MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

DEPARTMENT OF INFORMATION & COMMUNICATION TECHNOLOGY

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL

CERTIFICATE

This is to certify that Ms./Mr. ……...…… Reg.No. …..…… Section: …… Roll No: …..…… has satisfactorily completed the lab exercises prescribed for Data Mining and Predictive Analysis Lab [ICT 3262] of Fifth Semester B. Tech. (CCE) Degree at MIT, Manipal, in the academic year July-December 2020.

Date: …….....

Signature of the faculty

CONTENTS

LAB NO.	TITLE	PAGE NO.	SIGNATURE	REMARKS
	COURSE OBJECTIVES, OUTCOMES AND EVALUATION PLAN	i		
	INSTRUCTIONS TO THE STUDENTS	ii		
1	CREATING PHYSICAL DATA MODEL WITH IBM-INFOSPHERE	1		
2	CREATING DATA FLOWS WITH IBM-INFOSPHERE RAPID MINER OPERATORS FOR PREPROCESSING	19		
3	CREATING CONTROL FLOW WITH IBM-INFOSPHERE	34		
4	RAPID MINER OPERATORS	39		
5	DATA VISUALIZATION AND MODELING USING CLASSIFICATION	52		
6	MINI PROJECT SYNOPSIS SUBMISSION	58		
7	APRIORI ALGORITHM	59		
8	K-MEANS ALGORITHM	63		
9	DECISION TREE ID3 ALGORITHM FOR CLASSIFICATION	65		
10	NAÏVE BAYES CLASSIFIER	68		
11	MNIPROJECT : IMPLEMENTATION	71		
12	MINIPROJECT : PROGRESS	72		
	REFERENCES	73		

Course Objectives

- Familiarization with Rapid miner and InfoSphere
- Implementation of frequent pattern finding algorithms for association rule mining
- Implementation of clustering algorithms
- Implementation of the classification and predictive algorithms
- Implementation of a mini project on application of the above

Course Outcomes

At the end of this course, students will be able to

- Compare three main categories of data mining such as association rule mining, clustering, and classification
- Implement algorithms under each category
- Gain exposure to various tools and techniques for data miing and predictive analysis
- Implement a mini project by applying data mining algorithms on bench mark data sets

Evaluation plan

Split up of 60 marks for Regular Lab Evaluation
Lab 1 to Lab 4: (20 Marks) [Record:8M Execution:4M Viva/Execution Test:8M] Lab 5 to 6 : (10 Marks) [Record:4M Execution:2M Viva/Execution Test:4M] Lab 7 to Lab 10: (30 Marks)[Record:12M Execution:6M Viva/Execution Test:12M]
End Semester Lab evaluation: 40 marks (Duration 2 hrs)
Miniproject : 20 Marks Endsem: 20Marks

INSTRUCTIONS TO THE STUDENTS

Pre- Lab Session Instructions

1. Students should carry the Lab Manual Book and the required stationery to every lab session
2. Be in time and follow the institution dress code
3. Must sign in the log register provided
4. Make sure to occupy the allotted seat and answer the attendance
5. Adhere to the rules and maintain the decorum

In- Lab Session Instructions

- Follow the instructions on the allotted exercises
- Show the program and results to the instructors on completion of experiments
- Prescribed textbooks and class notes can be kept ready for reference if required

General Instructions for the exercises in Lab

- Implement the given exercise individually and not in a group.
- The programs should meet the following criteria:
    - Programs should be interactive with appropriate prompt messages, error messages if any, and descriptive messages for outputs.
    - Comments should be used to give the statement of the problem.
    - Statements within the program should be properly indented.
- Plagiarism (copying from others) is strictly prohibited and would invite severe penalty in evaluation.
- In case a student misses a lab, he/ she must ensure that the experiment is completed before the next evaluation with the permission of the faculty concerned.
- Students missing out lab on genuine reasons like conference, sports or activities assigned by the Department or Institute will have to take **prior permission** from the HOD to attend **additional lab** (with other batch) and complete it **before** the student goes on leave. The student could be awarded marks for the write up for that day provided he submits it during the **immediate** next lab.
- Students who fall sick should get permission from the HOD for evaluating the lab records. However attendance will not be given for that lab.
- Students will be evaluated only by the faculty with whom they are registered even though they carry out additional experiments in other batch.
- Presence of the student during the lab end semester exams is mandatory even if the student assumes he has scored enough to pass the examination
- Minimum attendance of 75% is mandatory to write the final exam.
- If the student loses his book, he/she will have to rewrite all the lab details in the lab record.
- Questions for lab tests and examination are not necessarily limited to the questions in the manual, but may involve some variations and / or combinations of the questions.

THE STUDENTS SHOULD NOT

- Bring mobile phones or any other electronic gadgets to the lab.
- Go out of the lab without permission.

LAB NO. 1 **Date**

CREATING PHYSICAL DATA MODEL WITH IBM-INFOSPHERE

Objectives

1. To get acquainted with Infosphere.
2. To create physical data model.

Introduction to Infosphere

IBM InfoSphere is a licensed software that provides capabilities to build a datawarehouse in Db2. It helps to understand the data, cleanse and integrate data from heterogeneous sources to gain faster business insight, at lower cost. Infoshpere has two components i.e Data warehousing in Db2 application server and Data warehousing in Db2 client. The Db2 client is called Design Studio. Additionally, InfoSphere Warehouse provides tutorials that work with the sample database to show how to use the SQL Warehousing, Cubing Services, and Mining features of the product. InfoSphere Warehouse is a suite of products which can be used to build a complete data warehousing solution that includes a highly scalable relational database, data access capabilities, and front-end analysis tools.

Working with Design Studio

On click of Design Studio, create a new workspace as shown in Figure 1.1. The newly created workspace will contain all the works performed to create a data warehouse under one folder.

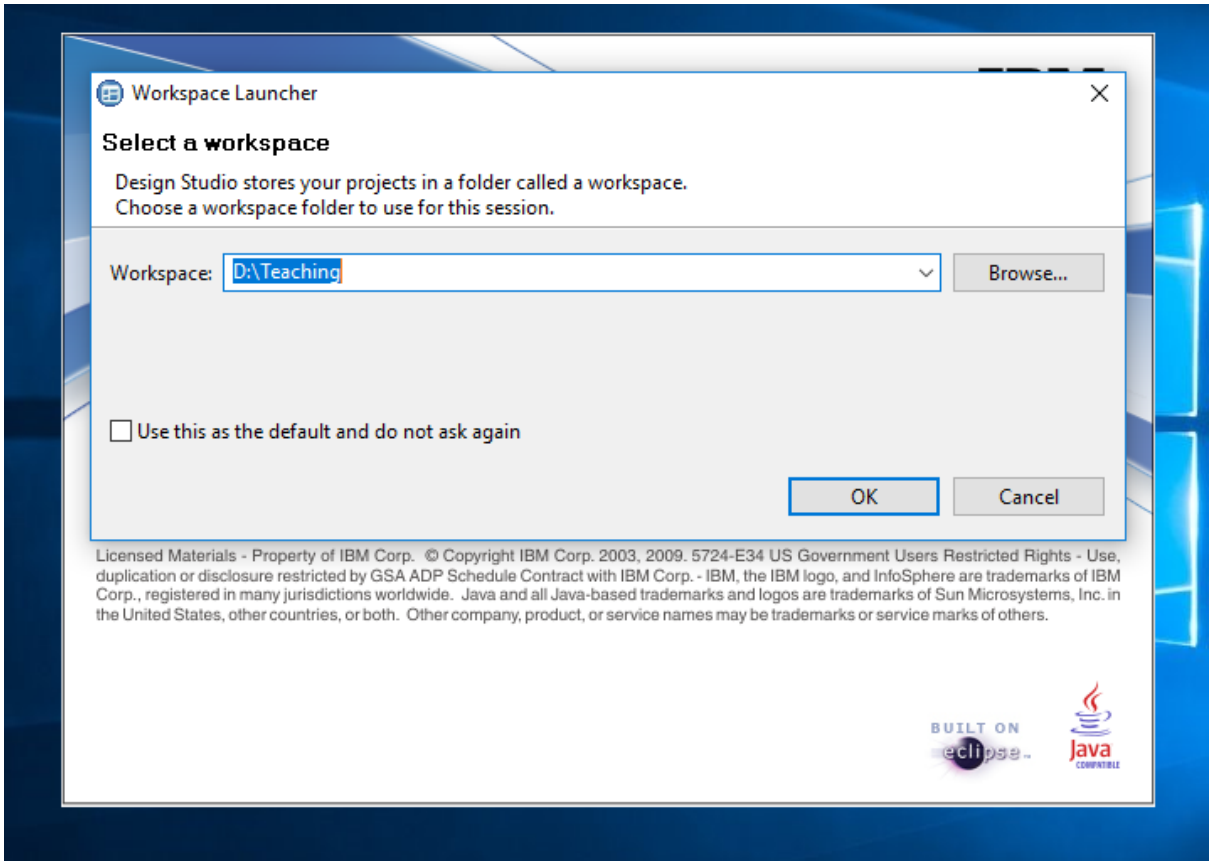

Figure 1.1: Workspace creation

On creation of a new workspace there appears a welcome page as shown in Figure 1.2. The welcome page furnishes details about various products present in the Infosphere suite.

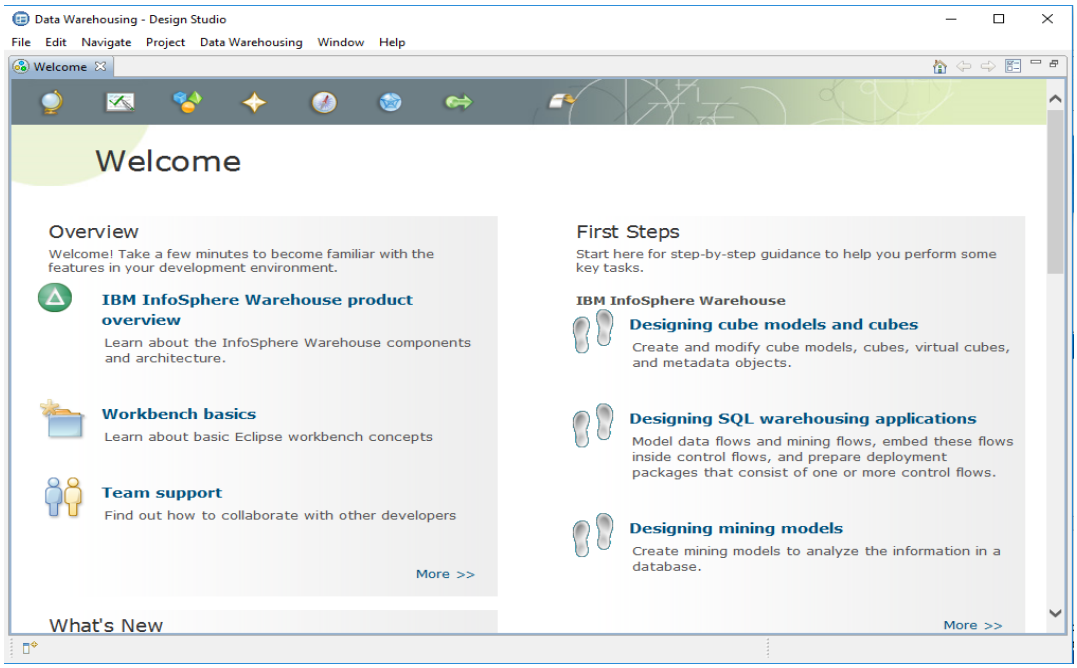

Figure 1.2: Welcome page of Design Studio

Infosphere work bench window will open after closing the welcome page, which has four main parts: Data Project Explorer, Data Source Explorer, Canvas, Properties (which also has SQL results, Job Status etc). Data Project Explorer is where new data warehousing projects appear, Data Source explorer is used to connect to server and view data. Canvas is used to develop data flows and control flows. Properties tab is used to set properties for schemas during data warehouse creation. Figure 1.3 shows Infosphere work bench.

