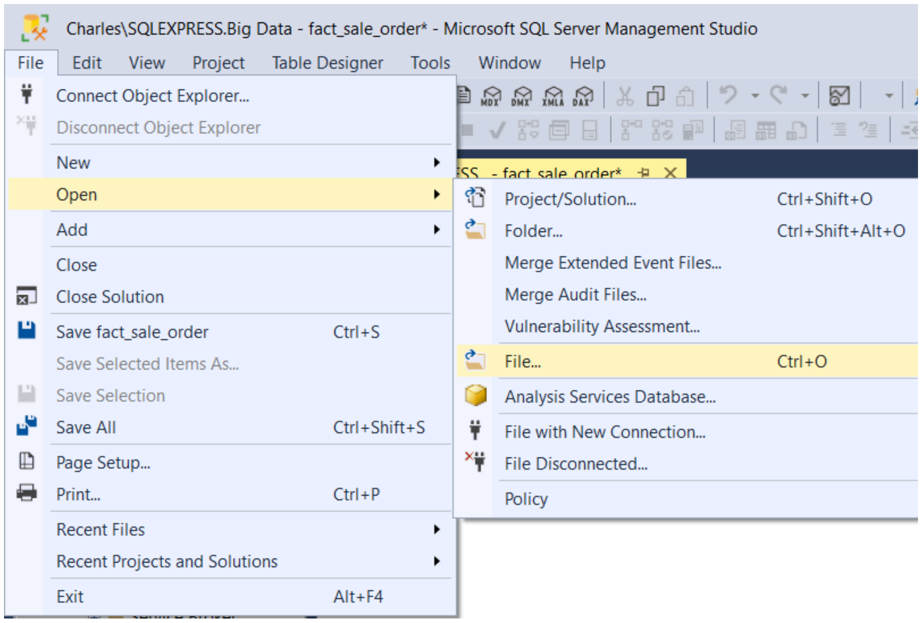
Connect to the SQL Server by choosing the server name, enter the user and password and click the Connect button.

Right-click the Databases node in the Object Explorer and select the New Database… menu item

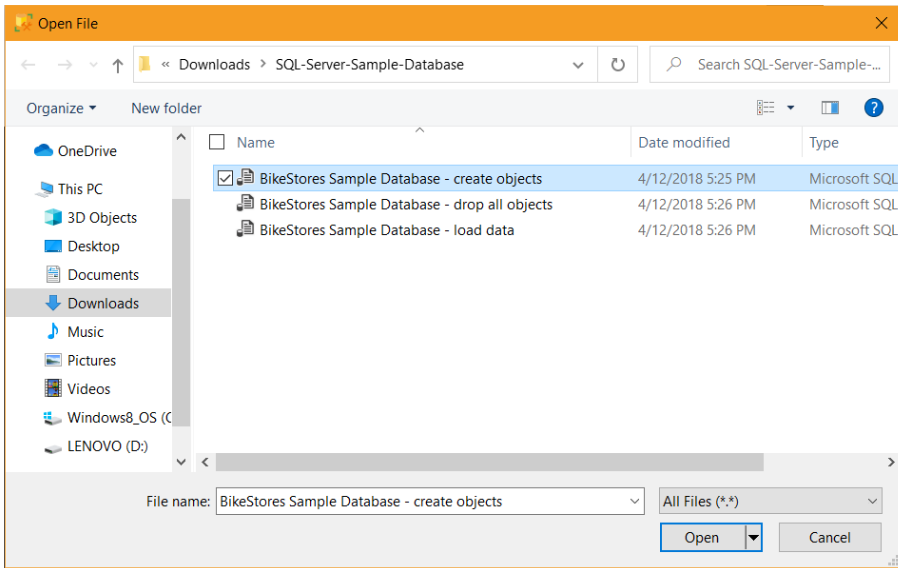


Enter the **Database name** as BikeStores and click the **OK** button to create the new database.

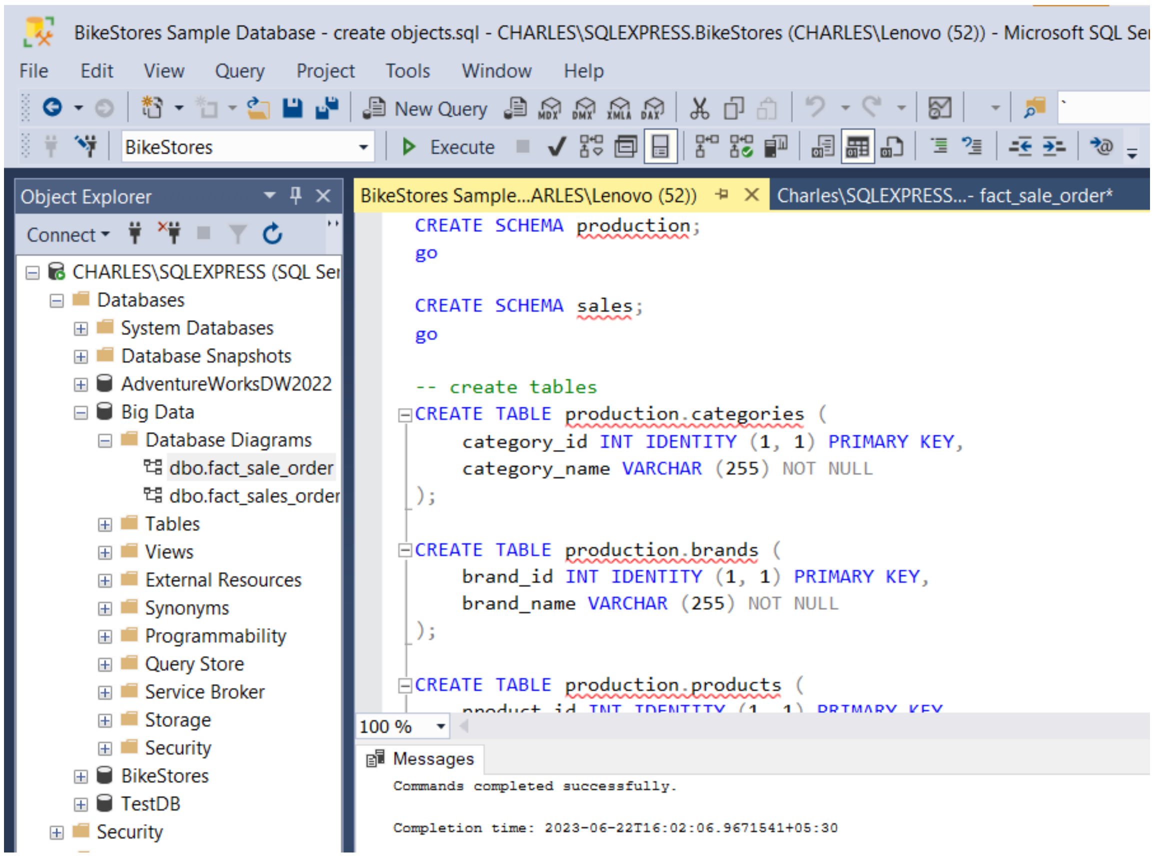
From the File menu, choose Open > File… menu item to open a script file.


Charles\SQLEXPRESS.Big Data - fact_sale_order* - Microsoft SQL Server Management Studio
File Edit View Project Table Designer Tools Window Help
Connect Object Explorer...
Disconnect Object Explorer
New
Open
Add
Close
Close Solution
Save fact_sale_order Ctrl+S
Save Selected Items As...
Save Selection
Save All Ctrl+Shift+S
Page Setup...
Print... Ctrl+P
Recent Files
Recent Projects and Solutions
Exit Alt+F4
Project/Solution... Ctrl+Shift+O
Folder... Ctrl+Shift+Alt+O
Merge Extended Event Files...
Merge Audit Files...
Vulnerability Assessment...
File... Ctrl+O
Analysis Services Database...
File with New Connection...
File Disconnected...
Policy

Select the **BikeStores Sample Database – create** objects.sql file and click the Open button


Open File
Downloads › SQL-Server-Sample-Database
Search SQL-Server-Sample-...
Organize
New folder
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows8_OS (C
LENOVO (D:)
Name
Date modified
Type
BikeStores Sample Database - create objects 4/12/2018 5:25 PM Microsoft SQL
BikeStores Sample Database - drop all objects 4/12/2018 5:26 PM Microsoft SQL
BikeStores Sample Database - load data 4/12/2018 5:26 PM Microsoft SQL
File name: BikeStores Sample Database - create objects
All Files (*.*)
Open
Cancel

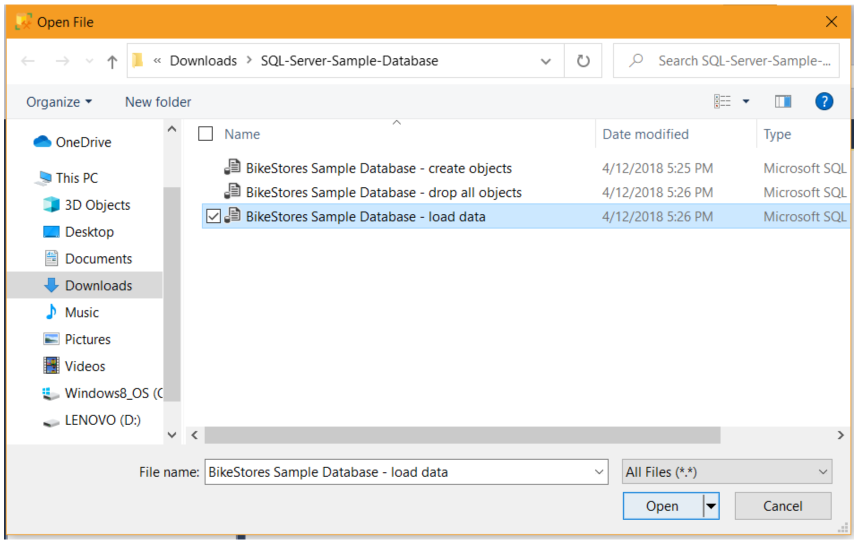
Click the **Execute** button to execute the SQL script



```
CREATE SCHEMA production;
go

CREATE SCHEMA sales;
go

-- create tables
CREATE TABLE production.categories (
    category_id INT IDENTITY (1, 1) PRIMARY KEY,
    category_name VARCHAR (255) NOT NULL
);

CREATE TABLE production.brands (
    brand_id INT IDENTITY (1, 1) PRIMARY KEY,
    brand_name VARCHAR (255) NOT NULL
);

CREATE TABLE production.products (
```

```
Commands completed successfully.

Completion time: 2023-06-22T16:02:06.9671541+05:30
```

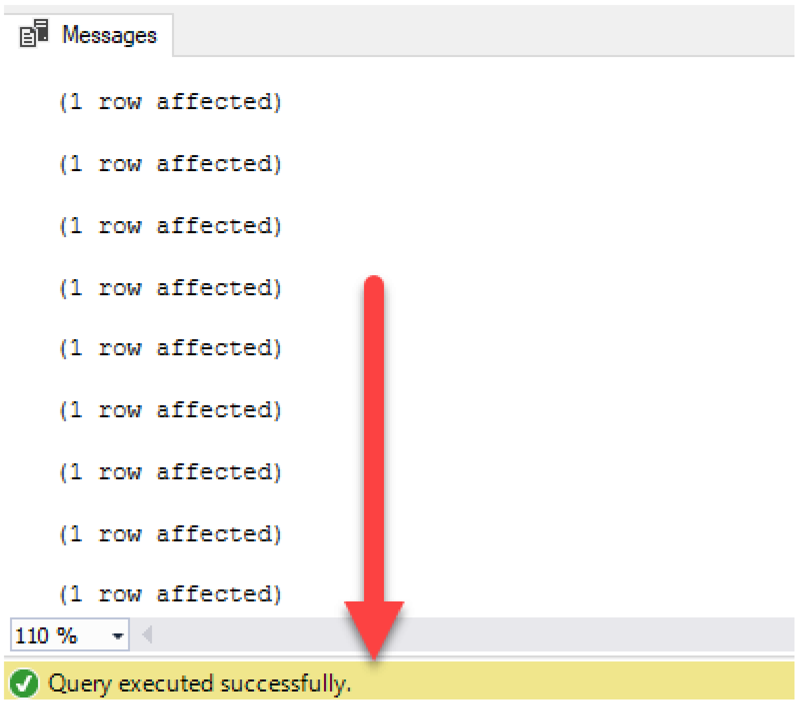
Choose the **BikeStores Sample Database – load data.sql** file and click the Open button.



Name	Date modified	Type
BikeStores Sample Database - create objects	4/12/2018 5:25 PM	Microsoft SQL
BikeStores Sample Database - drop all objects	4/12/2018 5:26 PM	Microsoft SQL
BikeStores Sample Database - load data	4/12/2018 5:26 PM	Microsoft SQL

File name: BikeStores Sample Database - load data
All Files (*.*)

Click the **Execute** button to load data into the tables.



```
Messages

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)
```

Query executed successfully.

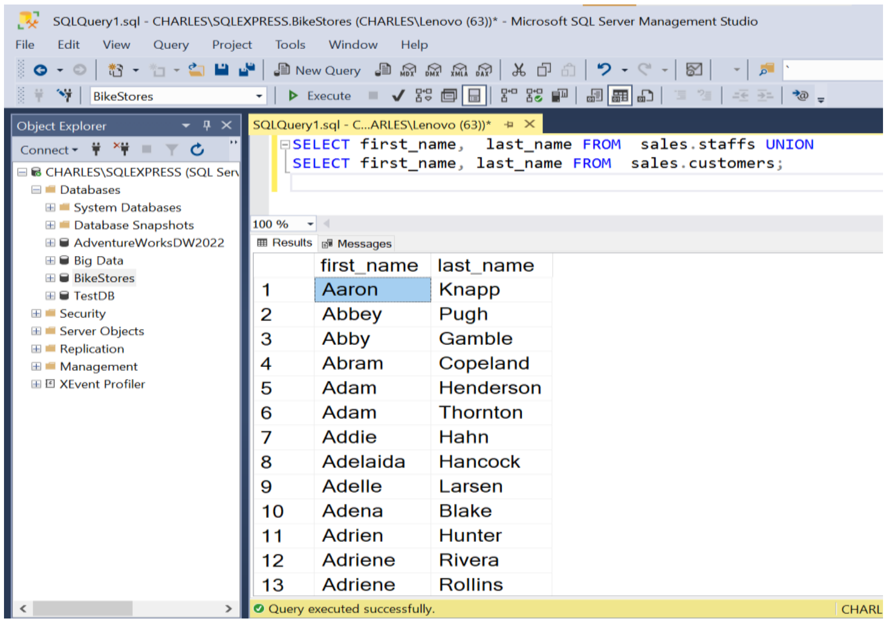
Execute query to display the values from `production.products table`.