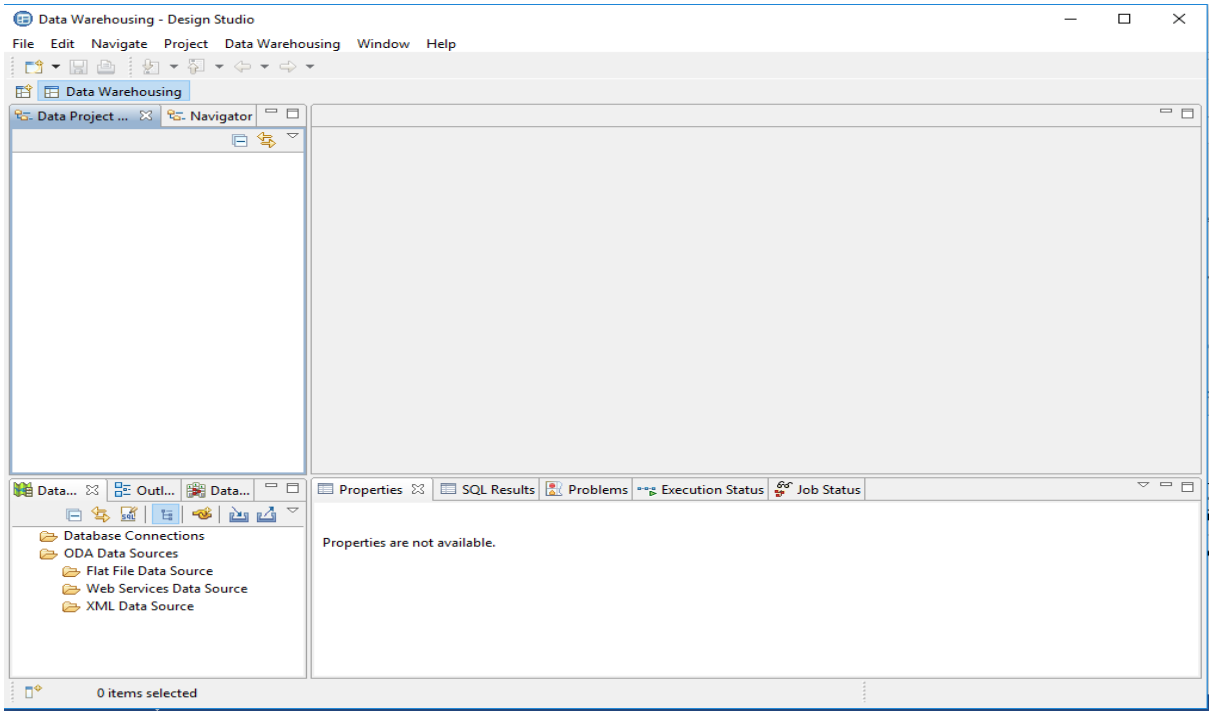

Figure 1.3: Infosphere work bench

Data Source Explorer

Before creating a new data warehouse project, it is required to connect to external data warehouse/database present on the server side. This is necessary because the new data warehouse that is created may obtain data from the already available datawarehouse/database.

To obtain database connectivity perform the following steps:

Data Source Explorer → Database connections → New →Connection Parameter window will open. Set the parameter as shown in Figure 1.4

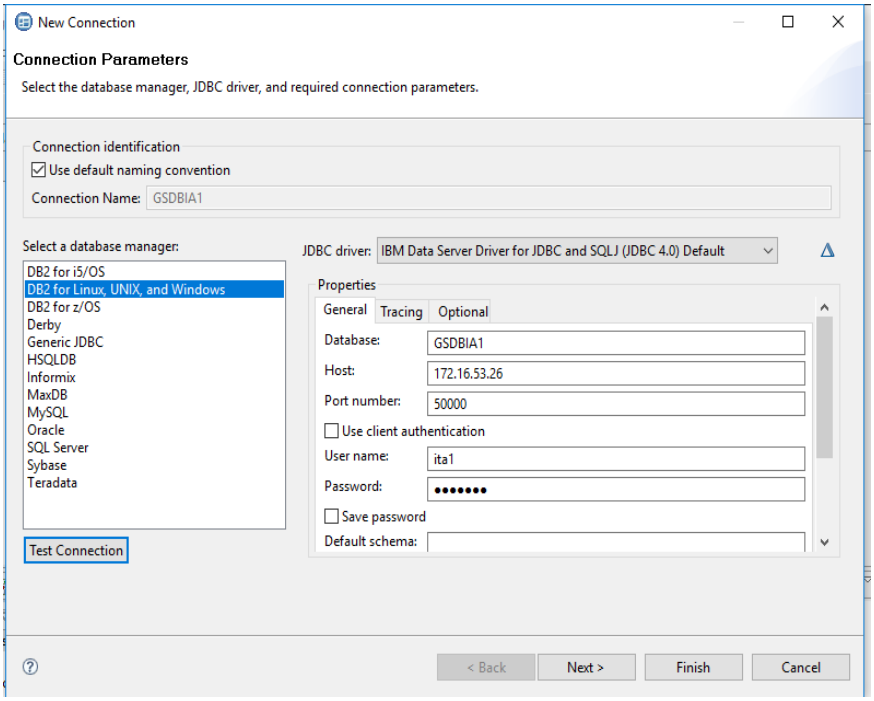

Figure 1.4: Connection configuration

NOTE: The host IP, database, username and password are subjected to changes.

Click on "**Test Connection**" button in order to check whether the connection is successful. A "**connection succeeded**" message will be shown once connection is set up. Click on "**Finish**" to navigate to main window. Connection to the database will appear as shown in the Figure 1.5.

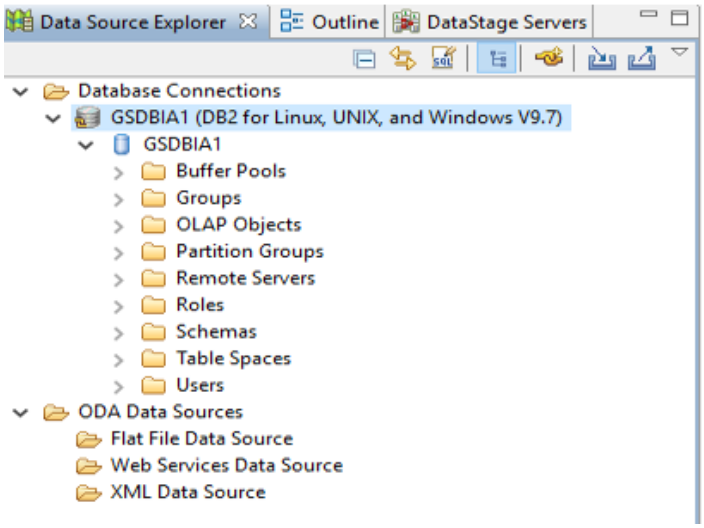

Figure 1.5: Database connection

Various schemas present under the connection can be viewed by expanding **Schemas** as shown in Figure 1.6.

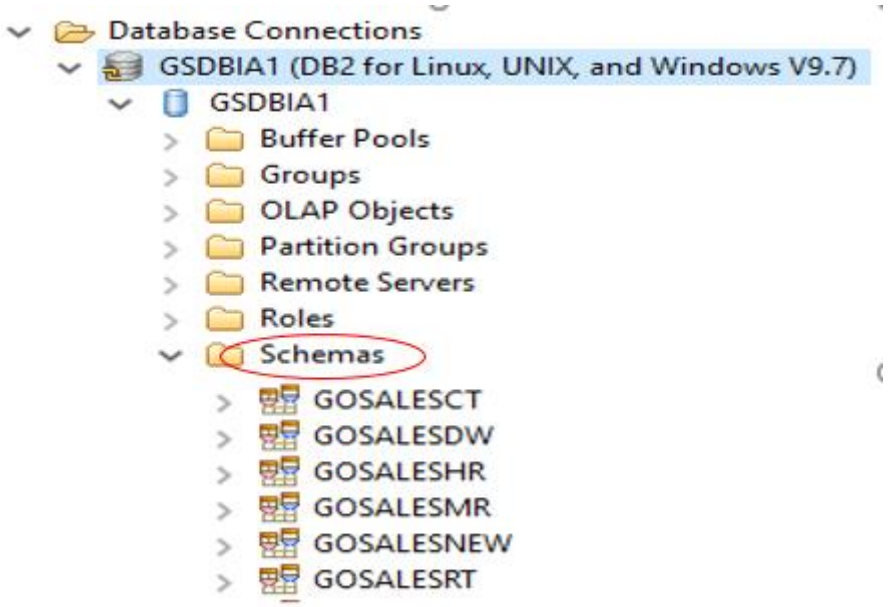

Figure 1.6: Various schemas provided by IBM

A schema, tables under the schema, its attributes and data type of the attributes can be explored by expanding the schema as shown in Figure 1.7

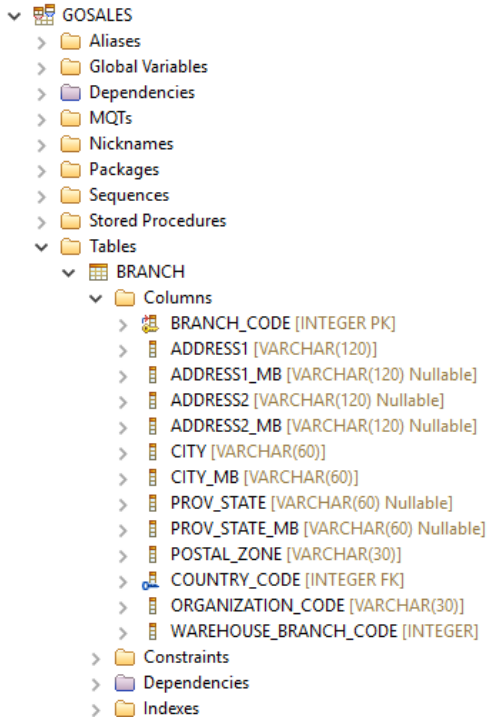

Figure 1.7: Expanded schema “GOSALES”

The data in a table can be viewed as (right click)TABLE → Data →Sample Contents. Figure 1.8 shows the steps to view table data.

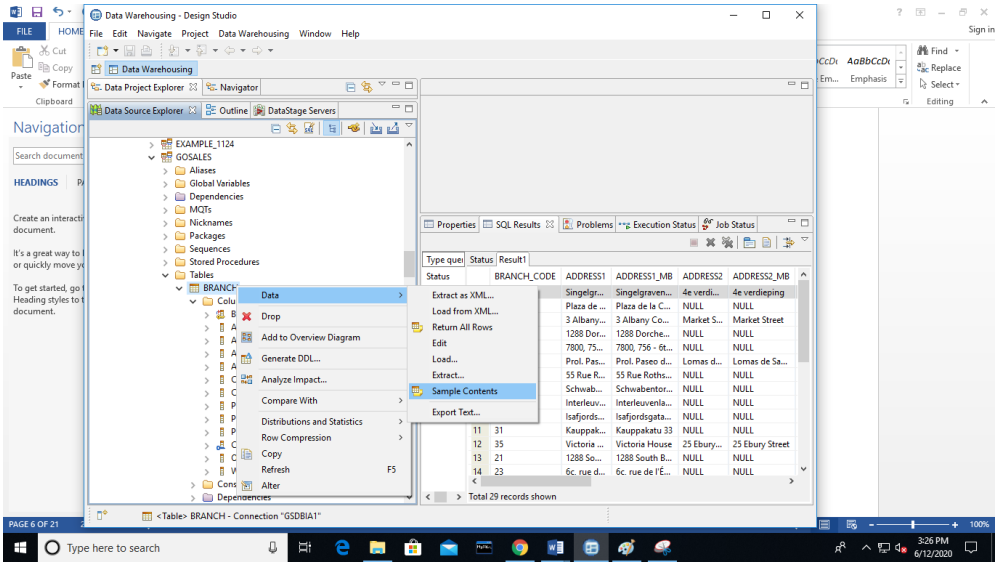

Figure 1.8: View of data in a table

Data Project Explorer

Data project explorer is used to view data warehousing projects. A data warehouse is created within a physical model. There are two ways to create a physical model. They are:

i. **Create using reverse engineering:** Existing tables and data of the tables from the server is used to create data warehouse.

ii. **Create using template:** Tables are created from scratch and data is dumped newly.