QUERY OPERATIONS ON THE DATA WAREHOUSE

UNION OPERATION

SELECT first_name, last_name FROM sales.staffs UNION

SELECT first_name, last_name FROM sales.customers;



```
SELECT first_name,  last_name FROM  sales.staffs UNION
SELECT first_name, last_name FROM  sales.customers;
```

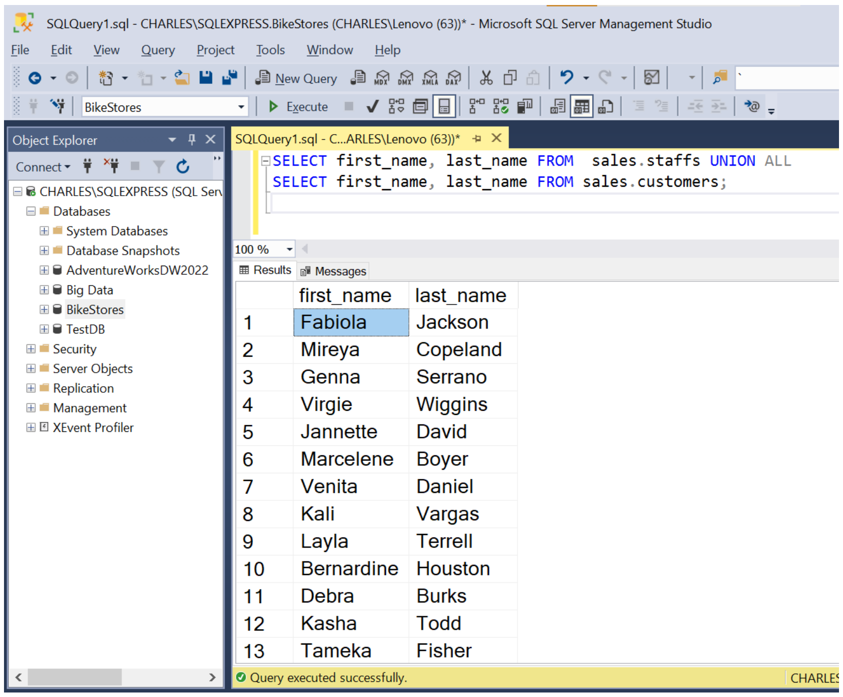
	first_name	last_name
1	Aaron	Knapp
2	Abbey	Pugh
3	Abby	Gamble
4	Abram	Copeland
5	Adam	Henderson
6	Adam	Thornton
7	Addie	Hahn
8	Adelaida	Hancock
9	Adelle	Larsen
10	Adena	Blake
11	Adrien	Hunter
12	Adriene	Rivera
13	Adriene	Rollins

Query executed successfully.

UNION ALL OPERATION

SELECT first_name, last_name FROM sales.staffs UNION ALL

SELECT first_name, last_name FROM sales.customers;



```
SELECT first_name, last_name FROM  sales.staffs UNION ALL
SELECT first_name, last_name FROM sales.customers;
```

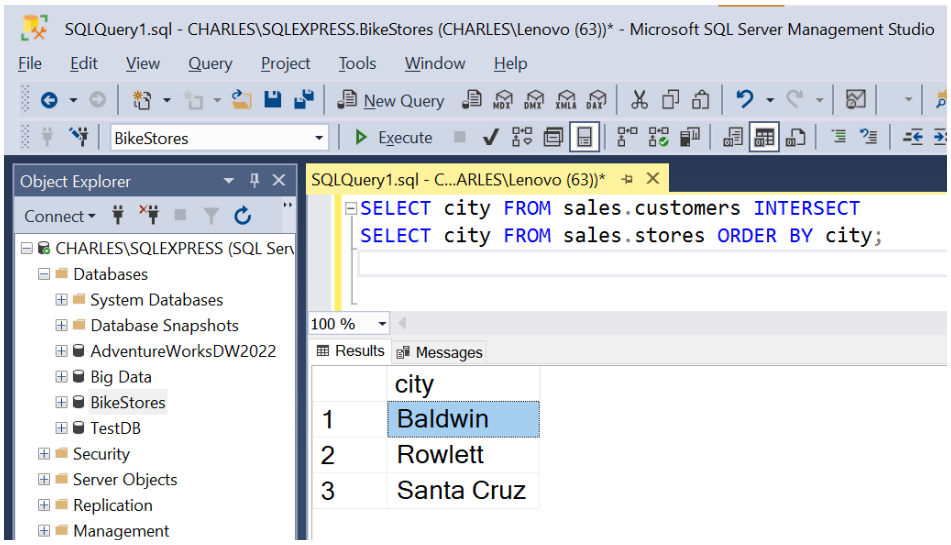
	first_name	last_name
1	Fabiola	Jackson
2	Mireya	Copeland
3	Genna	Serrano
4	Virgie	Wiggins
5	Jannette	David
6	Marcelene	Boyer
7	Venita	Daniel
8	Kali	Vargas
9	Layla	Terrell
10	Bernardine	Houston
11	Debra	Burks
12	Kasha	Todd
13	Tameka	Fisher

Query executed successfully.

INTERSECT OPERATION

SELECT city FROM sales.customers INTERSECT

SELECT city FROM sales.stores ORDER BY city;



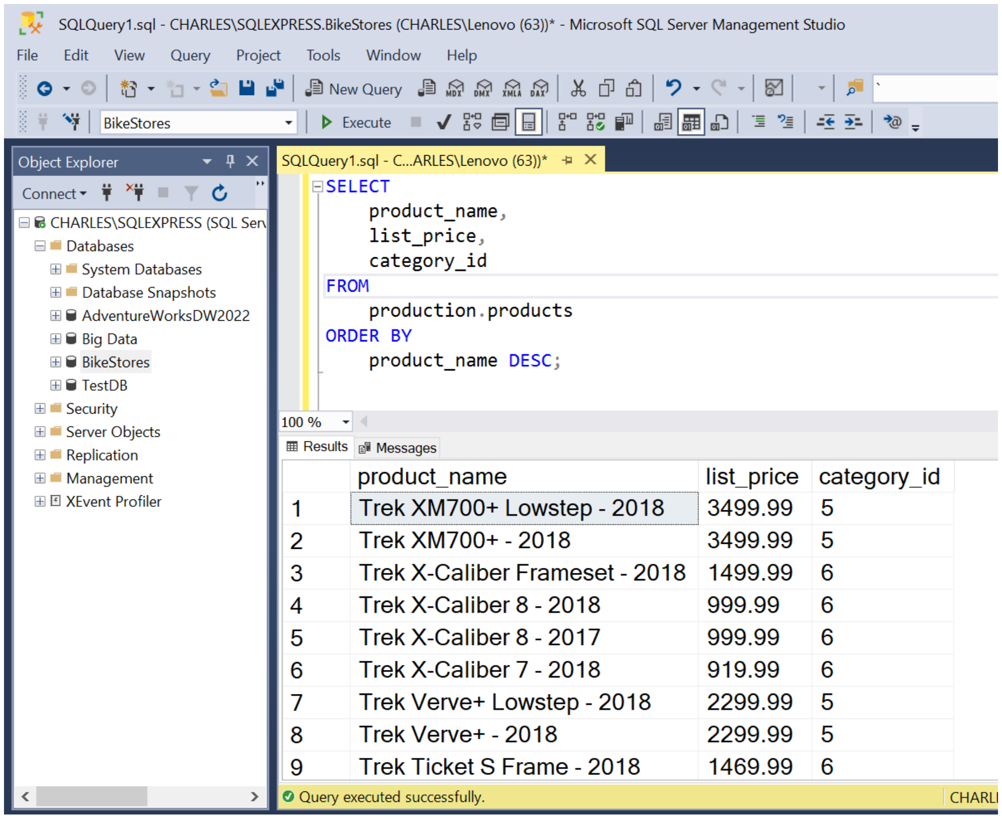
```
SELECT city FROM sales.customers INTERSECT
SELECT city FROM sales.stores ORDER BY city;
```

	city
1	Baldwin
2	Rowlett
3	Santa Cruz

JOIN OPERATION

INNER JOIN

```
SELECT
    product_name,
    list_price,
    category_id
FROM
    production.products
ORDER BY
    product_name DESC;
```



SQLQuery1.sql - CHARLES\SQLEXPRESS.BikeStores (CHARLES\Lenovo (63))* - Microsoft SQL Server Management Studio

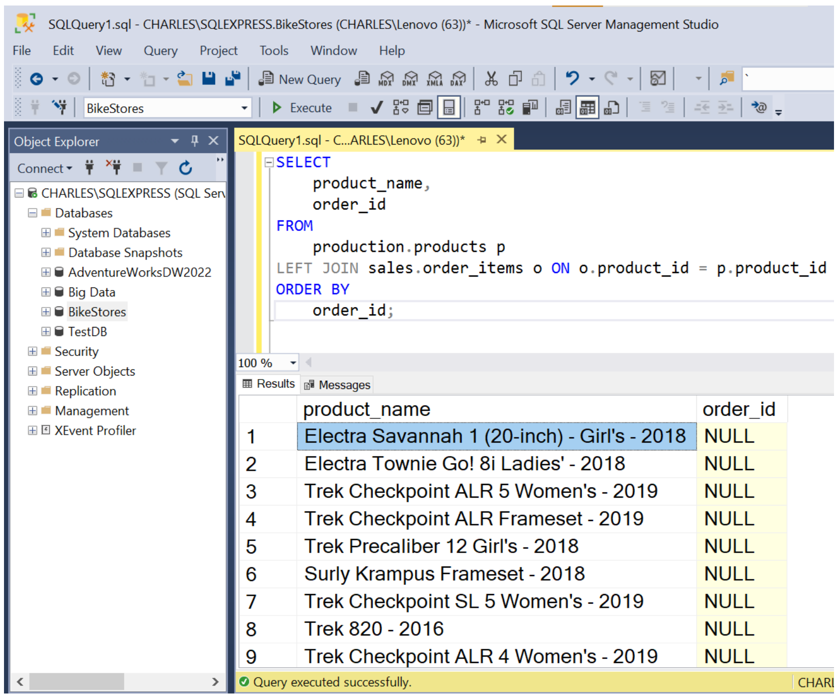
```
SELECT
    product_name,
    list_price,
    category_id
FROM
    production.products
ORDER BY
    product_name DESC;
```

```
   product_name                  list_price  category_id
1  Trek XM700+ Lowstep - 2018    3499.99     5
2  Trek XM700+ - 2018            3499.99     5
3  Trek X-Caliber Frameset - 2018 1499.99    6
4  Trek X-Caliber 8 - 2018       999.99      6
5  Trek X-Caliber 8 - 2017       999.99      6
6  Trek X-Caliber 7 - 2018       919.99      6
7  Trek Verve+ Lowstep - 2018    2299.99     5
8  Trek Verve+ - 2018            2299.99     5
9  Trek Ticket S Frame - 2018    1469.99     6
```

Query executed successfully.

LEFT JOIN

```
SELECT
    product_name,
    order_id
FROM
    production.products p
LEFT JOIN sales.order_items o ON o.product_id = p.product_id
ORDER BY
    order_id;
```



SQLQuery1.sql - CHARLES\SQLEXPRESS.BikeStores (CHARLES\Lenovo (63))* - Microsoft SQL Server Management Studio

```
SELECT
    product_name,
    order_id
FROM
    production.products p
LEFT JOIN sales.order_items o ON o.product_id = p.product_id
ORDER BY
    order_id;
```

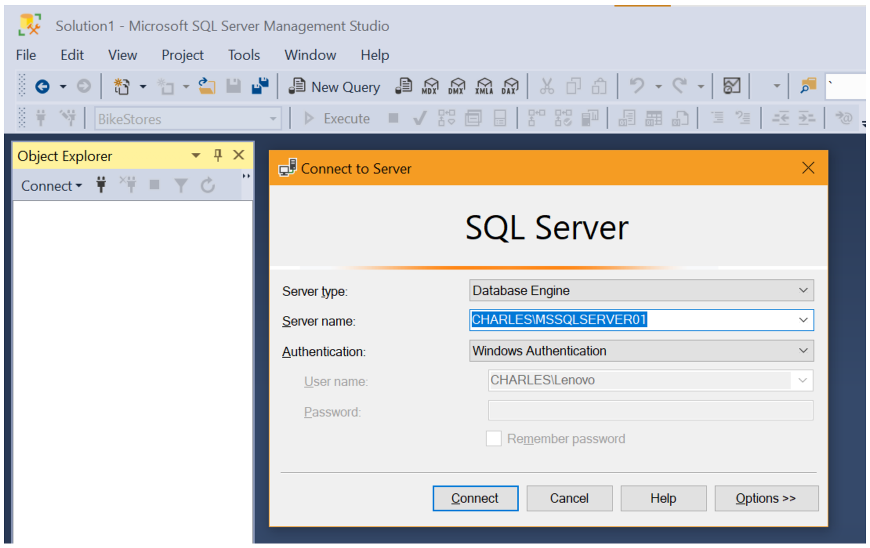
	product_name	order_id
1	Electra Savannah 1 (20-inch) - Girl's - 2018	NULL
2	Electra Townie Go! 8i Ladies' - 2018	NULL
3	Trek Checkpoint ALR 5 Women's - 2019	NULL
4	Trek Checkpoint ALR Frameset - 2019	NULL
5	Trek Precaliber 12 Girl's - 2018	NULL
6	Surly Krampus Frameset - 2018	NULL
7	Trek Checkpoint SL 5 Women's - 2019	NULL
8	Trek 820 - 2016	NULL
9	Trek Checkpoint ALR 4 Women's - 2019	NULL

Query executed successfully.