To create a new data warehousing project: File → New→ Data warehousing project→ **New Project** window will appear where a project name needs to be given →Finish. Figure 1.9 shows the creation of Data warehousing project.

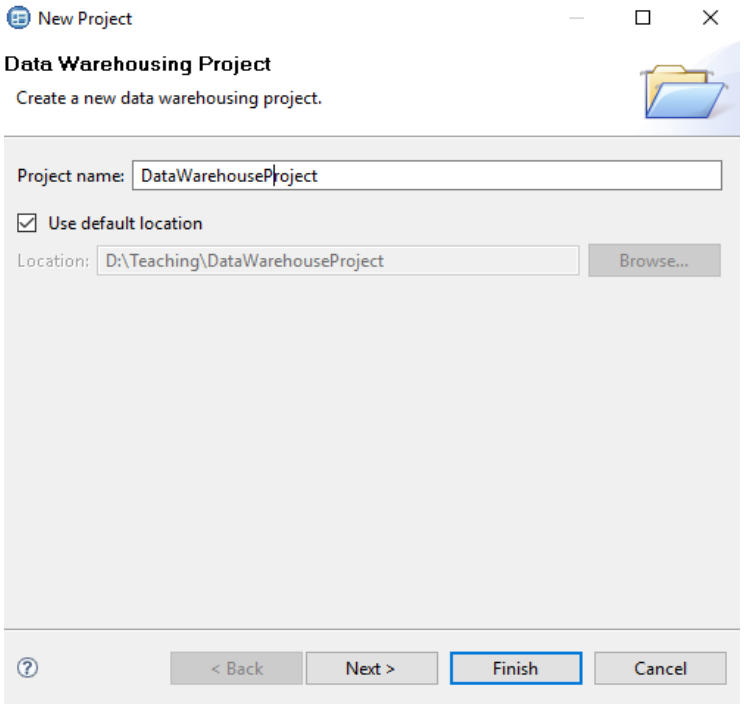

Figure 1.9: Data warehousing project creation

On expanding the newly created data warehousing project various folders would appear. Among all, **Data models**, **Data Flows** and **Control Flows** are important to us. To create a new physical data model, (right click) Data model → New→ Physical Data model. The window in Figure 1.10 appears.

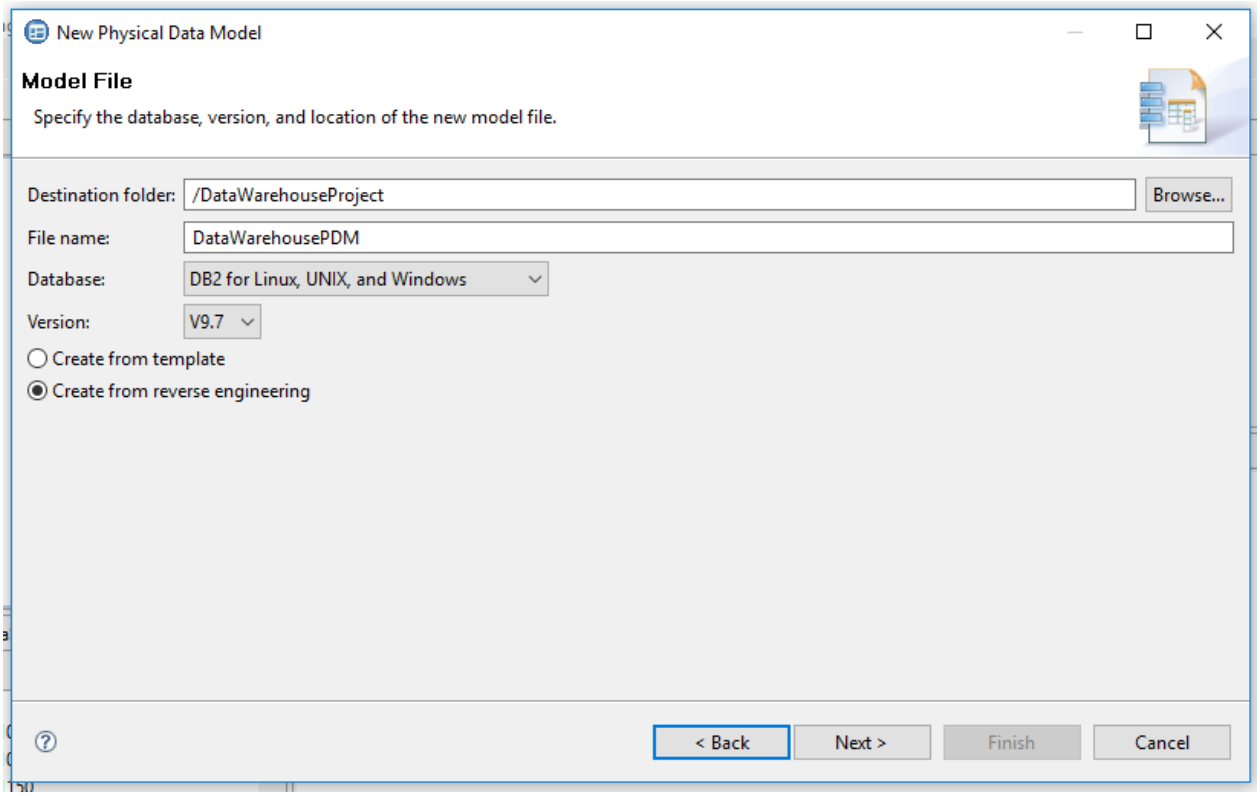

Figure 1.10: New physical model creation

In new physical model window, choose the following configurations:
File name: A new name to the database model
Database: Db2 for Linux, UNIX, Windows
Version: V9.7 /* Db2 version in server is 9.7*/
Choose "**Create from template**" or "**Create from reverse engineering**".

In the following section, creation of a new data warehouse with two tables **BRANCH** AND **COUNTRY** from **GOSALES SCHEMA** using Reverse Engineering process is explored.

Exploring create from Reverse Engineering:
To create a data warehouse using reverse engineering, provide the above mentioned configuration in **New Physical Data Model Window**, Click on **Next.** In **Source** window select **Database** → In **Select Connection** window → select the connection name to which connection has to be established→ **Click on Next** → In Select schemas, select GOSALES (because tables from GOSALES are imported for the creation of a new data warehouse) → Data elements (Let the default selections remain as it is)
Once a new physical data model is created using "**reverse engineering**", the project under "Data Project explorer" appears as shown in Figure 1.11

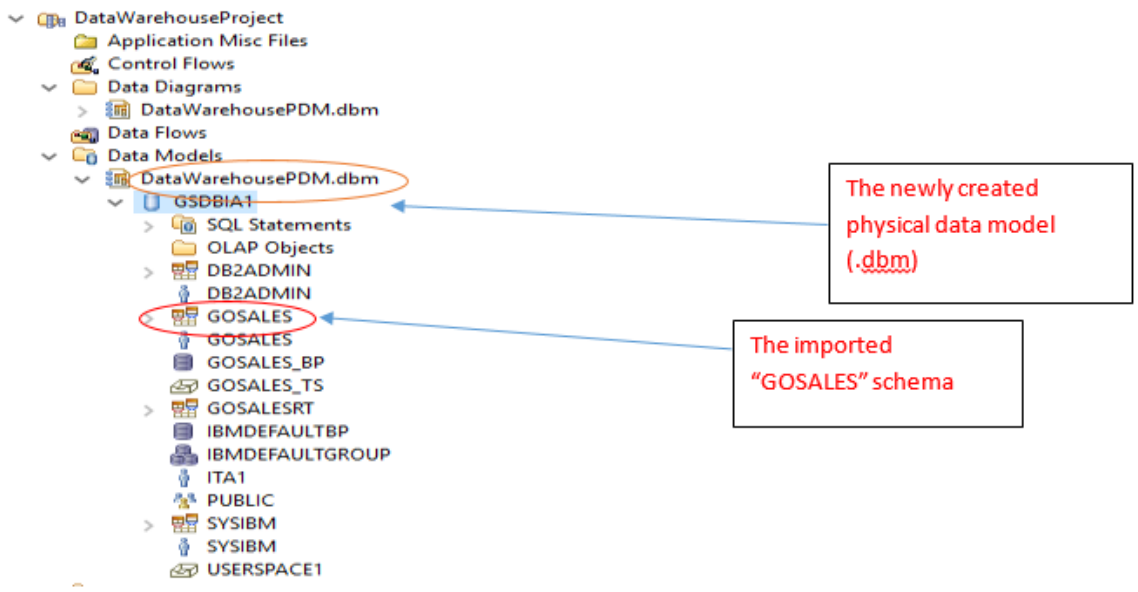

Figure 1.11 Data Project explorer after reverse engineering process.

The GOSALES schema has many tables under it. Since the need is only **Branch** and **Country** table to create data warehouse, a new schema, "**BRANCH_COUNTRY_SCHEMA**" is created which will have two tables BRANCH and COUNTRY. To create a new schema, (right click) connection name →Add Data Object → Schema. The steps are shown in Figure 1.12.

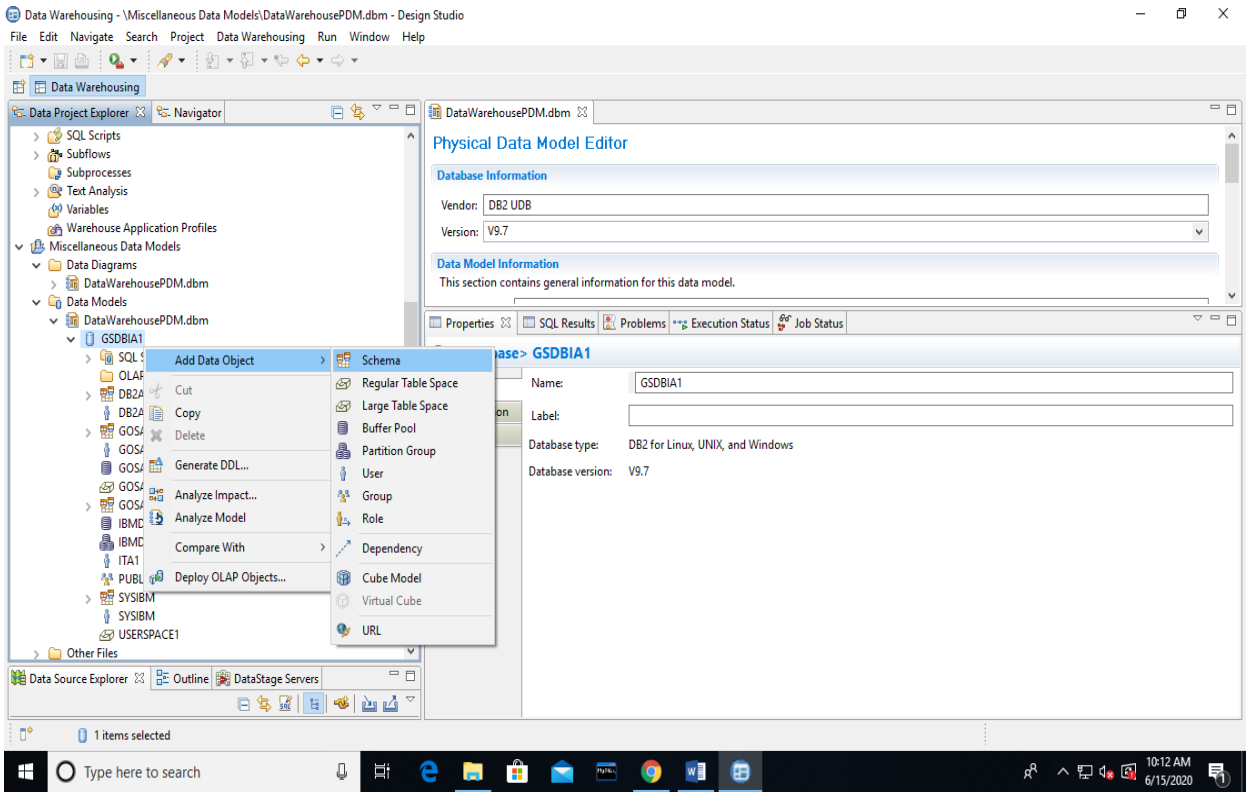

Figure 1.12: Creation of new schema "BRANCH_COUNTRY_SCHEMA"

By default, Design Studio will name the newly created schema as "**Schema**". The default name can be changed in Properties tab as shown in Figure 1.13

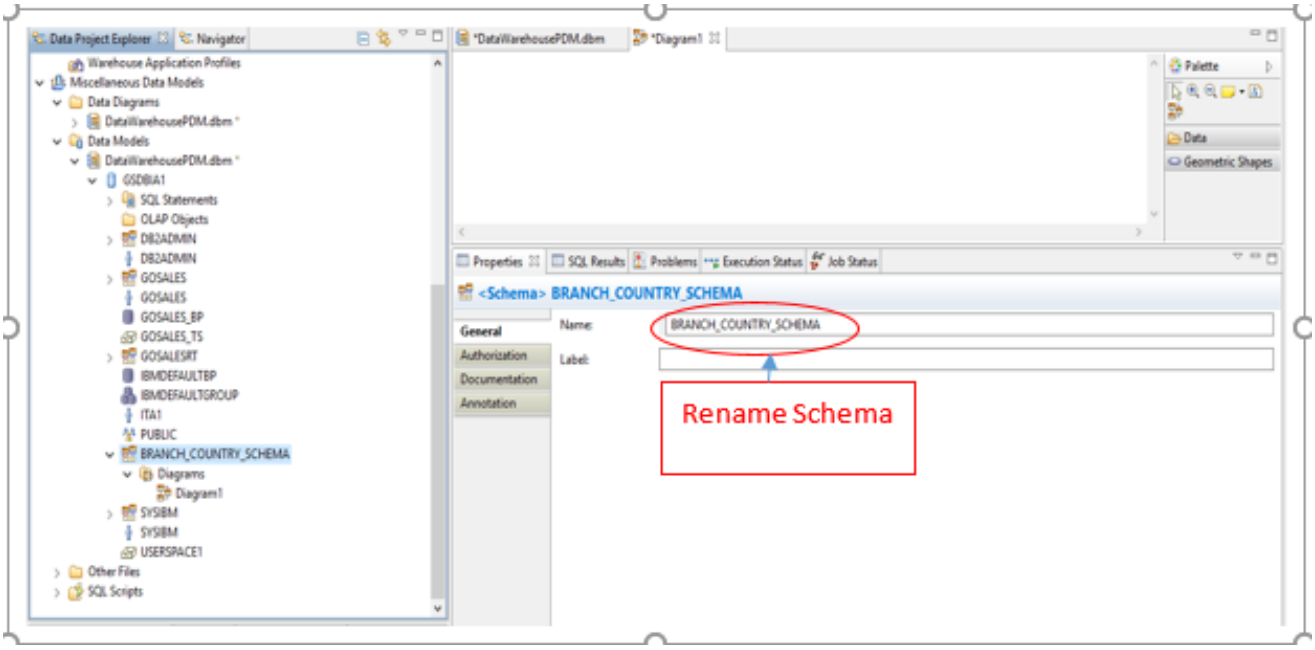

Figure 1.13: To rename the schema name.

Once the schema is created, import the tables "**BRANCH**" and "**COUNTRY**" into "**BRANCH_COUNTRY_SCHEMA**" from the imported schema "**GOSALES**" by using the copy option on right click of **BRANCH** AND **COUNTRY** tables in **GOSALES.** Figure 1.14 and Figure 1.15 shows the import process on copy.

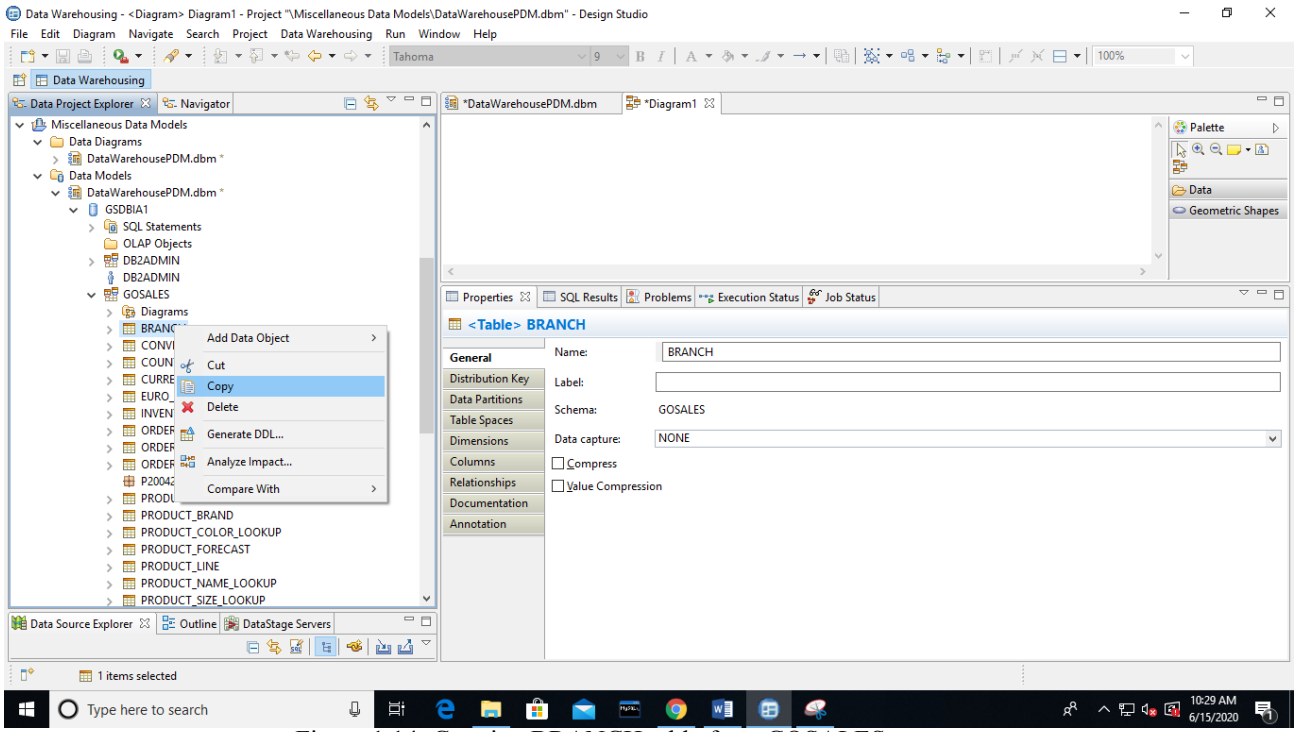

Figure 1.14: Copying BRANCH table from GOSALES

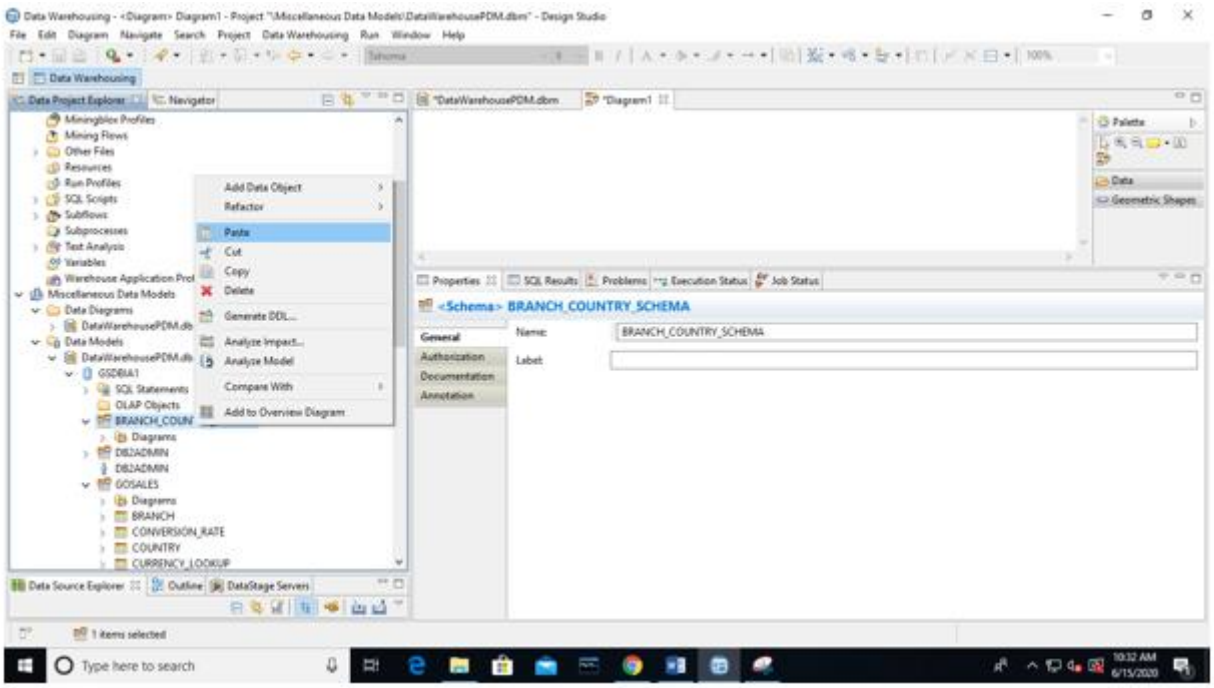

Figure 1.15: Pasting BRANCH table in BRANCH_COUNTRY_SCHEMA

Similar process is carried out to create **COUNTRY** table under BRANCH_COUNTRY_SCHEMA

Figure 1.16 shows, **BRANCH_COUNTRY_SCHEMA,** after importing both the tables "BRANCH" and "COUNTRY" into it.

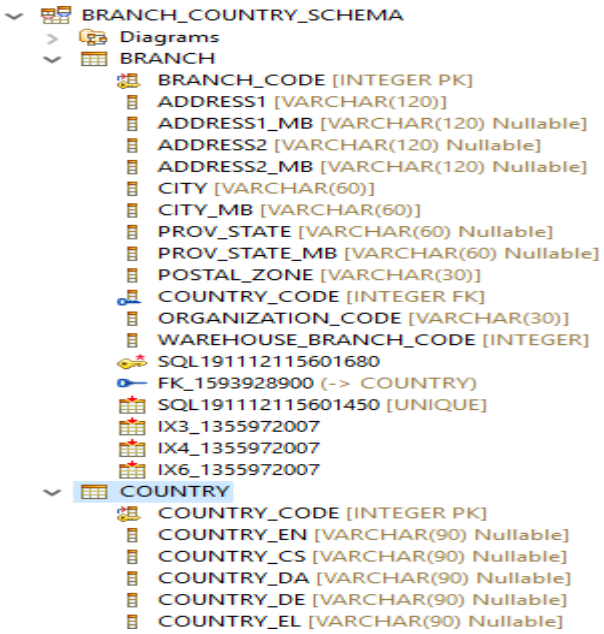

Figure 1.16: After importing BRANCH and COUNTRY into BRANCH_COUNTRY_SCHEMA

BRANCH and **COUNTRY** has many attributes in it. Eliminate other attributes by retaining branch_code, address1, city from BRANCH and Country_code and Country_EN in COUNTRY. This is required to do as the data warehouse created will have attributes that are required as per the need. To delete attributes from a table, select all attributes by pressing on **ctrl** key , press **delete** key. Figure 1.17 shows the BRANCH table after deleting all other attributes.