TASK 2b:

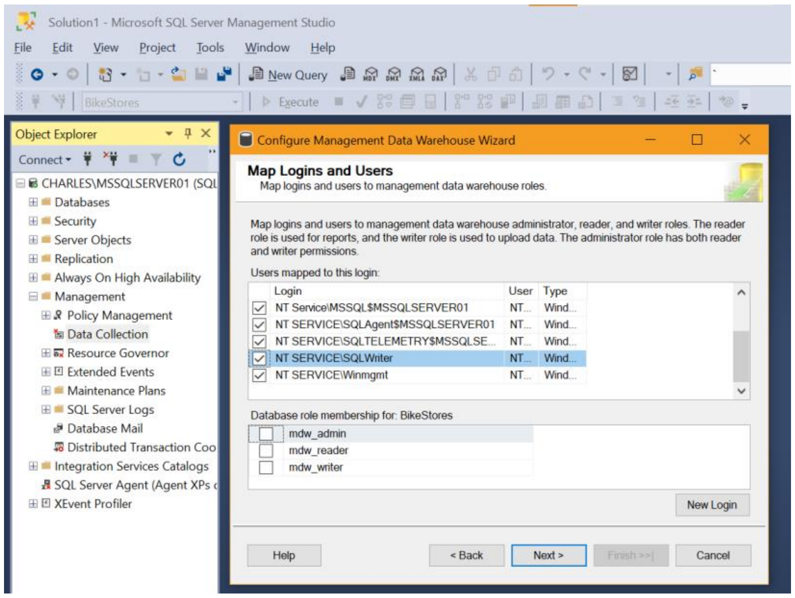
PROCEDURE:

1. Install Microsoft SQL server Management Studio
2. Install MS Sql Server Developer Edition
3. Connect to the SQL Server



Server type: Database Engine
Server name: CHARLES\MSSQLSERVER01
Authentication: Windows Authentication
User name: CHARLES\Lenovo
Password:
Remember password

4. Choose Management -> Data Collection in the left side navigation bar
5. Right click on the Data collection - > Tasks -> Configure Management Data Warehouse and Configure the data warehouse settings
6. Select Server Name and Database name and click next button
7. Change the settings in map login and users
   a. User mapped to this login
   b. Database membership for BikeStores


Solution1 - Microsoft SQL Server Management Studio
File Edit View Project Tools Window Help
New Query
BikeStores
Execute
Object Explorer
Connect
CHARLES\MSSQLSERVER01 (SQL
Databases
Security
Server Objects
Replication
Always On High Availability
Management
Policy Management
Data Collection
Resource Governor
Extended Events
Maintenance Plans
SQL Server Logs
Database Mail
Distributed Transaction Coo
Integration Services Catalogs
SQL Server Agent (Agent XPs
XEvent Profiler
Configure Management Data Warehouse Wizard
New Login
Help
< Back
Next >
Finish >>|
Cancel

Map Logins and Users

Map logins and users to management data warehouse roles.

Map logins and users to management data warehouse administrator, reader, and writer roles. The reader role is used for reports, and the writer role is used to upload data. The administrator role has both reader and writer permissions.

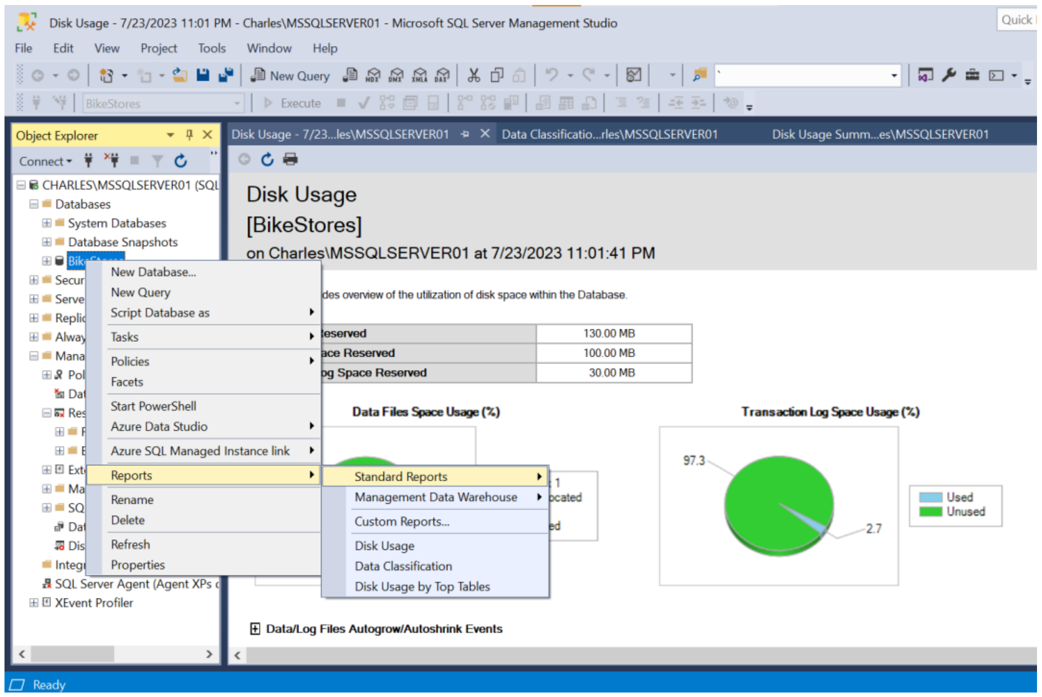
Users mapped to this login:

	Login	User	Type
✓	NT Service\MSSQL$MSSQLSERVER01	NT...	Wind...
✓	NT SERVICE\SQLAgent$MSSQLSERVER01	NT...	Wind...
✓	NT SERVICE\SQLTELEMETRY$MSSQLSE...	NT...	Wind...
✓	NT SERVICE\SQLWriter	NT...	Wind...
✓	NT SERVICE\Winmgmt	NT...	Wind...

Database role membership for: BikeStores

☐	mdw_admin
☐	mdw_reader
☐	mdw_writer

8. Check the Settings and Click finish button
9. Moniter and generate the various reports for the BikeStores, Right click the Bikestores-> Reports -> Standard Reports and select the reports needed.


Disk Usage - 7/23/2023 11:01 PM - Charles\MSSQLSERVER01 - Microsoft SQL Server Management Studio
File Edit View Project Tools Window Help
New Query
BikeStores
Execute
Object Explorer
Connect
CHARLES\MSSQLSERVER01 (SQL
Databases
System Databases
Database Snapshots
Disk Usage - 7/23...les\MSSQLSERVER01
Data Classificatio...rles\MSSQLSERVER01
Disk Usage Summ...es\MSSQLSERVER01
New Database...
New Query
Script Database as
Tasks
Policies
Facets
Start PowerShell
Azure Data Studio
Azure SQL Managed Instance link
Reports
Rename
Delete
Refresh
Properties
Standard Reports
Management Data Warehouse
Custom Reports...
Disk Usage
Data Classification
Disk Usage by Top Tables
Data Files Space Usage (%)
Transaction Log Space Usage (%)
97.3
2.7
Used
Unused
SQL Server Agent (Agent XPs
XEvent Profiler
Ready

Disk Usage
[BikeStores]
on Charles\MSSQLSERVER01 at 7/23/2023 11:01:41 PM

...des overview of the utilization of disk space within the Database.

...eserved	130.00 MB
...ace Reserved	100.00 MB
...og Space Reserved	30.00 MB

Data/Log Files Autogrow/Autoshrink Events

RESULT:

Date: **Task No: 3**

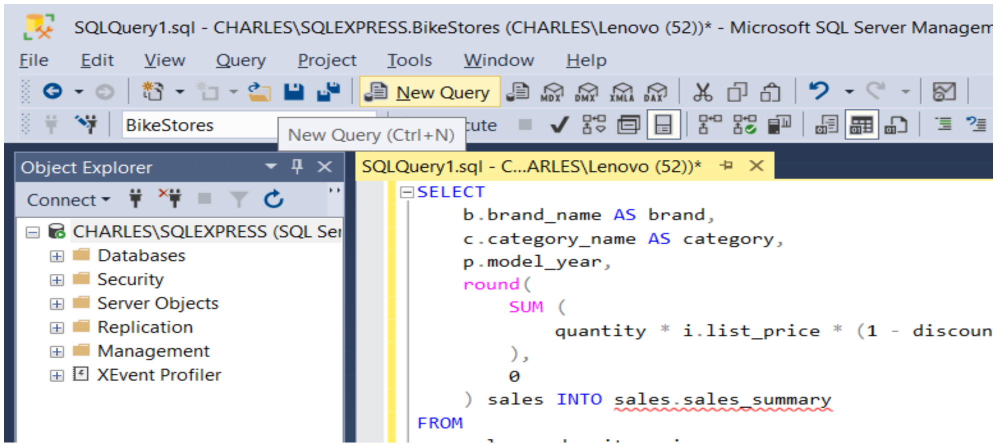
Perform Data Cube Operations (OLAP Operations) using SQL Queries Rollup, Rolldown, Slicing, Dicing
Database: MySQL

AIM:

PROCEDURE:

1) create the sales.sales_summary table by using the following query:

```
SELECT
    b.brand_name AS brand,
    c.category_name AS category,
    p.model_year,
    round(
        SUM (
            quantity * i.list_price * (1 - discount)
        ),
        0
    ) sales INTO sales.sales_summary
FROM
    sales.order_items i
INNER JOIN production.products p ON p.product_id = i.product_id
INNER JOIN production.brands b ON b.brand_id = p.brand_id
INNER JOIN production.categories c ON c.category_id = p.category_id
GROUP BY
    b.brand_name,
    c.category_name,
    p.model_year
ORDER BY
    b.brand_name,
    c.category_name,
    p.model_year;
```


SQLQuery1.sql - CHARLES\SQLEXPRESS.BikeStores (CHARLES\Lenovo (52))* - Microsoft SQL Server Managem
File Edit View Query Project Tools Window Help
New Query
BikeStores
New Query (Ctrl+N)
Object Explorer
Connect
CHARLES\SQLEXPRESS (SQL Se
Databases
Security
Server Objects
Replication
Management
XEvent Profiler
SQLQuery1.sql - C...ARLES\Lenovo (52))*

```
SELECT
    b.brand_name AS brand,
    c.category_name AS category,
    p.model_year,
    round(
        SUM (
            quantity * i.list_price * (1 - discoun
        ),
        0
    ) sales INTO sales.sales_summary
FROM
```

2) CUBE:

Refer task 2a:

The CUBE is a subclause of the GROUP BY clause that allows you to generate multiple grouping sets. The following illustrates the general syntax of the CUBE:

```
SELECT
   d1,
   d2,
   d3,
   aggregate_function (c4)
FROM
   table_name
GROUP BY
   d1,
   CUBE (d2, d3);
```

Example:

```
SELECT
   brand,
   category,
   SUM (sales) sales
FROM
   sales.sales_summary
GROUP BY
   CUBE(brand, category);
```

3) ROLLUP

Refer task 2a:

SQL Server ROLLUP syntax

The general syntax of the SQL Server ROLLUP is as follows:

```
SELECT
    d1,
    d2,
    d3,
    aggregate_function(c4)
FROM
    table_name
GROUP BY
    ROLLUP (d1, d2, d3);
```

Upload and configure the sample data warehouse (refer task2a)

Select the New Query, then new query tab will be opened.

ROLLUP Query

```
SELECT
    brand,
    category,
    SUM (sales) sales
FROM
    sales.sales_summary
GROUP BY
    ROLLUP(brand, category);
```

4) DRILLDOWN

This is a reverse of the ROLL UP operation. The data is aggregated from a higher level summary to a lower level summary/detailed data.

```
SELECT
    brand,
    category,
    SUM (sales) sales
FROM
    sales.sales_summary
GROUP BY
    ROLLUP(brand, category)
HAVING SUM (sales) > 20000;
```

5) SLICING:

A slice in a multidimensional array is a column of data corresponding to a single value for one or more members of the dimension. It helps the user to visualize and gather the information specific to a dimension.

```
SELECT
    *
FROM
    sales.customers
WHERE
    state = 'CA';
```

6) DICING:

Dicing is similar to slicing, but it works a little bit differently. Dicing, on the other hand, is more of a zoom feature that selects a subset over all the dimensions, but for specific values of the dimension.

```
SELECT
    city,
    COUNT (*)
FROM
    sales.customers
WHERE
    state = 'CA'
GROUP BY
    city
HAVING
    COUNT (*) > 10
ORDER BY
    city;
```

7) PIVOT

The steps to make a query a pivot table:

1) First, select a base dataset for pivoting.
2) Second, create a temporary result by using a derived table or common table expression (CTE)
3) Third, apply the PIVOT operator.

- First, select category name and product id from the production.products and production.categories tables as the base data for pivoting:

```
SELECT
    category_name,
    product_id
FROM
    production.products p
    INNER JOIN production.categories c
        ON c.category_id = p.category_id
```

- Second, create a temporary result set using a derived table:

```
SELECT * FROM (
    SELECT
        category_name,
        product_id
    FROM
        production.products p
        INNER JOIN production.categories c
            ON c.category_id = p.category_id
) t
```

- Third, apply the PIVOT operator:

```
SELECT * FROM
(
    SELECT
        category_name,
        product_id
    FROM
        production.products p
        INNER JOIN production.categories c
            ON c.category_id = p.category_id
) t
PIVOT(
    COUNT(product_id)
    FOR category_name IN (
        [Children Bicycles],
        [Comfort Bicycles],
        [Cruisers Bicycles],
        [Cyclocross Bicycles],
        [Electric Bikes],
        [Mountain Bikes],
        [Road Bikes])
) AS pivot_table;
```

RESULT:

Date: **Task No: 4**

Implement matrix multiplication with Map Reduce

AIM:

Task 4.1

VMWare AND Apache Hadoop

To Install VMWare AND Apache Hadoop, follow the steps

STEP 1. First of all, enter to the official site of VMware and download VMware Workstation https://www.vmware.com/tryvmware/?p=workstation-w

STEP 2. After downloading VMware workstation, install it on your PC

Step3: Installing Java 8 version.

Openjdk version "1.8.0_91"

OpenJDK Runtime Environment (build 1.8.0_91-8u91-b14-3ubuntu1~16.04.1-b14)

OpenJDK 64-Bit Server VM (build 25.91-b14, mixed mode)

This output verifies that OpenJDK has been successfully installed.