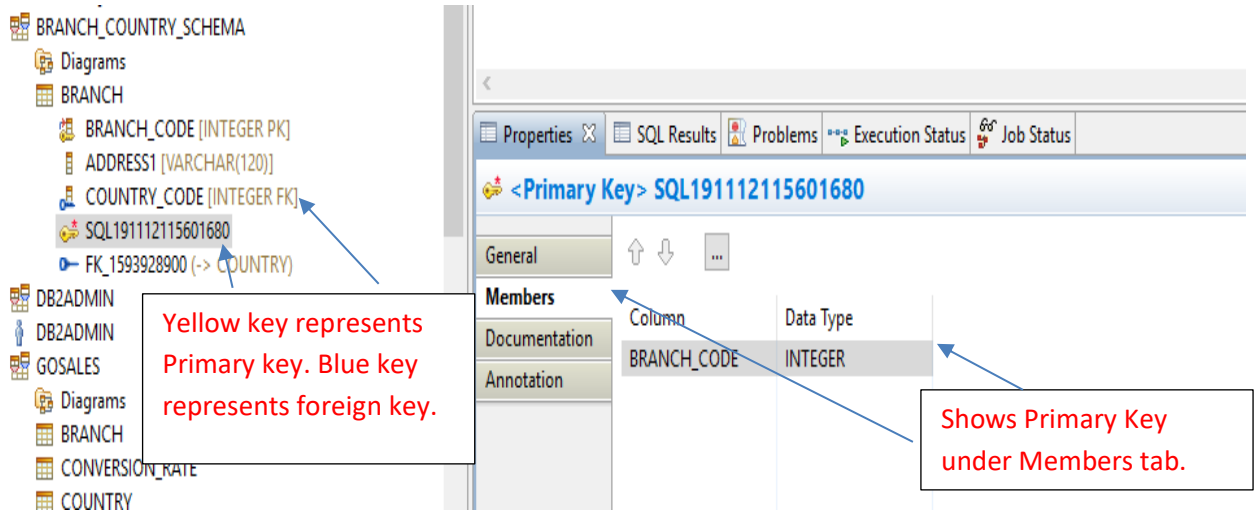

Figure 1.17: After deleting all other attributes which is not required for the data warehouse

The same process has to be repeated with **COUNTRY** table. After preparing both the tables, **BRANCH_COUNTRY_SCHEMA** schema (data warehouse) is shown in Figure 1.18.

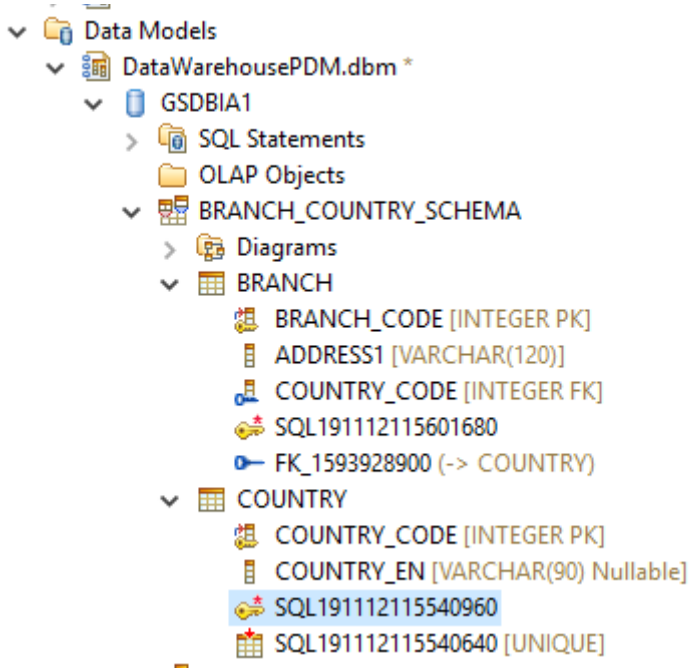

Figure 1.18: After preparing the tables BRANCH and COUNTRY in BRANCH_COUNTRY_SCHEMA

The schema **BRANCH_COUNTRY_SCHEMA** is created on the client side. For anyone to work on the schema created by you, it is required to generate it on the server side.

To generate the schema on the server side: Do the following:
(Right click) schema (**BRANCH_COUNTRY_SCHEMA) → Generate DDL** → A window, Generate DDL will appear. Click on the checkboxes as shown in Figure 1.19

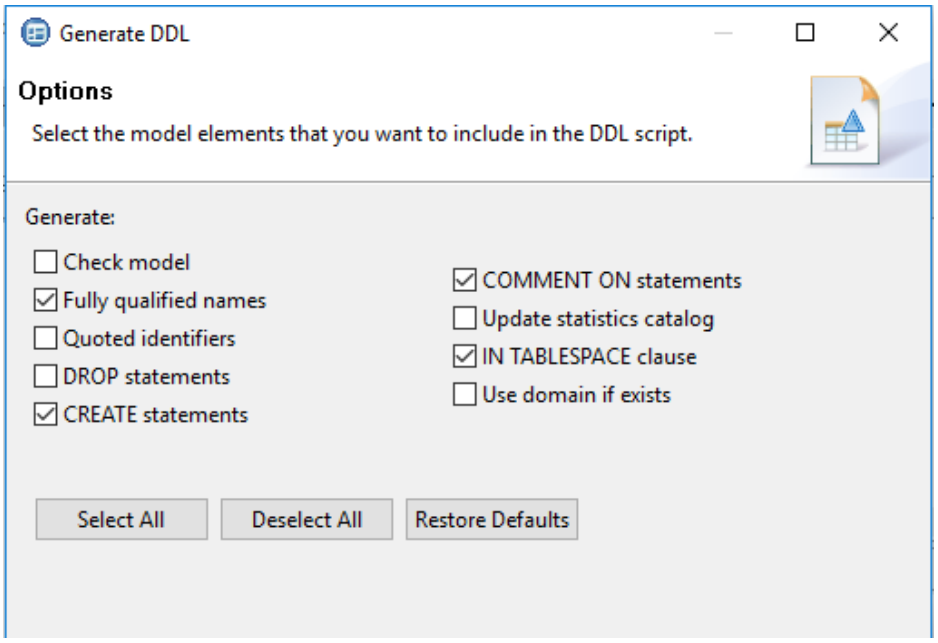

Figure 1.19: Configurations during DDL generation

NOTE: Fully qualified names take the absolute path of the object in a schema.

Click on **Next.** In the next window, check the options as shown in Figure 1.20.

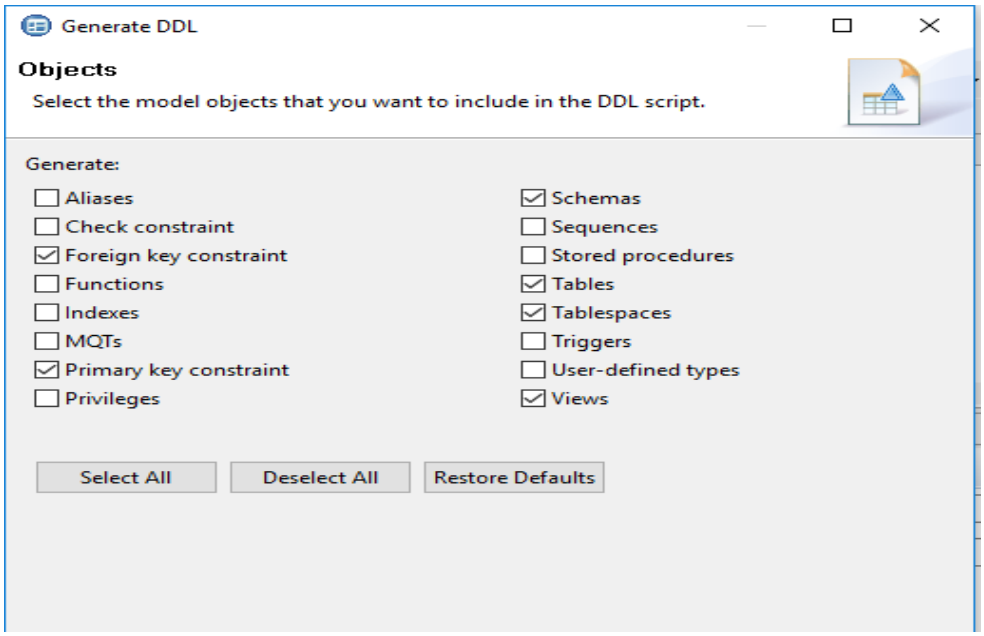

Figure 1.20: Configurations during DDL generation

Click on **Next**. In the next window, **Save and Run DDL**, Check Run DDL on Server as shown in Figure 1.21. Also, the DDL generated by Design Studio can be seen in this window.

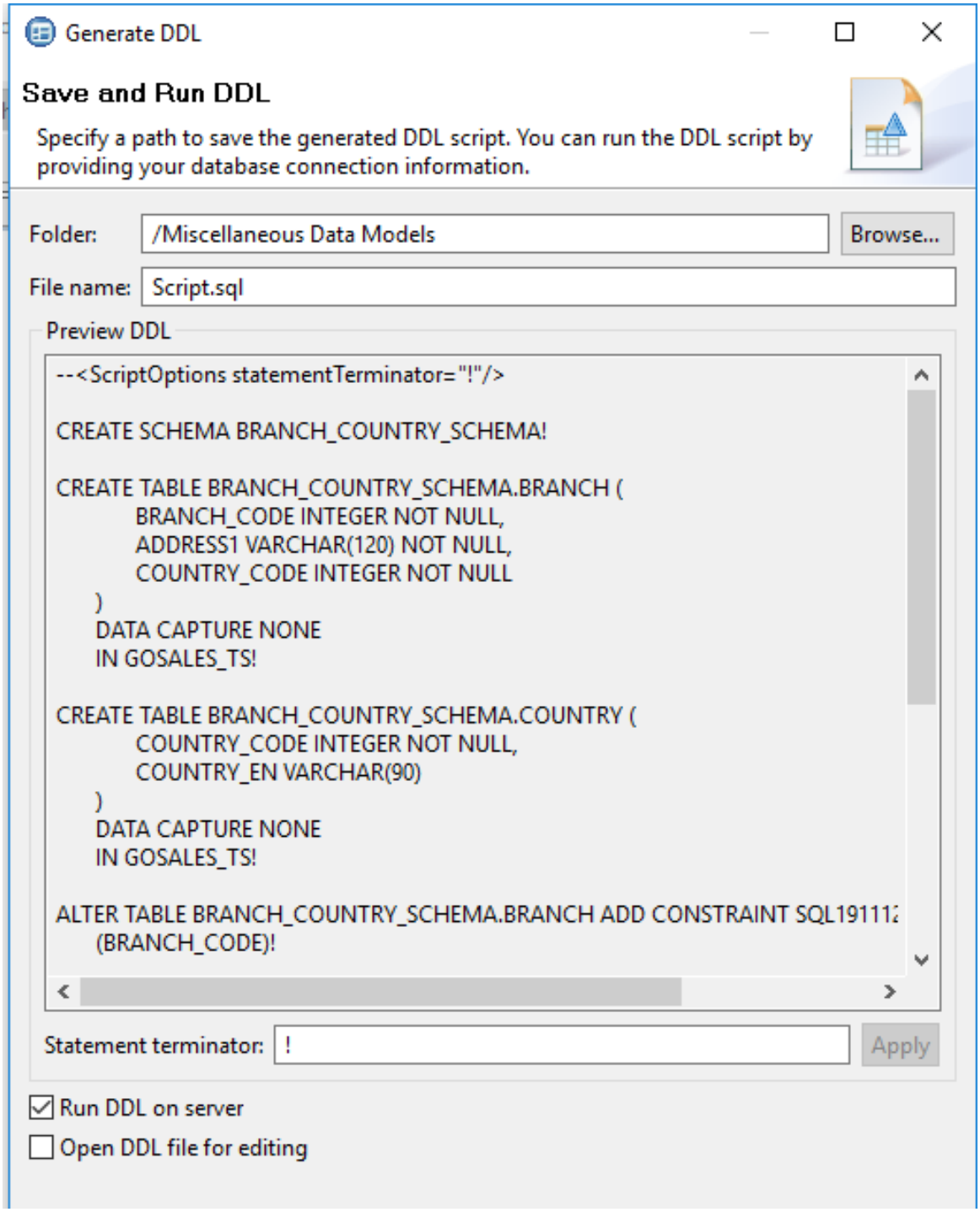

Figure 1.21: Configurations during DDL generation

Click on **Next**. Select the connection name to which connection has to be established. Click on **Next** and **Finish.** Once schema is generated successfully on sever side, Succeeded message will be shown under SQL Results tab as shown in Figure 1.22. The newly generated schema should be visible under Data Source Explorer, in the set connection as shown in Figure 1.23

NOTE: The connection has to be refreshed, in order to view the newly created schema.

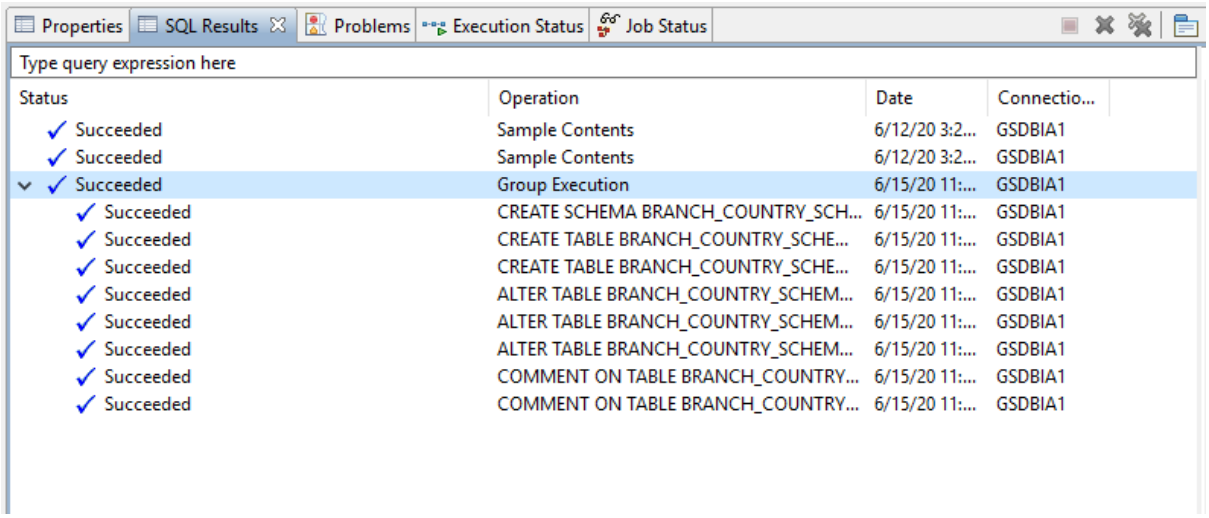

Figure 1.22: On Success of schema generation

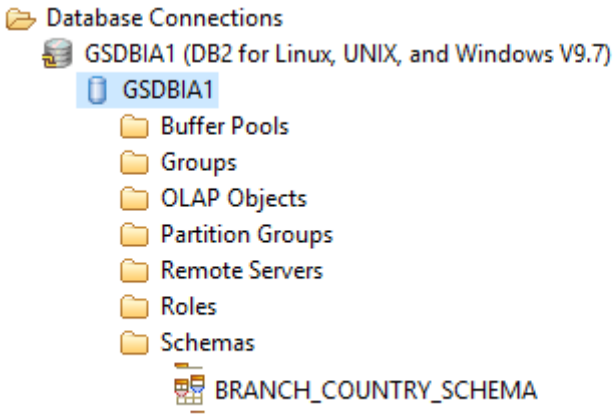

Figure 1.23: Schema appearing under the set connection

Hence, the schema **BRANCH_COUNTRY_SCHEMA** is generated successfully on server side using **Create from Reverse Engineering process.** The newly created schema **BRANCH_COUNTRY_SCHEMA is** empty in its tables because, even though tables are imported from the existing schema **GOSALES,** the data is not carried. To load data from GOSALES. BRANCH to BRANCH_COUNTRY_SCHEMA. BRANCH and GOSALES. COUNTRY to BRANCH_COUNTRY_SCHEMA. COUNTRY, data flows need to be created. In general, to populate data into a table from a table which already exists in the server, dataflow need to be employed which will be shown in LAB 4.

Exploring create from Template:

To explore **Create from Template**, create a new data warehousing project → create new Physical Data Model → choose "**Create from Template**" as options shown in Figure 1.10. Click on **Finish**

Once a new physical model is created, create a new schema with the name **BRANCH_COUNTRY_TEMPLATE.** To add table (right click) schema → Add Data Object →Table. The screenshot for the same is shown in Figure 1.24

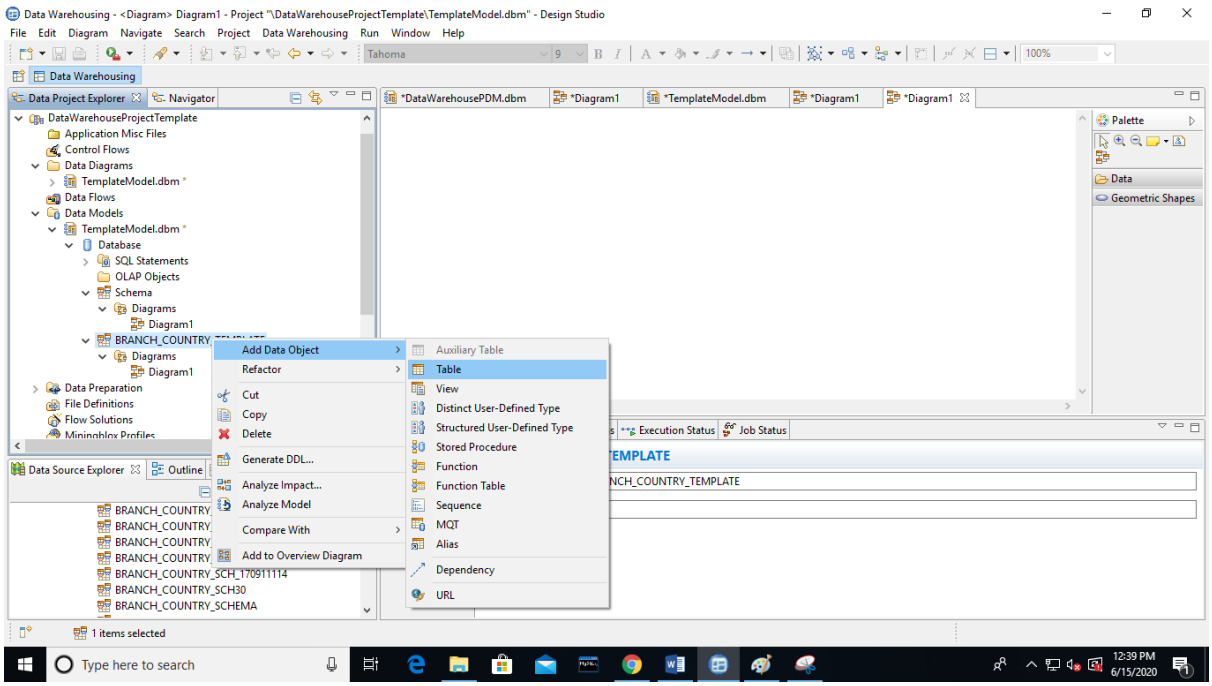

Figure 1.24: To create a table in a schema

Newly created table can be renamed under Properties tab. Create two tables namely "BRANCH" and "COUNTRY" as shown in Figure 1.24. The attributes to the tables can be added as (right click) table → Add data object → Column. Figure 1.25 shows the schema after adding one attribute BRANCH_CODE into BRANCH table.

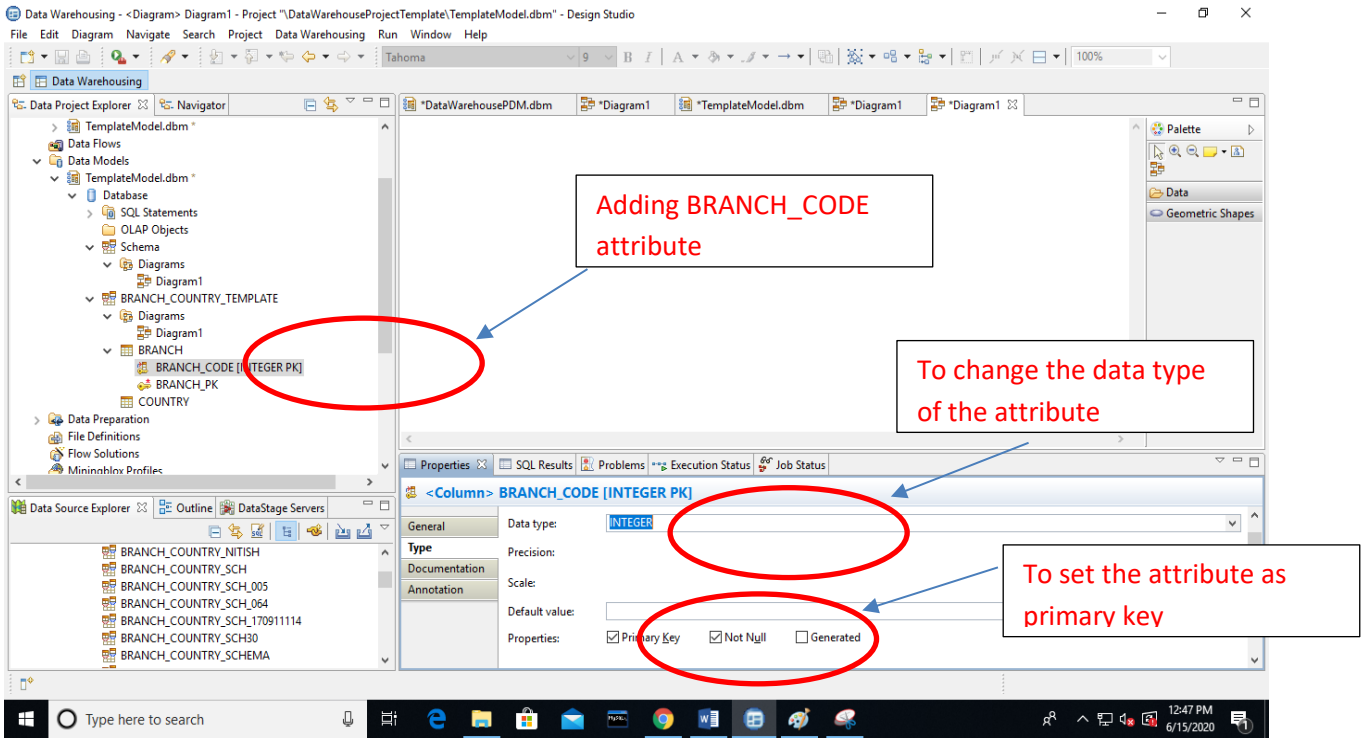

Figure 1.25: Creation of new attribute in BRANCH TABLE

NOTE: If the attribute type is char, the length can be changed under Type tab of Properties tab.