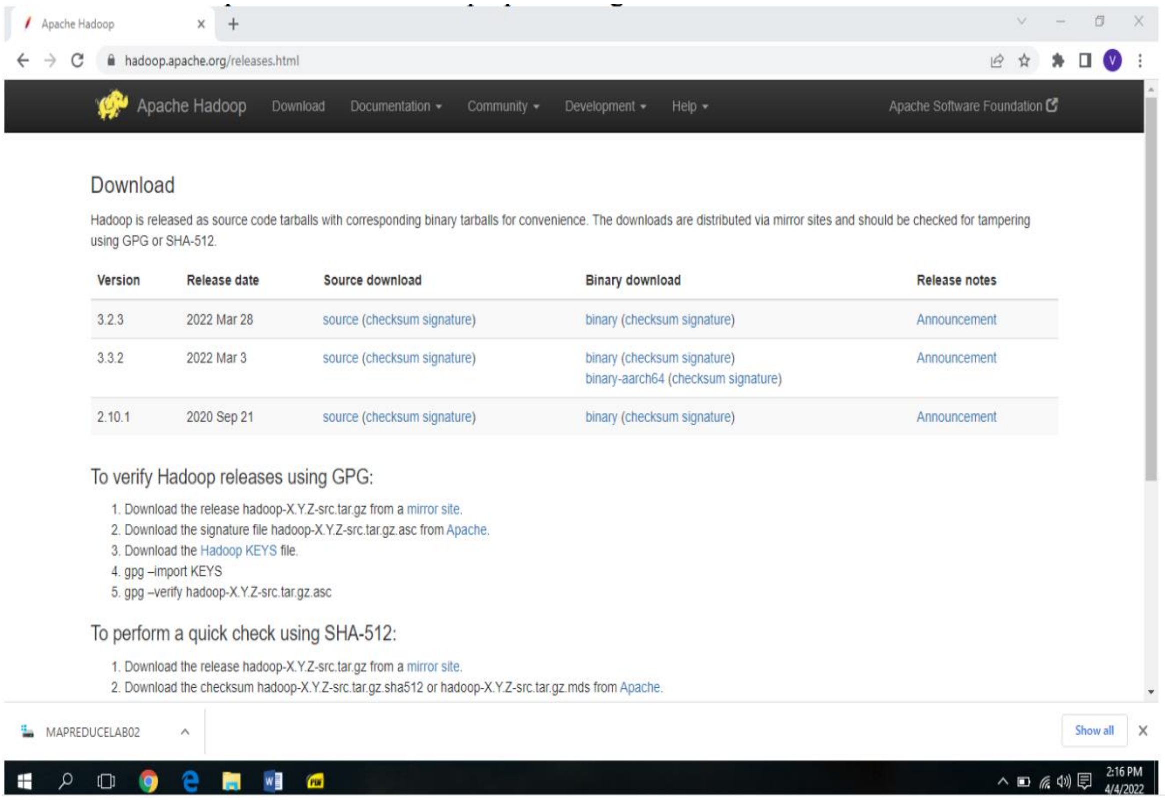
Note: To set the path for environment variables. i.e. JAVA_HOME

Step4: Installing Hadoop

With Java in place, we'll visit the Apache Hadoop Releases page to find the most

recent stable release. Follow the binary for the current release:

Download Hadoop from www.hadoop.apache.org



Download

Hadoop is released as source code tarballs with corresponding binary tarballs for convenience. The downloads are distributed via mirror sites and should be checked for tampering using GPG or SHA-512.

Version	Release date	Source download	Binary download	Release notes
3.2.3	2022 Mar 28	source (checksum signature)	binary (checksum signature)	Announcement
3.3.2	2022 Mar 3	source (checksum signature)	binary (checksum signature) binary-aarch64 (checksum signature)	Announcement
2.10.1	2020 Sep 21	source (checksum signature)	binary (checksum signature)	Announcement

To verify Hadoop releases using GPG:

1. Download the release hadoop-X.Y.Z-src.tar.gz from a mirror site.
2. Download the signature file hadoop-X.Y.Z-src.tar.gz.asc from Apache.
3. Download the Hadoop KEYS file.
4. gpg –import KEYS
5. gpg –verify hadoop-X.Y.Z-src.tar.gz.asc

To perform a quick check using SHA-512:

1. Download the release hadoop-X.Y.Z-src.tar.gz from a mirror site.
2. Download the checksum hadoop-X.Y.Z-src.tar.gz.sha512 or hadoop-X.Y.Z-src.tar.gz.mds from Apache.

Task 4.2

Develop a MapReduce program to implement Matrix Multiplication.

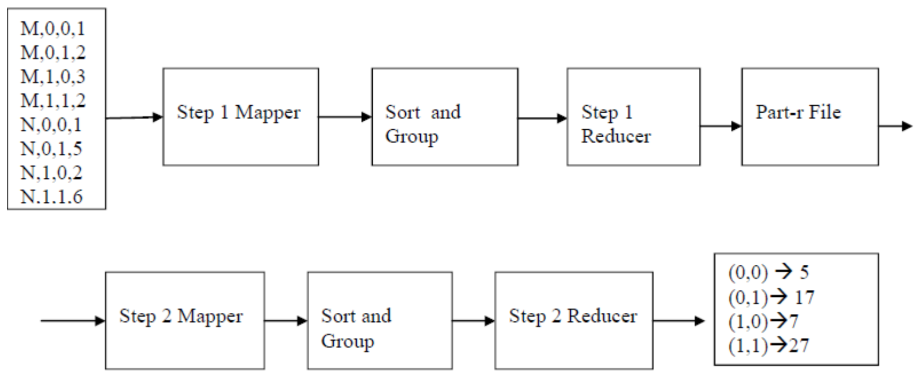
Assumes that you are using the Hadoop MapReduce framework and have two input matrices, Matrix A and Matrix B, stored in separate files.

Mapper Function: The mapper function takes as input a row from Matrix A and a column from Matrix B. It emits key-value pairs, where the key is the index of the resulting cell in the output matrix, and the value is the partial product of the corresponding elements from Matrix A and Matrix B.

Reducer Function:

The reducer function takes as input a key representing the index of a cell in the output matrix and a list of values representing the partial products for that cell. It calculates the final product by summing up the partial products and emits the result as the output.

Flow Diagram



The implementation should perform matrix multiplication using the MapReduce paradigm by distributing the computation across multiple mappers and reducers, resulting in efficient processing of large-scale matrices.

Task 4.3

Develop a MapReduce program to find the maximum temperature in each year.

A MapReduce program to find the maximum temperature in each year using the MapReduce paradigm. Assume that you are using the Hadoop MapReduce framework and have input data in the format: <Year> <Temperature>.

Mapper Function:

The mapper function takes as input a line from the input data file. It extracts the year and temperature from each line and emits key-value pairs, where the key is the year, and the value is the temperature.

Reducer Function:

The reducer function takes as input a year and a list of temperatures associated with that year. It iterates through the list of temperatures to find the maximum temperature and emits the result as the output.

RESULT:

Date:

Task No: 5

Write a Spark application to perform word count in the input file.

AIM:

5.1: Apache Spark Installation

PROCEDURE:

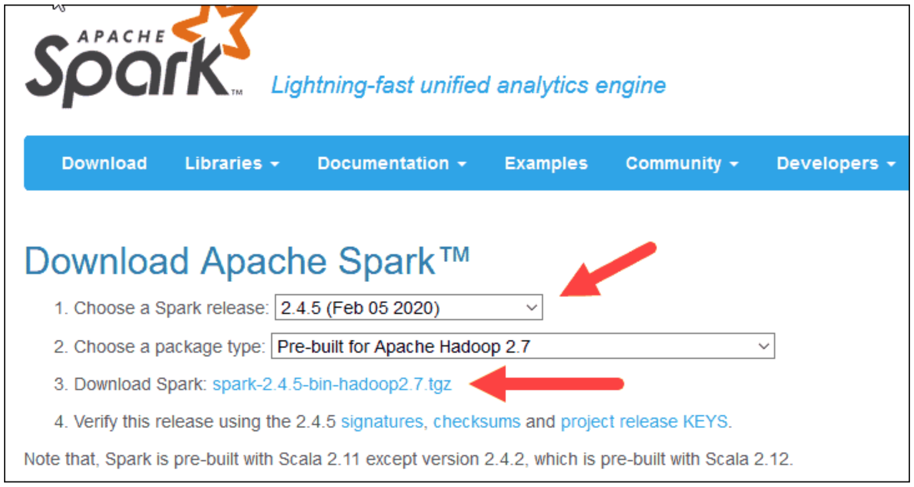
1. Download and install Java Development Kit (JDK) version 8 or higher, and ensure the RAM size, least 8 GB.
2. Download and install Python latest version from https://www.python.org/
3. Visit the Apache Spark website at https://spark.apache.org/downloads.html to download the latest stable release of Spark. [figure 1]
4. Type the following command in the command prompt to check the java and python version:
   - java --version
   - python --version
5. Create a new folder named Spark in the root of your C: drive and locate the Spark file you downloaded.
6. Right-click the file and extract it to C:\Spark using the tool you have on your system
7. Download the winutils.exe file for the underlying Hadoop version for the Spark installation, from the URL https://github.com/cdarlint/winutils
   [figure 2]
8. Configure Environment Variables
   - Click Start and type environment.
   - Select the result labeled Edit the system environment variables.
   - A System Properties dialog box appears. In the lower-right corner, click Environment Variables and then click New in the next window.
   - For Variable Name type SPARK_HOME. [figure 3,4]
   - For Variable Value type C:\Spark\spark-2.4.5-bin-hadoop2.7 and click OK. If you changed the folder path, use that one instead.
   - In the top box, click the Path entry, then click Edit. Be careful with editing the system path. Avoid deleting any entries already on the list.
   - The system highlights a new line. Enter the path to the Spark folder C:\Spark\spark-2.4.5-bin-hadoop2.7\bin. We recommend using %SPARK_HOME%\bin to avoid possible issues with the path. [figure 5,6,7]
   - Repeat this process for Hadoop and Java. For Hadoop, the variable name is HADOOP_HOME and for the value use the path of the folder you created earlier:

C:\hadoop. Add C:\hadoop\bin to the Path variable field, but we recommend using %HADOOP_HOME%\bin.
- For Java, the variable name is JAVA_HOME and for the value use the path to your Java JDK directory (in our case it's C:\Program Files\Java\jdk1.8.0_251).

9. Launch Spark, To start Spark, enter the command
   - C:\Spark\spark-2.4.5-bin-hadoop2.7\bin>spark-shell
   - If you set the environment path correctly, you can type spark-shell to launch Spark.
   - Finally, the Spark logo appears, and the prompt displays the Scala shell.
   - Open a web browser and navigate to http://localhost:4040/.

10. Launch pyspark, enter the command

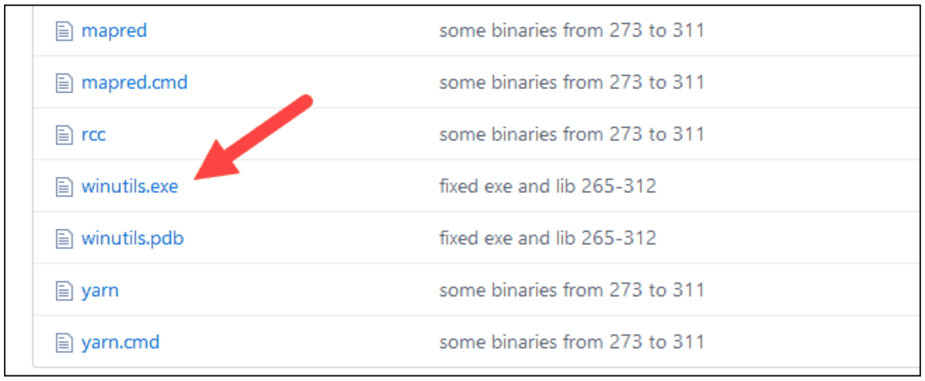
```
C:\Spark\spark-2.4.5-bin-hadoop2.7\bin>pyspark
```



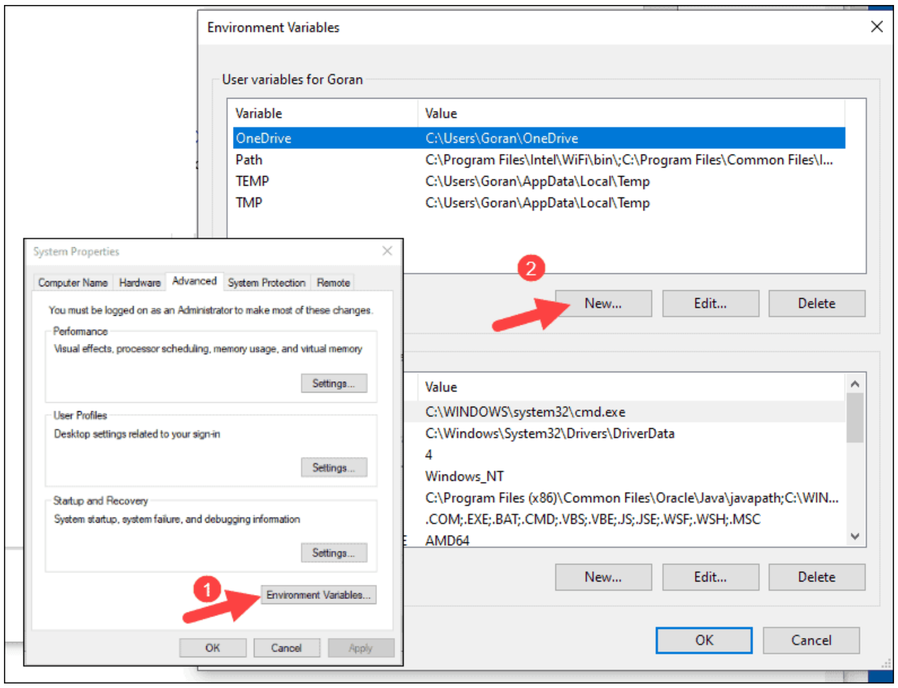
Download Apache Spark™

1. Choose a Spark release: 2.4.5 (Feb 05 2020)
2. Choose a package type: Pre-built for Apache Hadoop 2.7
3. Download Spark: spark-2.4.5-bin-hadoop2.7.tgz
4. Verify this release using the 2.4.5 signatures, checksums and project release KEYS.

Note that, Spark is pre-built with Scala 2.11 except version 2.4.2, which is pre-built with Scala 2.12.



mapred	some binaries from 273 to 311
mapred.cmd	some binaries from 273 to 311
rcc	some binaries from 273 to 311
winutils.exe	fixed exe and lib 265-312
winutils.pdb	fixed exe and lib 265-312
yarn	some binaries from 273 to 311
yarn.cmd	some binaries from 273 to 311



Variable	Value
OneDrive	C:\Users\Goran\OneDrive
Path	C:\Program Files\Intel\WiFi\bin\;C:\Program Files\Common Files\I...
TEMP	C:\Users\Goran\AppData\Local\Temp
TMP	C:\Users\Goran\AppData\Local\Temp

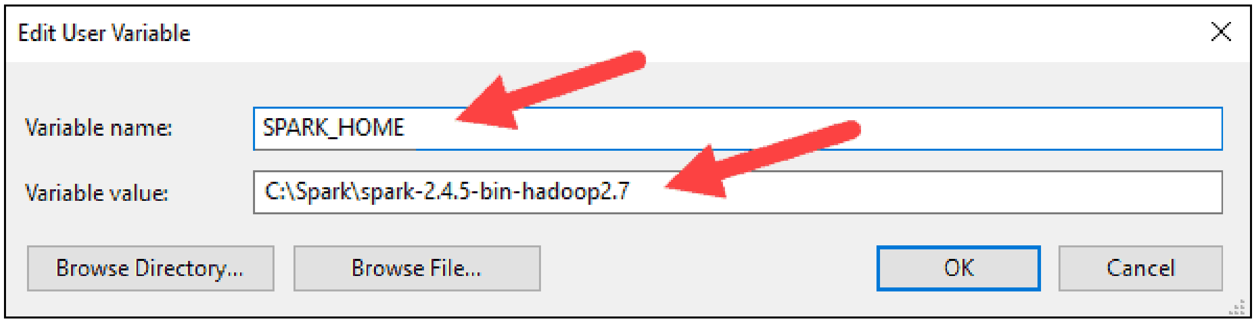
Value
C:\WINDOWS\system32\cmd.exe
C:\Windows\System32\Drivers\DriverData
4
Windows_NT
C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\WIN...
.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
AMD64

You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory

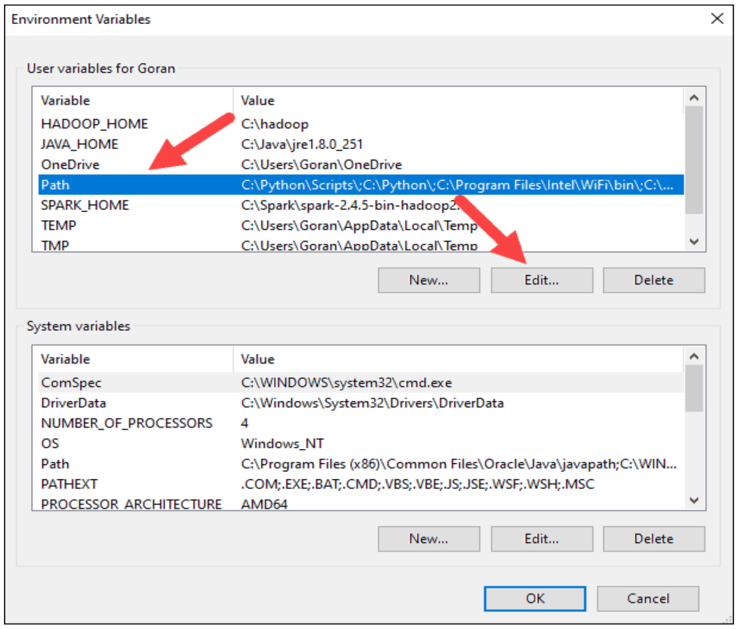
User Profiles
Desktop settings related to your sign-in

Startup and Recovery
System startup, system failure, and debugging information


Edit User Variable

Variable name: SPARK_HOME

Variable value: C:\Spark\spark-2.4.5-bin-hadoop2.7

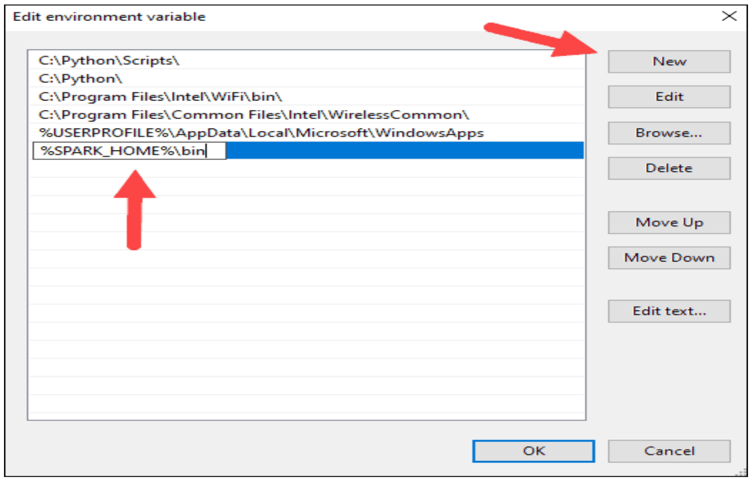

Environment Variables

User variables for Goran

Variable	Value
HADOOP_HOME	C:\hadoop
JAVA_HOME	C:\Java\jre1.8.0_251
OneDrive	C:\Users\Goran\OneDrive
Path	C:\Python\Scripts\;C:\Python\;C:\Program Files\Intel\WiFi\bin\;C:\...
SPARK_HOME	C:\Spark\spark-2.4.5-bin-hadoop2...
TEMP	C:\Users\Goran\AppData\Local\Temp
TMP	C:\Users\Goran\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	4
OS	Windows_NT
Path	C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\WIN...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64


Edit environment variable

```
C:\Python\Scripts\
C:\Python\
C:\Program Files\Intel\WiFi\bin\
C:\Program Files\Common Files\Intel\WirelessCommon\
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
%SPARK_HOME%\bin
```

5.2: Spark application to perform word count

PROCEDURE:

1. Check the spark installation, environmental variables setup
2. Import necessary libraries from pyspark
3. Create a SparkConf and SparkContext (or SparkSession):
4. Load the input data
5. Read the input file and Calculating words count
6. Perform the word count operation
7. Save the output, Stop the SparkContext or SparkSession
8. Stopping Spark-Session and Spark context

Program:

```
import findspark

findspark.init()

from pyspark.sql import SparkSession

spark = SparkSession.builder\

                .master("local")\

                .appName('Firstprogram')\

                .getOrCreate()

sc=spark.sparkContext

text_file = sc.textFile("firstprogram.txt")

counts = text_file.flatMap(lambda line: line.split(" ")) \

                      .map(lambda word: (word, 1)) \

                     .reduceByKey(lambda x, y: x + y)

output = counts.collect()

for (word, count) in output:

   print("%s: %i" % (word, count))

sc.stop()

spark.stop()
```

File Name: firstprogram.txt

Chennai formerly known as Madras, is the capital city of Tamil Nadu, the southernmost Indian state.

Output:

```
Chennai: 1
Formerly: 1
Known: 1
as: 1
Madras: 1
Is: 1
the: 2
Capital: 1
City: 1
Of: 1
Tamil: 1
Nadu: 1
southernmost: 1
Indian: 1
state.: 1
```

RESULT:

Date: **Task No: 6**

Implement CURD operations on Casandra

AIM:

PROCEDURE:

1. Install Java Development Kit (JDK) version 8 or higher.
2. Install Python stable version
3. Visit the Apache Cassandra website at https://cassandra.apache.org/download/ to download the latest stable release of Cassandra.
4. Set up the required, environment variables.
5. Modify the configuration files cassandra.yaml to adjust the settings according to your requirements.
6. Open the command prompt at bin folder and Start the Cassandra server using the command “cassandra”
7. Open the another command prompt at bin folder and start the command-line interface (CLI) using the command “cqls”
8. Create the keyspace and tables, then execute the CURD operations – create, update, read, delete.
9. Execute various no-sql queries and find the results.
10. Stop the cassandra server.

Create Keyspace:

A keyspace is like RDBMS database which contains column families, indexes, user defined types, data center awareness, strategy used in keyspace, replication factor, etc.

CREATE KEYSPACE <identifier> WITH <properties>

Or

Create keyspace KeyspaceName with replicaton={'class':strategy name,

'replication_factor': No of replications on different nodes}

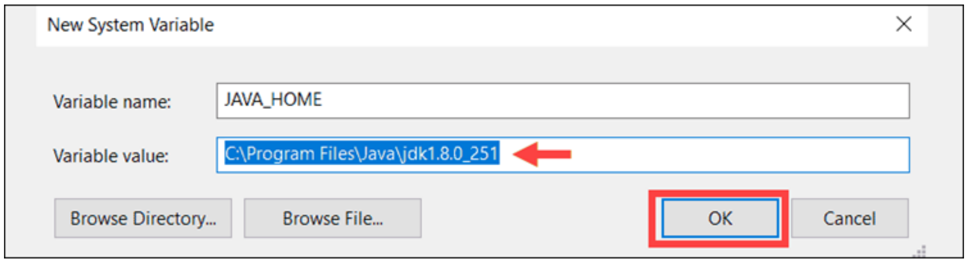
Alter Keyspace:

ALTER KEYSPACE <identifier> WITH <properties>

Or

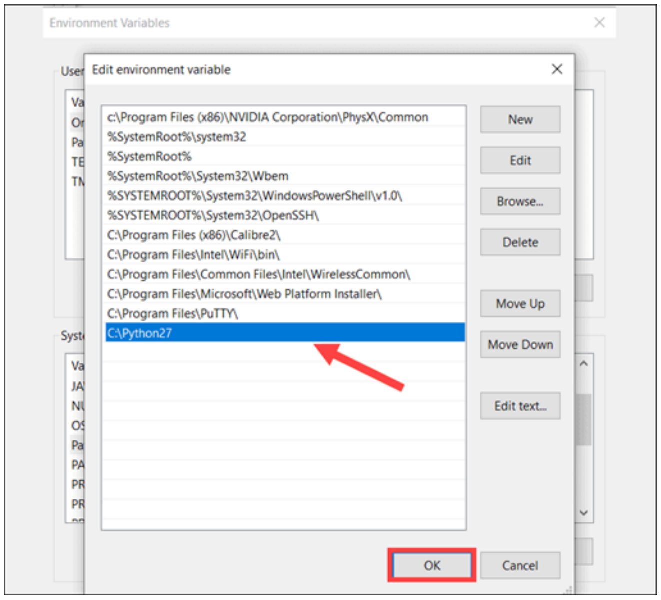
ALTER KEYSPACE "KeySpace Name"

WITH replication = {'class': 'Strategy name', 'replication_factor' : 'No.Of replicas'};


New System Variable

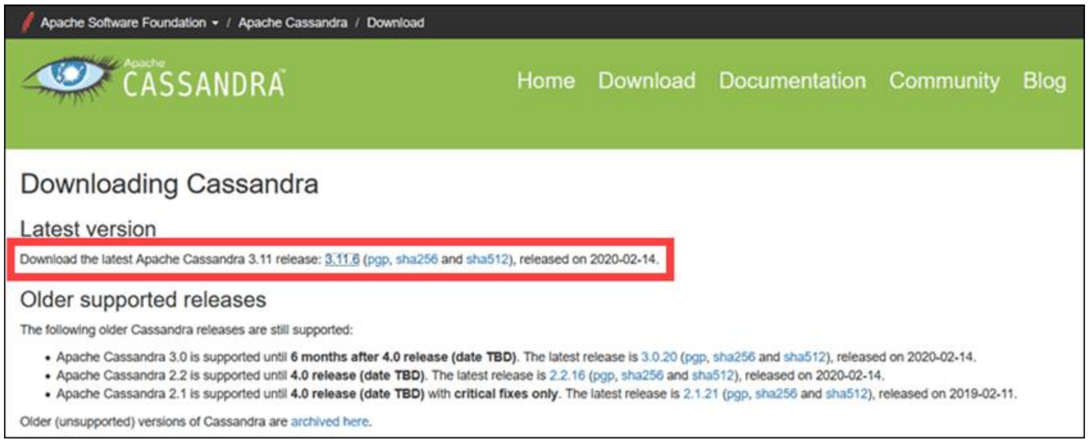
Variable name: JAVA_HOME

Variable value: C:\Program Files\Java\jdk1.8.0_251


Environment Variables

Edit environment variable

```
c:\Program Files (x86)\NVIDIA Corporation\PhysX\Common
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files (x86)\Calibre2\
C:\Program Files\Intel\WiFi\bin\
C:\Program Files\Common Files\Intel\WirelessCommon\
C:\Program Files\Microsoft\Web Platform Installer\
C:\Program Files\PuTTY\
C:\Python27
```



Downloading Cassandra

Latest version

Download the latest Apache Cassandra 3.11 release: 3.11.6 (pgp, sha256 and sha512), released on 2020-02-14.

Older supported releases

The following older Cassandra releases are still supported:

- Apache Cassandra 3.0 is supported until **6 months after 4.0 release (date TBD)**. The latest release is 3.0.20 (pgp, sha256 and sha512), released on 2020-02-14.
- Apache Cassandra 2.2 is supported until **4.0 release (date TBD)**. The latest release is 2.2.16 (pgp, sha256 and sha512), released on 2020-02-14.
- Apache Cassandra 2.1 is supported until **4.0 release (date TBD)** with **critical fixes only**. The latest release is 2.1.21 (pgp, sha256 and sha512), released on 2019-02-11.

Older (unsupported) versions of Cassandra are archived here.