Figure 1.26 shows the **BRANCH** table after adding all the required attributes.

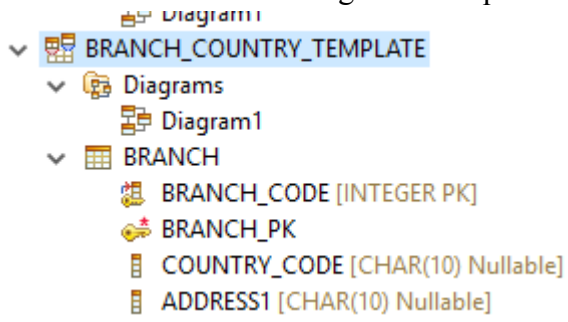

Figure 1.26: BRANCH table using create from template

Figure 1.27 shows COUNTRY table after adding all the required attributes.

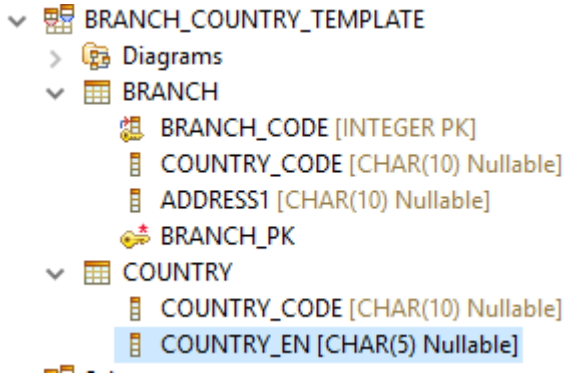

Figure 1.27: COUNTRY table using create from template

The COUNTRY_CODE in BRANCH is a foreign key to COUNTRY. To set an attribute as foreign key, (right click) table BRANCH → Add Data Object → Foreign key as shown in Figure 1.28

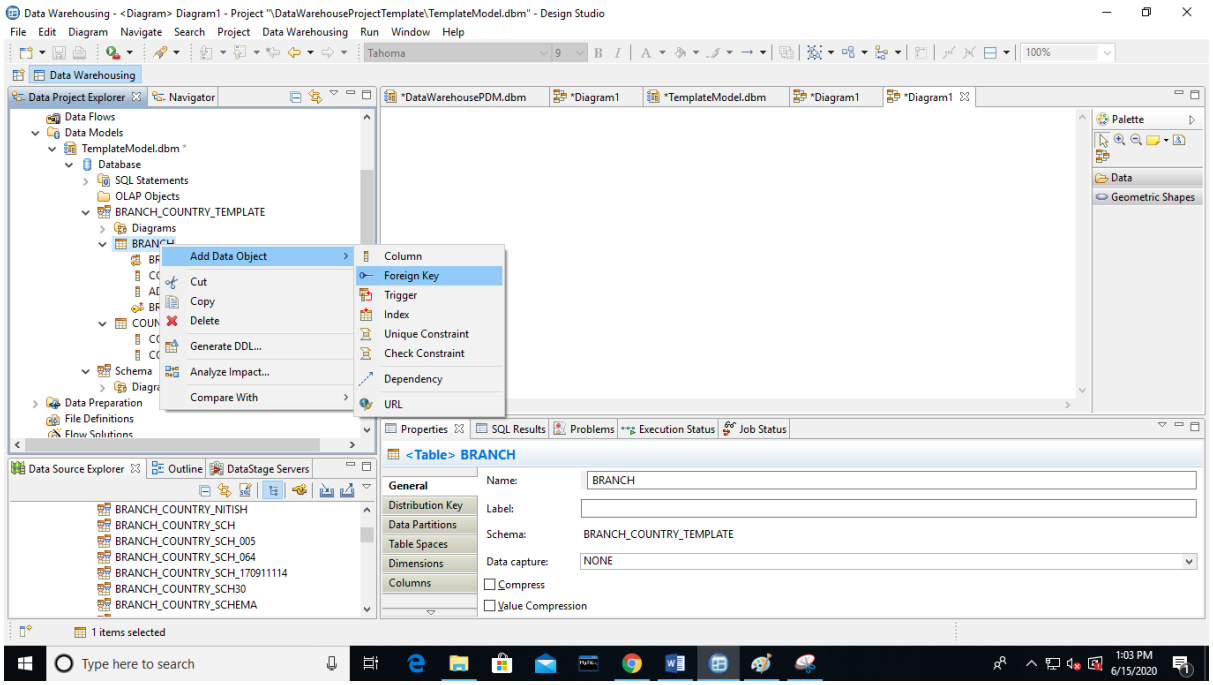

Figure 1.28: Setting foreign key on BRANCH table

On click of foreign key, the window (see figure 1.29) will open where parent table has to be selected, i.e **COUNTRY**. Click on **OK**

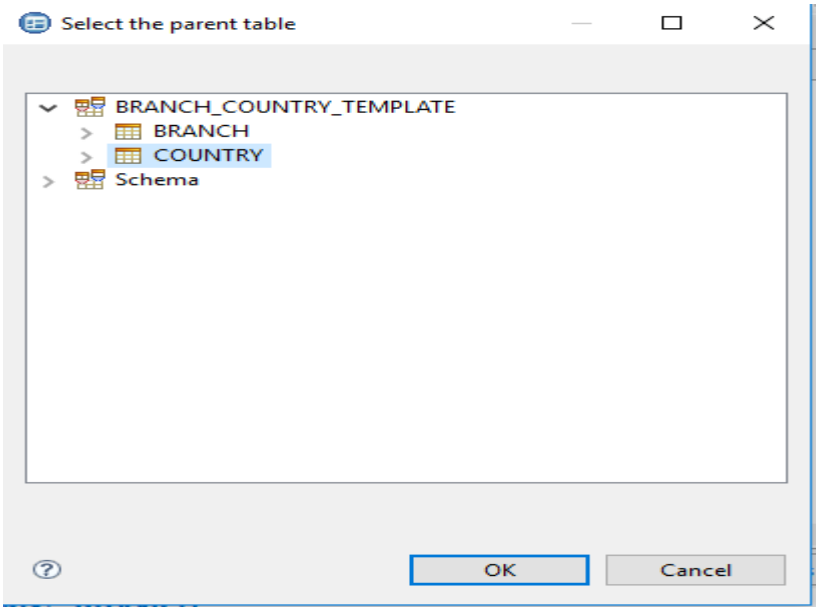

Figure 1.29: Selecting parent table i.e COUNTRY

On click of OK, following window will appear where the configurations shown in Figure 1.30 has to be performed under **GENERAL** Tab of Properties.

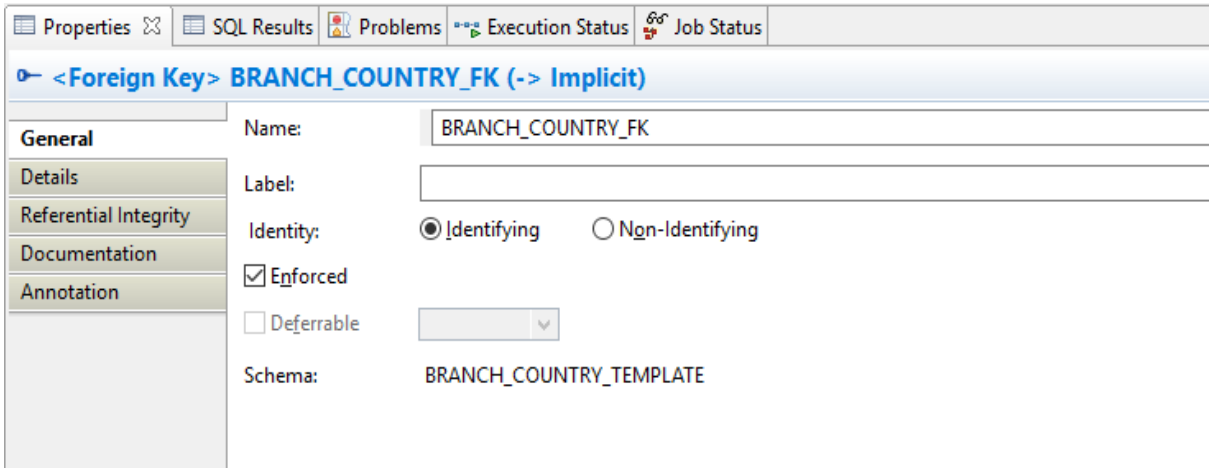

Figure 1.30: Creation of foreign key

Select Details → Key Columns → **…** button (ellipses button) → Select COUNTRY_CODE in **Select columns** window as shown in Figure 1.31. **Select Unique Constraint or Index** as **<PRIMARY KEY>**

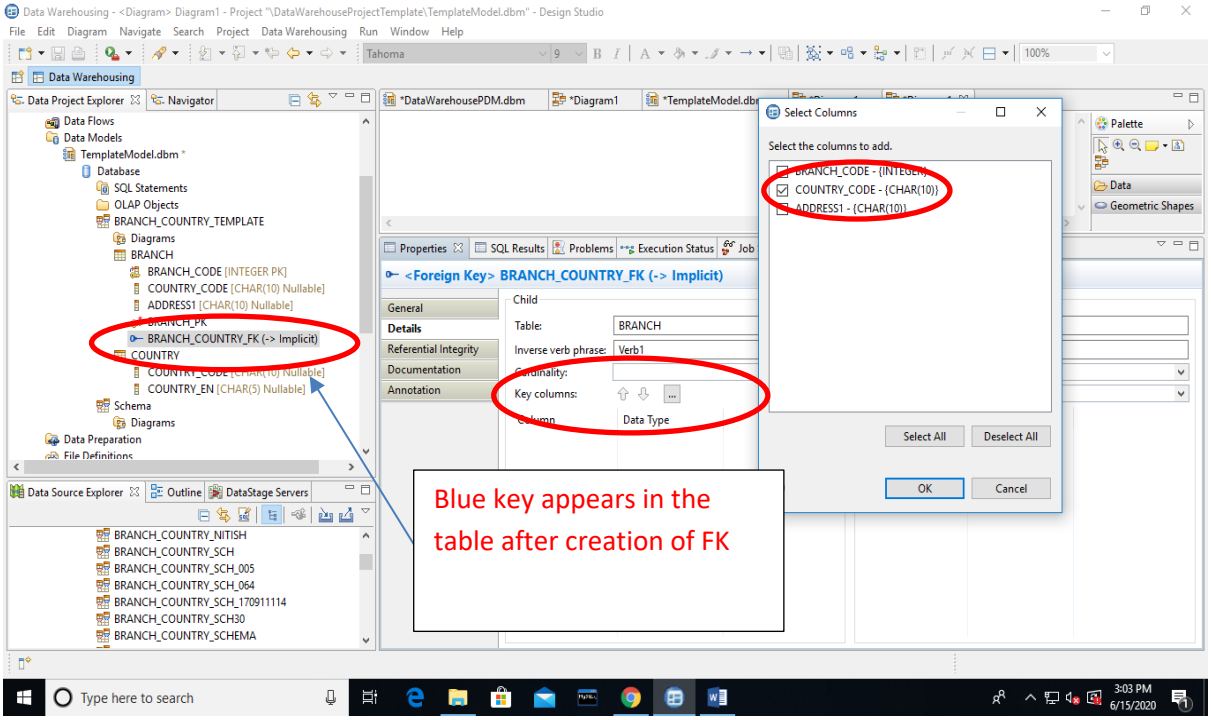

Figure 1.31: Creation of foreign key

Once the schema is created completely, it needs to be generated in the server so that anyone can use the schema created.

Once the schema is created in the server, manually data needs to be loaded into the tables. To load data into the Table BRANCH, (right click)Table BRANCH → Data →Edit. Enter the table as shown in Figure 1.32.