Drop Keyspace:

```
DROP  keyspace KeyspaceName ;
```

Create Table Operation:

Syntax of Create Operation-

```
CREATE (TABLE | COLUMNFAMILY) <tablename>
('<column-definition>' , '<column-definition>')
(WITH <option> AND <option>)
```

Sample No-SQL queries

```
CREATE TABLE student(
   student_id int PRIMARY KEY,
   student_name text,
   student_city text,
   student_fees varint,
   student_phone varint
   );
```

CURD OPERATION:

Create Data:

```
INSERT INTO student (student_id, student_fees, student_name)
VALUES(1,5000, 'Ajeet');
INSERT INTO student (student_id, student_fees, student_name)
VALUES(2,3000, 'Kanchan');
INSERT INTO student (student_id, student_fees, student_name)
VALUES(3, 2000, 'Shivani');
```

READ Data

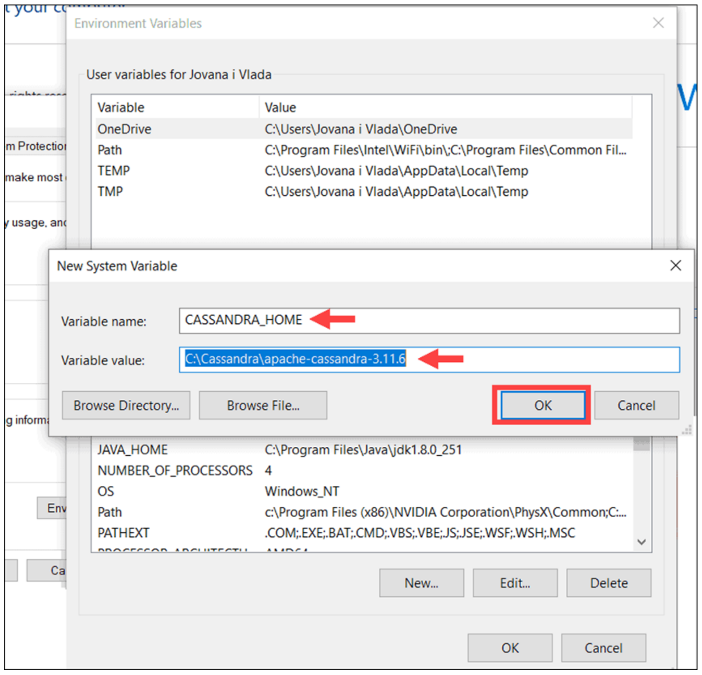
```
SELECT  * FROM student;
SELECT * FROM student WHERE student_id=2;
```

Update Data:

```
UPDATE student SET student_fees=10000,student_name='Rahul'
WHERE student_id=2;
```

DELETE Data:

```
DELETE student_fees FROM student WHERE student_id=4;
```

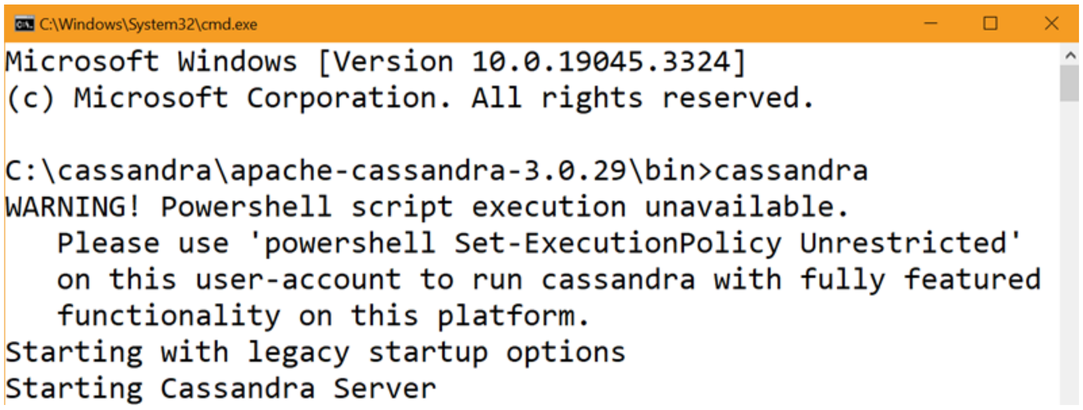


Variable	Value
OneDrive	C:\Users\Jovana i Vlada\OneDrive
Path	C:\Program Files\Intel\WiFi\bin\;C:\Program Files\Common Fil...
TEMP	C:\Users\Jovana i Vlada\AppData\Local\Temp
TMP	C:\Users\Jovana i Vlada\AppData\Local\Temp

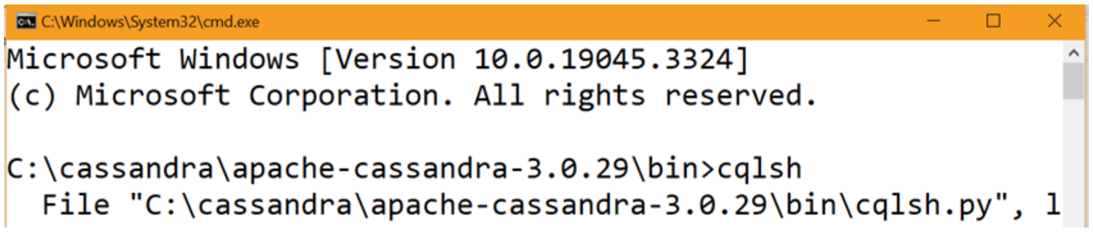
Variable name: CASSANDRA_HOME

Variable value: C:\Cassandra\apache-cassandra-3.11.6

JAVA_HOME	C:\Program Files\Java\jdk1.8.0_251
NUMBER_OF_PROCESSORS	4
OS	Windows_NT
Path	c:\Program Files (x86)\NVIDIA Corporation\PhysX\Common;C:...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC



```
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

C:\cassandra\apache-cassandra-3.0.29\bin>cassandra
WARNING! Powershell script execution unavailable.
   Please use 'powershell Set-ExecutionPolicy Unrestricted'
   on this user-account to run cassandra with fully featured
   functionality on this platform.
Starting with legacy startup options
Starting Cassandra Server
```



```
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

C:\cassandra\apache-cassandra-3.0.29\bin>cqlsh
  File "C:\cassandra\apache-cassandra-3.0.29\bin\cqlsh.py", l
```

RESULT:

Date: **Task No: 7**

Implementing Producer and Consumer problem in kafca.

AIM:

PROCEDURE:

1. Install Java Development Kit (JDK) version 8 or higher.
2. Install ZooKeeper: Kafka relies on ZooKeeper for distributed coordination. Kafka comes with an inbuilt Zookeeper
3. Visit the Apache Kafka website (https://kafka.apache.org/downloads) and download the latest version of Kafka for Windows.
4. Extract the contents of the Apache Kafka ZIP file to a required directory
5. Configure Environment Variables and path in System and User settings.
6. Start zookeeper by execute the zookeeper-server-start.bat script and pass the zookeeper configuration file path

   ```
   cd C:\bigdata\kafka2.5
   start cmd /k bin\windows\zookeeper-server-start.bat config\zookeeper.properties
   ```

7. Start Kafka by executing the kafka-server-start.bat script and pass broker configuration file path.

   ```
   cd C:\bigdata\kafka2.5
   start cmd /k bin\windows\kafka-server-start.bat config\server.properties
   ```

8. Open a new command prompt, and create new Kafka topic.

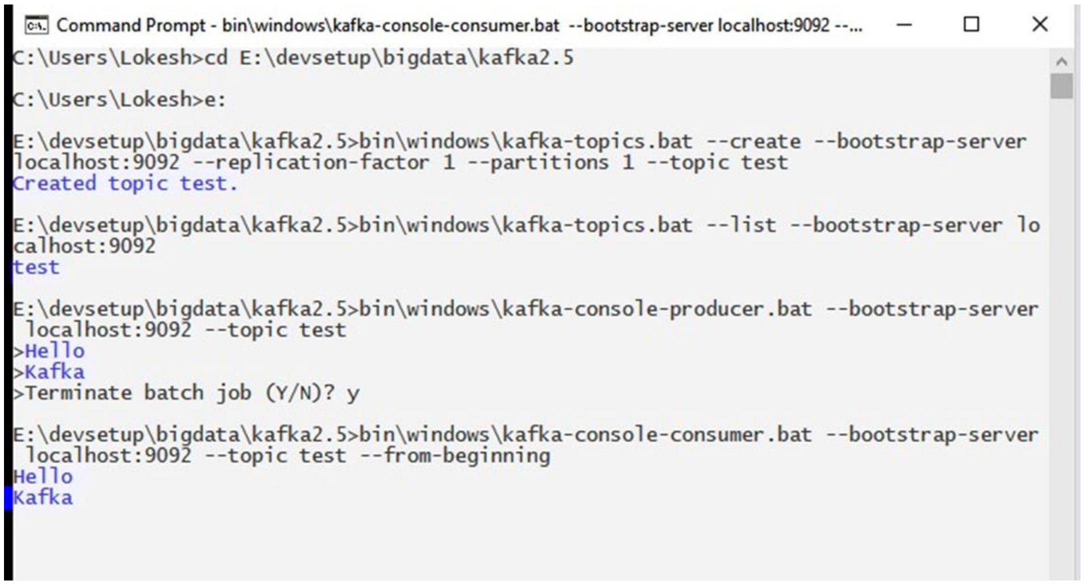
   ```
   bin\windows\kafka-topics.bat --create --bootstrap-server localhost:9092 --replication-factor 1 --
   partitions 1 --topic test
   Created topic test.
   ```

9. List all the topics to verify the created topic is present in this Topics list

   ```
   bin\windows\kafka-topics.bat --list --bootstrap-server localhost:9092
   test
   ```

10. Produce some messages and submit to test topic. I added two messages i.e. “Hello” and “Kafka”.

    ```
    bin\windows\kafka-console-producer.bat --bootstrap-server localhost:9092 --topic test
    ```



Command Prompt - bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --...

```
C:\Users\Lokesh>cd E:\devsetup\bigdata\kafka2.5

C:\Users\Lokesh>e:

E:\devsetup\bigdata\kafka2.5>bin\windows\kafka-topics.bat --create --bootstrap-server
localhost:9092 --replication-factor 1 --partitions 1 --topic test
Created topic test.

E:\devsetup\bigdata\kafka2.5>bin\windows\kafka-topics.bat --list --bootstrap-server lo
calhost:9092
test

E:\devsetup\bigdata\kafka2.5>bin\windows\kafka-console-producer.bat --bootstrap-server
 localhost:9092 --topic test
>Hello
>Kafka
>Terminate batch job (Y/N)? y

E:\devsetup\bigdata\kafka2.5>bin\windows\kafka-console-consumer.bat --bootstrap-server
 localhost:9092 --topic test --from-beginning
Hello
Kafka
```

11. Consume the messages and submit to test topic.

    ```
    bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test --from-beginning
    ```

    ```
    Hello

    Kafka
    ```

RESULT:

Date: **Task No: 8**

Implement basis commands in HIVE.

AIM:

8 A)

PROCEDURE:

1. Install Java Development Kit (JDK) version 8 or higher.
2. Install Hadoop-2.9.2, or use any other STA   BLE version for Hadoop.
3. Install MySQL Query Browser
4. Download and install Hive-3.1.2, or use any other STABLE version for Hive.
5. Set the environmental variables for HIVE_HOME and Path variables %HIVE_HOME%\bin
6. Open a NEW Command Window, check the path settings

   echo %HIVE_HOME%
7. Edit the Properties of hive-site.xml and Replace the value for <Your IP Address> with the IP Address of your System and replace <Your drive Folder> with the Hive folder Path.
8. Removing Special Characters and Adding few More Properties in hive-site.xml File.
9. Creating Hive User in MySQL Database for reading and writing data from it.
10. Grant permission to Users and Creating Metastore
11. Starting Hadoop, Hive Metastore and Hive shell prompt

    start-all.cmd

    hive --service metastore
12. Open a new cmd window and run the below command to start Hive

    > hive

Hive Basic Commands:

Show Databases:

SHOW DATABASES;

Use Database:

USE database_name;

Show Tables:

```
SHOW TABLES;
```

Create a Database:

```
CREATE DATABASE new_database;
```

Create a Table:

```
CREATE TABLE my_table (
    id INT,
    name STRING,
    age INT
);
```

Load Data into a Table:

```
LOAD DATA LOCAL INPATH '/path/to/datafile.csv' OVERWRITE INTO TABLE my_table;
```

Query Data:

```
SELECT * FROM my_table WHERE age > 25;
```

Insert Data into a Table:

```
INSERT INTO TABLE my_table VALUES (1, 'John', 30);
```

Describe Table:

```
DROP TABLE my_table;
```

8 B)

PROCEDURE:

1. Install Java Development Kit (JDK) version 8 or higher.
2. Install Hadoop-2.9.2, or use any other STABLE version for Hadoop.
3. Install MySQL Query Browser and configure the Hadoop, msql and Hive
4. Starting Hadoop, Hive Metastore and Hive shell prompt

   start-all.cmd

   hive --service metastore
5. Open a new cmd window and run the below command to start Hive

   > hive
6. Execute the various Hql Commands in Hive shell
7. Stop the hive and other server

HQL COMMANDS:

Create Database

hive> create database demo;

hive> create a database if not exists demo;

hive> show databases;

hive> describe database extended demo;

Create Table

hive> create table demo.employee (Id int, Name string , Salary float)

row format delimited

fields terminated by ',' ;

hive> create table if not exists demo.employee (Id int, Name string , Salary float)

row format delimited

fields terminated by ',' ;

Alter Table

hive> alter table old_table_name rename to new_table_name;

hive> Alter table emp rename to employee_data;

Drop Table

hive> drop table new_employee;

Aggregate Functions in Hive

employee_data

Id	Name	Salary
1	Gaurav	30000
2	Aryan	20000
3	Vishal	40000
4	John	10000
5	Henry	25000
6	William	9000
7	Lisa	25000
8	Ronit	20000

Load the below data file “empl_details” from local drive to HQL table “employee_data” and perform HQL function**.**

```
hive> create table employee_data (Id int, Name string , Salary float)
row format delimited
fields terminated by ',' ;
hive> load data local inpath '/home/codegyani/hive/emp_details' into table employee_data;
hive> select max(Salary) from employee_data;
```

GROUP BY and HAVING Clause

```
hive> select department, sum(salary) from emp group by department;
hive> select department, sum(salary) from emp group by department having sum(salary)>=35000;
```

View and Indexes

HQL Views:

Views are generated based on user requirements and save any result set data as a view.

```
hive> CREATE VIEW emp_30000 AS
SELECT * FROM employee
WHERE salary>30000;
```

Dropping a View

```
hive> DROP VIEW emp_30000;
```

Creating an Index:

An Index is nothing but a pointer on a particular column of a table. Creating an index means creating a pointer on a particular column of a table.

```
hive> CREATE INDEX inedx_salary ON TABLE employee(salary)
AS 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler';
```

Dropping an Index:

```
hive> DROP INDEX index_salary ON employee;
```

RESULTS:

Date: **Task No: 9**

Write Pig Latin scripts sort, group, join, project, and filter the data

AIM:

PROCEDURE:

1) Install the prerequisite Stable software's Java Development Kit and Java Runtime Environment, Apache Hadoop.
2) Visit the Apache Pig download page: https://pig.apache.org/downloads.html
3) Download the latest stable release of Pig.
4) Extract the Pig Archive and Set Environment Variables for java, Apache Hadoop and Apache Pig

   For Java:

   JAVA_HOME=C:\Program Files\Java\jdk-1.8

   Path = C:\Program Files\Java\jdk-1.8\bin

   For Hadoop:

   HADOOP_HOME=C:\ApacheHadoop2.9.2

   Path = C:\ApacheHadoop2.9.2\bin

   For Pig:

   PIG_HOME = C:\Apachepig

   Path = C:\Apachepig\bin

   Path = C:\Apachepig\conf

5) Verify the Paths, Run following commands in a NEW Command Window

   ```
   echo %PIG_HOME%
   ```

6) Open the pig. cmd file in edit mode, and change the value of the HADOOP_BIN_PATH

   Old value:- %HADOOP_HOME%\bin

   New Value:- %HADOOP_HOME%\libexec

7) Edit Pig Configuration, go to the conf directory within your Pig installation directory, rename the pig.properties.template file to pig.properties. and set the exectype property to "local"

   ```
   exectype=local
   ```

8) Start Apache Pig, run the following command in a new Command Prompt as administrator

   ```
   C:\Users\Lenovo>echo %PIG_HOME%
   ```

   O/P: C:\ApachePig

   ```
   C:\Users\Lenovo>pig -version
   ```

Output:

Pig Latin scripts:

Input.txt

```
Rajiv,42
siddarth,45
Rajesh,40
Preethi,23
Trupthi,34
Archana,21
Robin,22
BOB,23
Maya,23
Sara,25
David,23
Maggy,22
```

Addressfile.txt

```
Rajiv,Chennai
Rajesh,Delhi
Trupthi,Hyderabad
Robin,Pune
Maya,Hyderabad
Anderson,Chennai
Antolina,Chennai
```

```
Apache Pig version 0.17.0 (r1797386)
compiled Jun 02 2017, 15:41:58
C:\Users\Lenovo>pig
Grunt Shell started:
grunt>
```

Load Data:

```
data = LOAD './input.txt' USING PigStorage(',') AS (name:chararray, age:int);
```

Sort Operator:

```
sortbyage = ORDER data BY age ASC|DESC;
dump sortbyage;
```

Group Operator:

```
grouped_data = GROUP data BY age;
dump grouped_data;
```

Filter Operator:

```
filterbyage = FILTER data BY (age>40);
dump filterbyage;
```

Inner Join Operator:

```
table1 = LOAD './input.txt' USING PigStorage(',') AS (name:chararray, age:int);
table2 = LOAD './addressfile.txt' USING PigStorage(',') AS (name:chararray, address:chararray);

joinbyname = JOIN table1 BY name, table2 BY name;
```

OuterJoin Operator;

```
LO = JOIN table1 BY name LEFT OUTER, table2 BY name;
dump LO;

RO = JOIN table1 BY name RIGHT, table2 BY name;
dump RO;

FO = JOIN table1 BY name FULL OUTER, table2 BY name;
dump FO;
```

Store Operator:

```
grouped_data = GROUP data BY age;
result = FOREACH grouped_data GENERATE group AS age, COUNT(data) AS count;
STORE result INTO 'output';
```

RESULT:

.