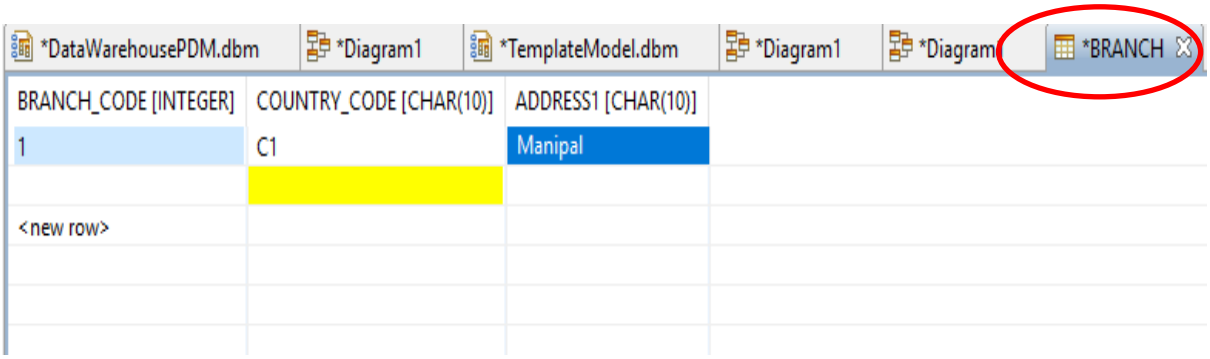

Figure 1.32: Loading data into the Table BRANCH

When the data is not saved, there shows up a * on the table name as shown in Figure 1.32. To save the data permanently, click on SAVE as shown in Figure 1.33

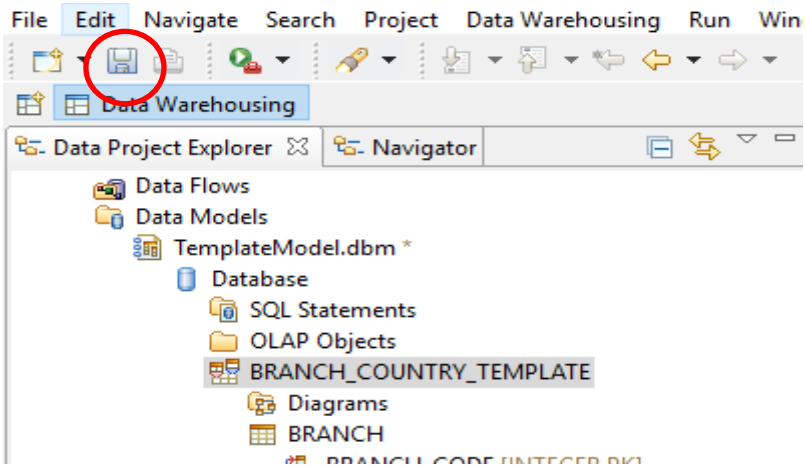

Figure 1.33: Saving the table data

Once the tables are populated with data, data flows can be constructed on these schemas in order to obtain results for the SQL queries.

Exercise

Create Physical model for the following databases using Infosphere.

1. Product(pid,pname)
   Supplier(sid,sname)
   Supply(pid,sid,qty)

2. Student (Regno, Name, Major, Bdate)
   Course (Course_ID, Cname, dept)
   Enroll (RegNo, Course_ID, marks)
   Book_Adoption (Course_ID, sem, ISBN)
   Text (book_ISBN, title, publisher, author)

Additional Exercise

Create Physical model for the following databases using Infosphere.

1. Person (driver _ id, name, address)
   Car (Regno, model, year)
   Accident (report_number, date, location)
   Owns (driver-id, Regno)
   Participated (driver-id, Regno, report_number, damage)

2. Employee (Name, SSN, Salary, DoJoin)
   Project (Pname, Pnumber, Mgr_ssn, PAddress)
   Project_Domain (Dnumber, Pno)
   Domain (Dnumber, Dname, Description)
   Works_On (Essn, Pno, hrs)

LAB NO. 2 **Date**

CREATING DATA FLOWS WITH IBM INFOSPHERE

Objective

1. To create and execute data flows

Introduction

Data flow models the SQL-based data movement and transformation activities that are executed by the Db2 database engine. A data flow consists of activities represented by graphical operators that extract data from flat files or relational tables, transform the data, and load it into a relational table in a data warehouse, data mart, or staging area.

Working of Data Flows

Data flow to load data into BRANCH_COUNTRY_SCHEMA. COUNTRY table from GOSALES. COUNTRY table

(Right click) data flow folder → New →Data Flow. The window shown in Figure 2.1 appears where a data flow name has to be given. The working mode for the data flow should be selected as Online (As online selection will help reflect changes in data in real time).

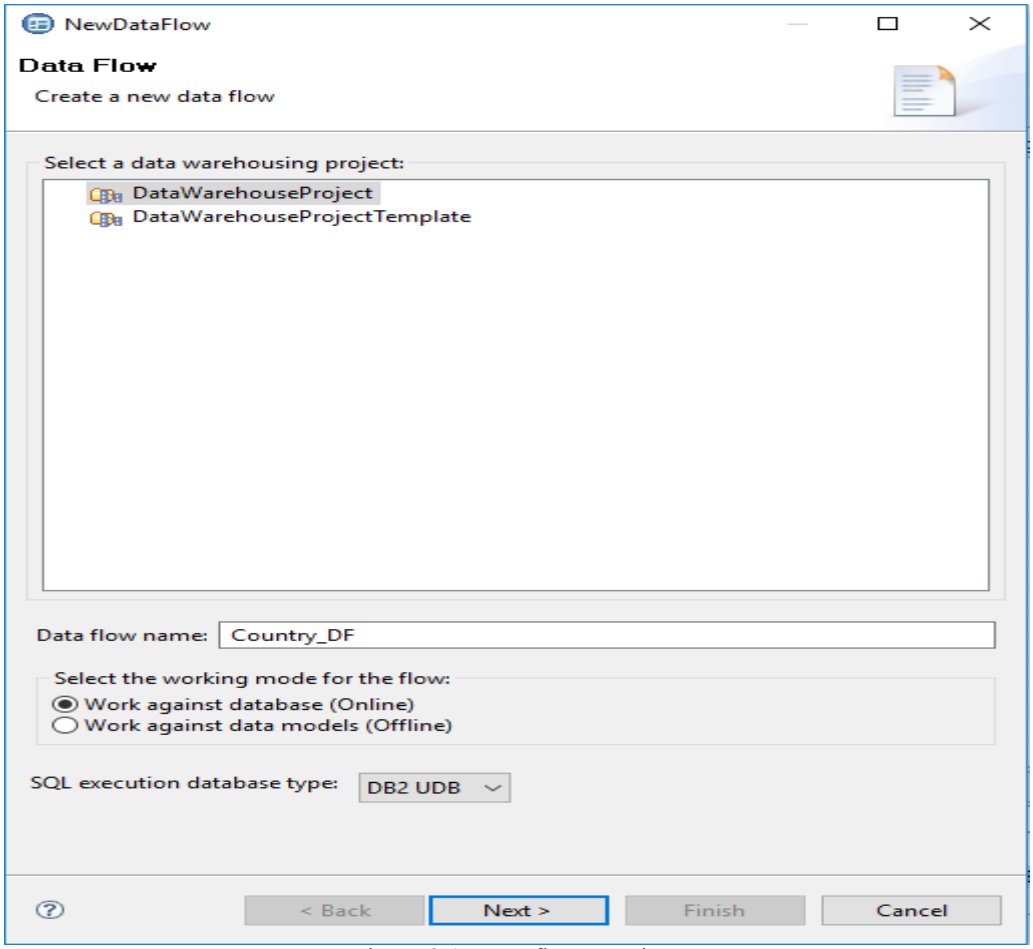

Figure 2.1: Data flow creation

Click on **Next** → Select the Connection name from where dataflow has to pick the data in **Select Connection window→ Finish.** The window will appear as shown in Figure 2.2

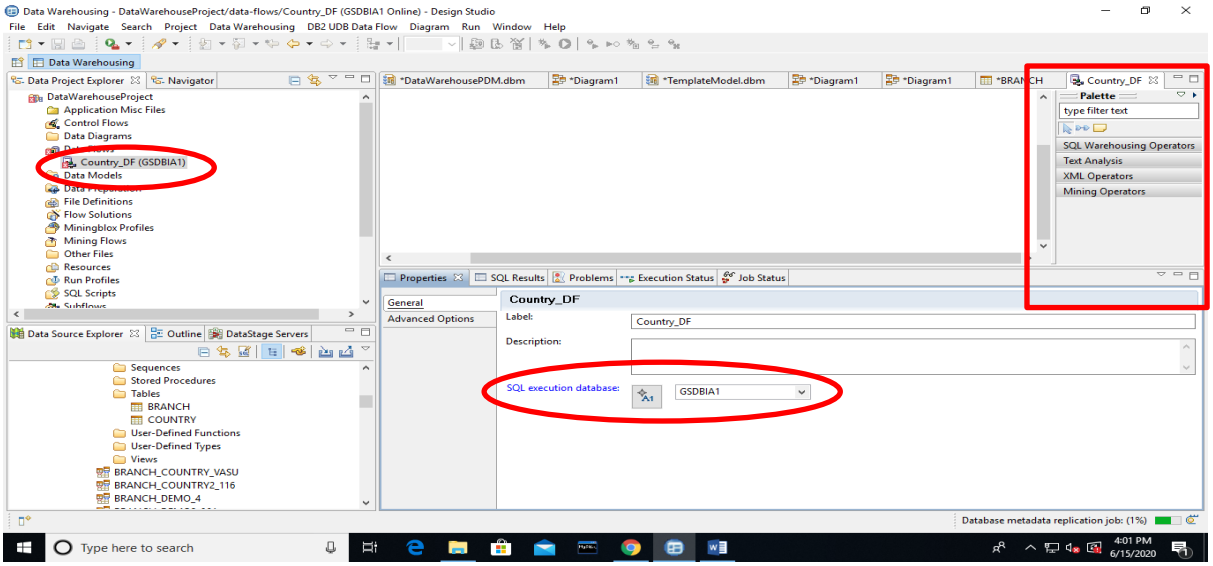

Figure 2.2: After creation of dataflow

The palette seen on the right top corner contains SQL warehousing operators which is needed to construct the data flow. Figure 2.3 shows the SQL operators, on expanding the SQL warehousing operators.

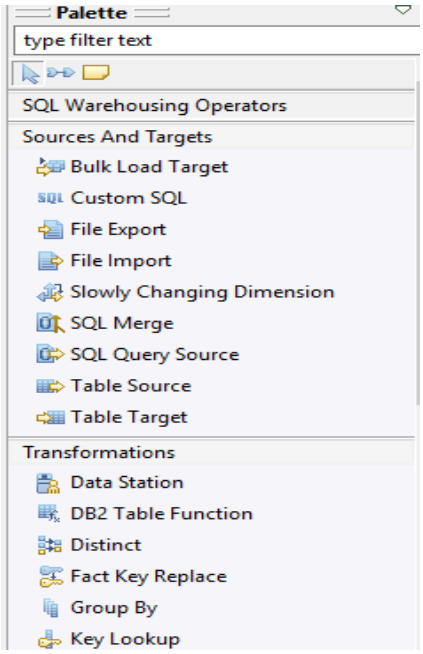

Figure 2.3: SQL warehousing operators.

The query to populate data into BRANCH_COUNTRY_SCHEMA. COUNTRY table from GOSALES. COUNTRY table is as follows:

```
Select gs.BRANCH_CODE, gs.ADDRESS1, gs.COUNTRY_CODE
FROM GOSALES.COUNTRY as gs.
```

Hence, one **Table source** operator to hold the data of GOSALES.COUNTRY table, **Select list** operator to select the required attributes and **Table target** operator to hold the resultant data is required. All these operators should be dragged and dropped on the canvas and connections should be made between the operators. Figure 2.4 shows the data flow to populate country table of BRANCH_COUNTRY_SCHEMA.