Date: **Task No: 10**

i.) Construct the Pig Latin Scripts to find character Count

ii. Construct the Pig Latin Scripts to find a max temp for each and every year

AIM:

PROCEDURE:

Construct the Pig Latin Scripts to find character Count

1) Load the data from LOCAL file system
2) Convert the Sentence into words using TOKENIZE Function with delimeter
3) Convert Column into Rows
4) Apply GROUP BY and Generate word count
5) Dump the word count

Construct the Pig Latin Scripts to find a max temp for each and every year

1) Load your data (assuming it's in a tab-delimited file)
2) Extract the year from the date column (assuming date is in 'YYYY-MM-DD' format)
3) Group the data by year
4) Calculate the maximum temperature for each year
5) Store the result in an output file

Pig Latin Scripts:

(i) Construct the Pig Latin Scripts to find character Count

File Name: wordcount.pig

```
data = LOAD './countwords.txt' USING PigStorage() AS (line:chararray);

Words = FOREACH data GENERATE FLATTEN(TOKENIZE(line,' ')) AS word;

Grouped = GROUP Words BY word;

wordcount = FOREACH Grouped GENERATE group, COUNT(Words);

DUMP wordcount;
```

(ii) Construct the Pig Latin Scripts to find a max temp for each and every year

File Name: tempanalysis.pig

OUTPUT:

```
grunt>exec wordcount.pig
(a,2)
(is,2)
(This,1)
(post,1)
(hadoop,2)
(bigdata,1)
(technology,1)

grunt>exec tempanalysis.pig
(2020,38.5)
(2021,36.7)
(2022,40.8)
(2023,42.8)
```

```
raw_data = LOAD './temperature.txt' USING PigStorage('\t') AS (date:chararray, temperature:double);

data_with_year = FOREACH raw_data GENERATE SUBSTRING(date, 0, 4) AS year, temperature;

grouped_data = GROUP data_with_year BY year;

max_temp_per_year = FOREACH grouped_data GENERATE group AS year,
MAX(data_with_year.temperature) AS max_temperature;

STORE max_temp_per_year INTO 'your_output_file.txt' USING PigStorage('\t');
```

Input Files:

countwords.txt

```
This is a hadoop post
hadoop is a bigdata technology
```

temperature.txt

```
9/12/2020	33.4
7/13/2021	30.45
9/14/2020	28.6
8/15/2023	30.4
6/16/2022	33.5
9/17/2023	38.6
9/18/2020	38.5
5/19/2023	36.4
9/20/2023	35.7
9/21/2021	36.7
2/22/2023	39.4
9/23/2023	40.2
3/24/2022	40.6
1/25/2022	40.8
9/26/2023	42.8
```

RESULT:

Date: **Task No: 11**

A . Collect Social Media Data from a Twitter to a Local File with the Topic ‘covid 19’

AIM:

PROCEDURE:

1. Install tweepy and json module in pycharm
2. Import tweepy and json module.
3. Create an account in Twitter Developer mode.
4. Get the Twitter API credentials – consumer key, consumer secret, access token, access token secret.
5. Authenticate with Twitter API
6. Search query and set, number of tweets to collect
7. Create File to save the collected data with file name 'covid19_tweets.json'
8. Collect tweets and save to a file

PROGRAM:

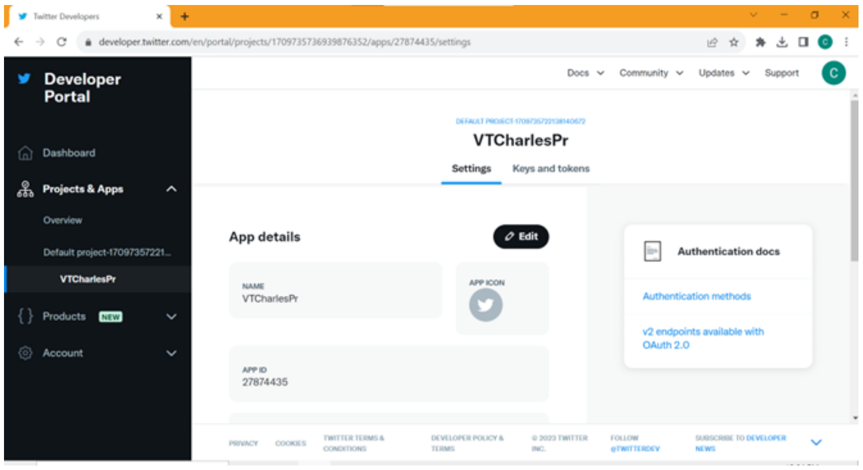
```
import tweepy
import json

consumer_key = 'KbKxl06BkeqAcwNhJnBMN5s5M'
consumer_secret = 'LaVFLQhzzy57mVDo9dbT60QOXnN6Ajm09Q6PAmSgPGoozZqhM9'
access_token = '1709726896194244609-w4KYxcGTyA4GHjlO7kJHzQlCWiVvSm'
access_token_secret = 'G8uyaHaIXoDUmFzVW3Ao6sAWRZ328Y2B6RraAeEU8p4e3'

auth = tweepy.OAuth2BearerHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth, wait_on_rate_limit=True)
search_query = 'covid 19'
num_tweets = 1000
output_file = 'covid19_tweets.json'
tweets = []
for tweet in tweepy.Cursor(api.search, q=search_query, lang='en').items(num_tweets):
    tweets.append(tweet._json)

with open(output_file, 'w', encoding='utf-8') as file:
    json.dump(tweets, file, ensure_ascii=False, indent=4)

print(f'{len(tweets)} tweets collected and saved to {output_file}')
```

OUTPUT:

'covid19_tweets.json'

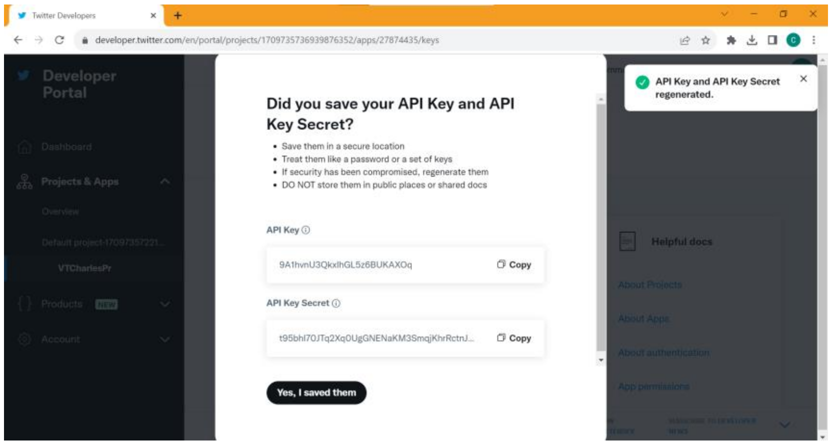
```
[{
   "State": "Andaman and Nicobar Islands",
   "Confirmed": 7651,
   "Recovered": 7518,
   "Deaths": 129,
   "Active": 4,
   "Last_Updated_Time": "13/08/2021 23:27:22",
   "Migrated_Other": 0,
   "State_code": "AN",
   "Delta_Confirmed": 0,
   "Delta_Recovered": 0,
   "Delta_Deaths": 0,
   "State_Notes": ""
 },
 {
   "State": "Andhra Pradesh",
   "Confirmed": 2066450,
   "Recovered": 2047722,
   "Deaths": 14373,
   "Active": 4355,
   "Last_Updated_Time": "13/08/2021 23:27:22",
   "Migrated_Other": 0,
   "State_code": "AP",
   "Delta_Confirmed": 0,
   "Delta_Recovered": 0,
   "Delta_Deaths": 0,
   "State_Notes": ""
 }]
```

.



NAME
VTCharlesPr

APP ID
27874435


developer.twitter.com/en/portal/projects/1709735736939876352/apps/27874435/keys

API Key and API Key Secret regenerated.

Did you save your API Key and API Key Secret?

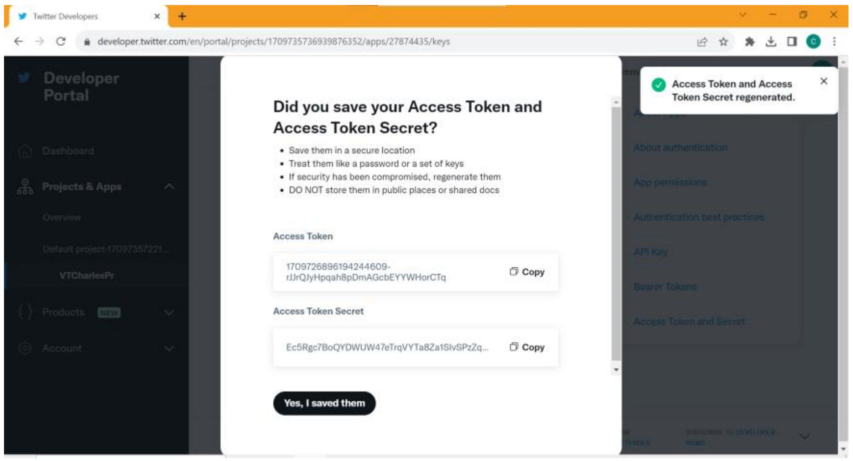
- Save them in a secure location
- Treat them like a password or a set of keys
- If security has been compromised, regenerate them
- DO NOT store them in public places or shared docs

API Key

9A1hvnU3QkxlhGL5z6BUKAXOq

API Key Secret

t95bhl70JTq2XqOUgGNENaKM3SmqjKhrRctnJ...


developer.twitter.com/en/portal/projects/1709735736939876352/apps/27874435/keys

Access Token and Access Token Secret regenerated.

Did you save your Access Token and Access Token Secret?

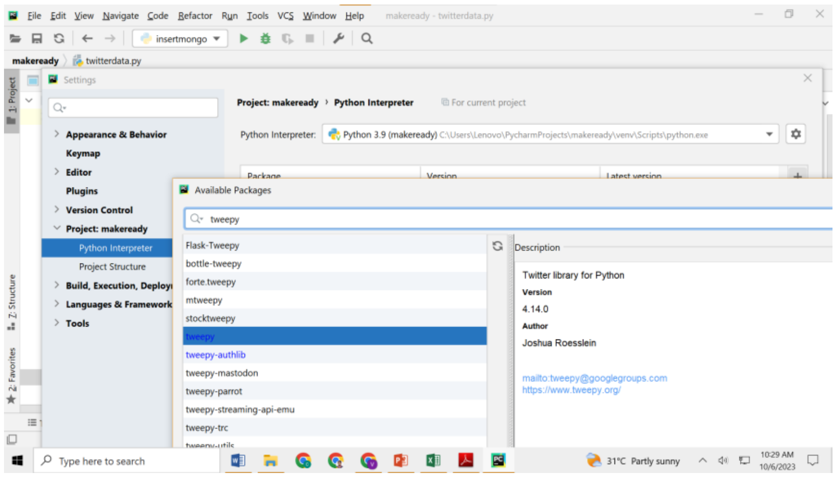
- Save them in a secure location
- Treat them like a password or a set of keys
- If security has been compromised, regenerate them
- DO NOT store them in public places or shared docs

Access Token

1709726896194244609-rJJrQJyHpqah8pDmAGcbEYYWHorCTq

Access Token Secret

Ec5Rgc7BoQYDWUW47eTrqVYTa8Za1SlvSPzZq...


makeready - twitterdata.py

Project: makeready › Python Interpreter

Python Interpreter: Python 3.9 (makeready) C:\Users\Lenovo\PycharmProjects\makeready\venv\Scripts\python.exe

Available Packages

tweepy

Flask-Tweepy
bottle-tweepy
forte.tweepy
mtweepy
stocktweepy
tweepy
tweepy-authlib
tweepy-mastodon
tweepy-parrot
tweepy-streaming-api-emu
tweepy-trc
tweepy-utils

Description

Twitter library for Python

Version

4.14.0

Author

Joshua Roesslein

mailto:tweepy@googlegroups.com
https://www.tweepy.org/

B. Download and Set Up MongoDB Server and a Client Mongodb Compass.

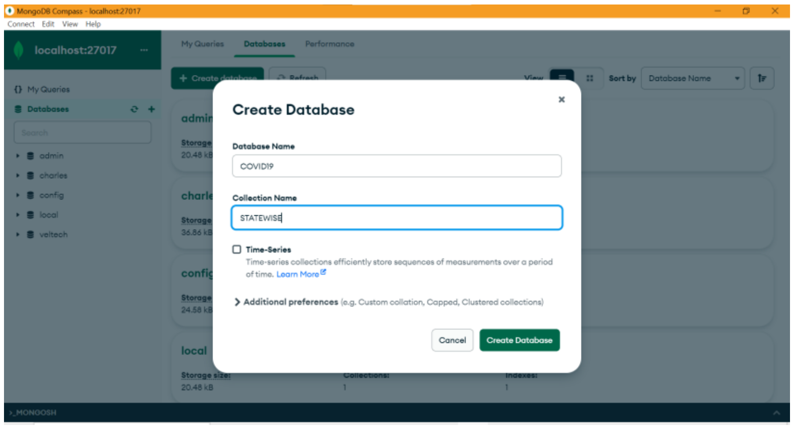
PROCEDURE:

1. Installing MongoDB on Windows:
2. Go to the official MongoDB website (https://www.mongodb.com/try/download/community) and download the MongoDB Community Server for Windows.
3. Run the downloaded installer (.msi file).
4. Choose a custom installation if you want to specify a custom installation directory or options.
5. After installation, MongoDB should start automatically as a Windows service. Check this in the Windows Services Manager.
6. Open Mongodb Compass, By default, MongoDB listens on port 27017.
7. Import the COIVD19, state-wise data into the Mongodb

<u>SAMPLE INPUT FILE:</u> [File Name: covid19statewise.json]

```
[{
   "State": "Andaman and Nicobar Islands",
   "Confirmed": 7651,
   "Recovered": 7518,
   "Deaths": 129,
   "Active": 4,
   "Last_Updated_Time": "13/08/2021 23:27:22",
   "Migrated_Other": 0,
   "State_code": "AN",
   "Delta_Confirmed": 0,
   "Delta_Recovered": 0,
   "Delta_Deaths": 0,
   "State_Notes": ""
  },
  {
   "State": "Andhra Pradesh",
   "Confirmed": 2066450,
   "Recovered": 2047722,
   "Deaths": 14373,
   "Active": 4355,
   "Last_Updated_Time": "13/08/2021 23:27:22",
   "Migrated_Other": 0,
   "State_code": "AP",
   "Delta_Confirmed": 0,
   "Delta_Recovered": 0,
   "Delta_Deaths": 0,
   "State_Notes": ""
  }]
.
```

OUTPUT:
Import Completed
38 Documents Imported



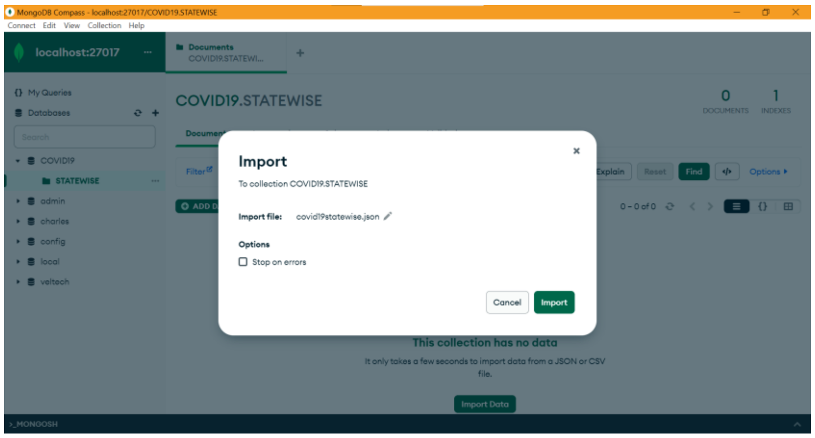
Create Database

Database Name: COVID19

Collection Name: STATEWISE

Time-Series
Time-series collections efficiently store sequences of measurements over a period of time. Learn More

Additional preferences (e.g. Custom collation, Capped, Clustered collections)



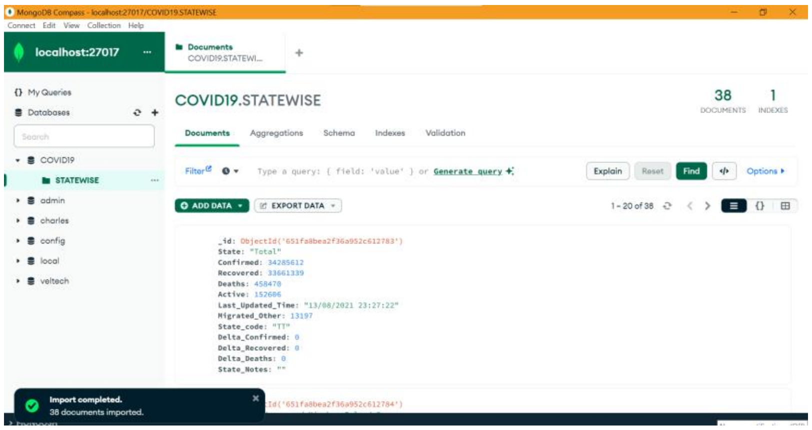
Import

To collection COVID19.STATEWISE

Import file: covid19statewise.json

Options
Stop on errors

This collection has no data
It only takes a few seconds to import data from a JSON or CSV file.



```
_id: ObjectId('651fa8bea2f36a952c612783')
State: "Total"
Confirmed: 34285612
Recovered: 33661339
Deaths: 458470
Active: 152686
Last_Updated_Time: "13/08/2021 23:27:22"
Migrated_Other: 13197
State_code: "TT"
Delta_Confirmed: 0
Delta_Recovered: 0
Delta_Deaths: 0
State_Notes: ""
```

Import completed.
38 documents imported.

RESULT:

Date: **Task No: 12**

A. Retrieve Analytic Information given from MongoDB

i. For each “place_type”, Find total favorite_count

ii. For each "country_code", find total "retweet_count"

AIM:

.

PROCEDURE:

1. Install pymongo in Pycharm IDE
2. Import the pymongo module in the analytics program
3. Configure the mongodb server using mongodb compass
4. Upload the country dataset in to Mongodb
5. Get the MongoDB URI, Database name and Collection Name
6. Establish the MongoDB connection using MongoDB URI, Database name and Collection Name
7. Aggregation pipeline to calculate the total favorite_count, retweet_count
8. Generate the results using aggregate functions.
9. Print the results, total favorite_count, retweet_count

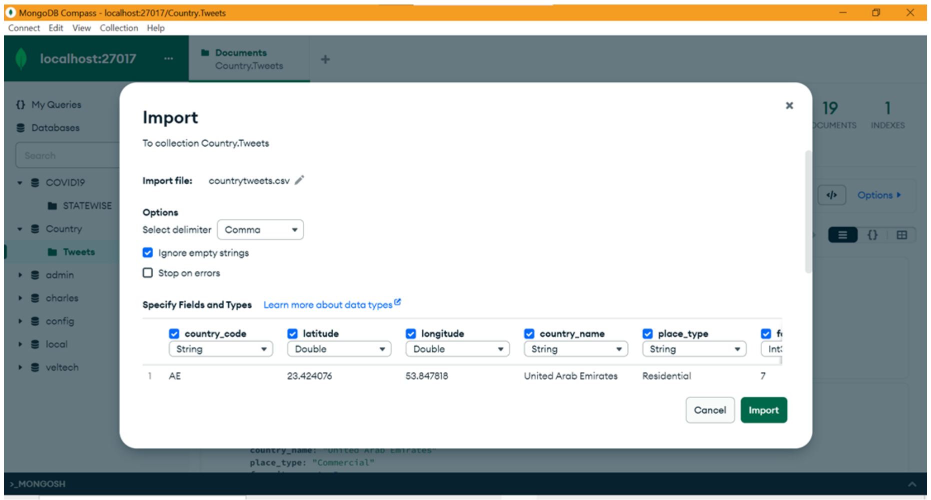
PROGRAM:

```
import pymongo
mongo_uri = "mongodb://localhost:27017/"
database_name = "Country"
collection_name = "Tweets"
client = pymongo.MongoClient(mongo_uri)
db = client[database_name]
collection = db[collection_name]

pipeline = [
    {
        "$group": {
            "_id": "$country_code",
            "total_favorite_count": {"$sum": "$favorite_count"},
            "total_retweet_count": {"$sum": "$retweet_count"}
        }
    }
]

result = list(collection.aggregate(pipeline))

if result:
    total_favorite_count = result[0]["total_favorite_count"]
```

OUTPUT:

```
Total favorite_count: 3

Total_retweet_count: 6
```

```
else:
    print("No tweets found in the collection.")

# Close the MongoDB connection
client.close()
```



MongoDB Compass - localhost:27017/Country.Tweets

Connect Edit View Collection Help

localhost:27017 — Documents Country.Tweets

Import

To collection Country.Tweets

Import file: countrytweets.csv

Options

Select delimiter: Comma

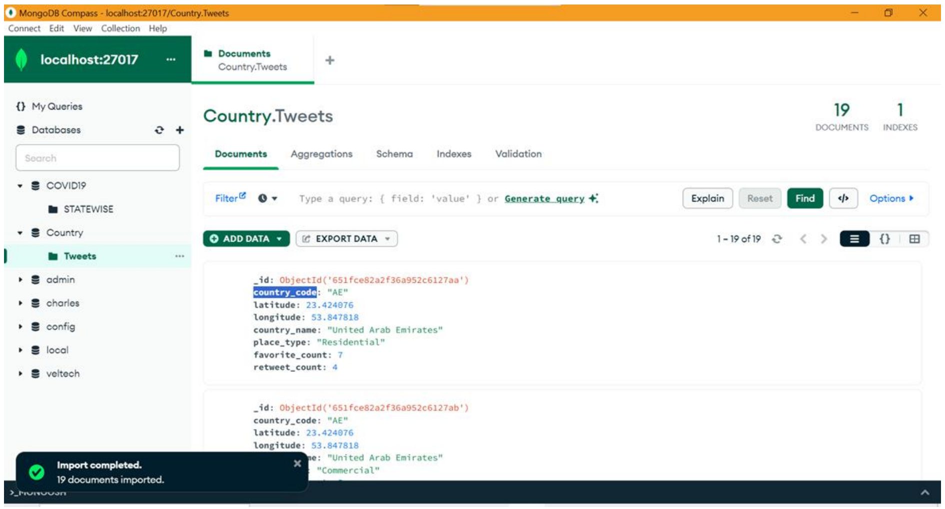
[✓] Ignore empty strings

[ ] Stop on errors

Specify Fields and Types Learn more about data types

	[✓] country_code	[✓] latitude	[✓] longitude	[✓] country_name	[✓] place_type	[✓] f
	String	Double	Double	String	String	Int
1	AE	23.424076	53.847818	United Arab Emirates	Residential	7

Cancel Import



MongoDB Compass - localhost:27017/Country.Tweets

Connect Edit View Collection Help

Country.Tweets — 19 DOCUMENTS, 1 INDEXES

Documents Aggregations Schema Indexes Validation

1 – 19 of 19

```
_id: ObjectId('651fce82a2f36a952c6127aa')
country_code: "AE"
latitude: 23.424076
longitude: 53.847818
country_name: "United Arab Emirates"
place_type: "Residential"
favorite_count: 7
retweet_count: 4

_id: ObjectId('651fce82a2f36a952c6127ab')
country_code: "AE"
latitude: 23.424076
longitude: 53.847818
```

Import completed.
19 documents imported.

```
total_retweet_count = result[1]["total_retweet_count"]
print(f"Total favorite_count: {total_favorite_count}")
print(f"Total_retweet_count: {total_retweet_count}")
```

SAMPLE DATASET:

country_code	**latitude**	**longitude**	**country_name**	**place_type**	**favorite_count**	**retweet_count**
AE	23.424076	53.847818	United Arab Emirates	Residential	7	4
AE	23.424076	53.847818	United Arab Emirates	Commercial	5	2
AR	-38.416097	-63.616672	Argentina	Educational	4	6
AR	-38.416097	-63.616672	Argentina	Cultural	5	5
IN	20.593684	78.96288	India	Religious	80	3
IN	20.593684	78.96288	India	Historical	100	6
TH	15.870032	100.992541	Thailand	Shopping	2	5
IN	20.593684	78.96288	India	Sports	55	3
AO	-11.202692	17.873887	Angola	Residential	2	6
AQ	-75.250973	-0.071389	Antarctica	Commercial	4	4
AR	-38.416097	-63.616672	Argentina	Educational	5	9
AS	-14.270972	-170.132217	American Samoa	Cultural	3	6
BD	23.684994	90.356331	Bangladesh	Religious	0	4
AU	-25.274398	133.775136	Australia	Historical	8	4
AW	12.52111	-69.968338	Aruba	Shopping	2	8
AZ	40.143105	47.576927	Azerbaijan	Sports	4	5
EG	26.820553	30.802498	Egypt	Tourist	2	2
IN	20.593684	78.96288	India	Tourist	8	5
BD	23.684994	90.356331	Bangladesh	Religious	9	2

RESULT:

B. Find out top 10 most frequent topic words of the entire tweet message texts of your collection after lemmatization/stemming and removing all the Stop Words.

AIM:

PROCEDURE:

1. Install NLTK data in the PyCharm IDE
2. Import the NLTK libraries WordNetLemmatizer, PorterStemmer, stopwords
3. Read the sample tweet message from Mongodb or assign tweet = ‘sample message’
4. Tokenize the tweet into words
5. Initialize lemmatizer and stemmer
6. Lemmatize and stem each word
7. Remove stop words
8. Define the list of English stop words
9. Join the filtered words back into a sentence
10. Print the results

PROGRAM:

```
import nltk
from nltk.stem import WordNetLemmatizer
from nltk.stem import PorterStemmer
from nltk.corpus import stopwords

nltk.download('punkt')
nltk.download('wordnet')
nltk.download('stopwords')

tweet = "The quick brown foxes are jumping over the lazy dogs' bones."

words = nltk.word_tokenize(tweet)
lemmatizer = WordNetLemmatizer()
stemmer = PorterStemmer()
lemmatized_words = [lemmatizer.lemmatize(word) for word in words]
stemmed_words = [stemmer.stem(word) for word in words]
words = nltk.word_tokenize(tweet)

stop_words = set(stopwords.words('english'))
filtered_words = [word for word in words if word.lower() not in stop_words]
filtered_text = ' '.join(filtered_words)
print("Original words:", words)
```

OUTPUT:

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Original words: ['The', 'quick', 'brown', 'foxes', 'are', 'jumping', 'over', 'the', 'lazy', 'dogs', "''", 'bones', '.']
Lemmatized words: ['The', 'quick', 'brown', 'fox', 'are', 'jumping', 'over', 'the', 'lazy', 'dog', "''", 'bone', '.']
Stemmed words: ['the', 'quick', 'brown', 'fox', 'are', 'jump', 'over', 'the', 'lazi', 'dog', "''", 'bone', '.']
Tweet without stop words: quick brown foxes jumping lazy dogs  bones .
```

```
print("Lemmatized words:", lemmatized_words)
print("Stemmed words:", stemmed_words)
print("Tweet without stop words:", filtered_text)
```

RESULT: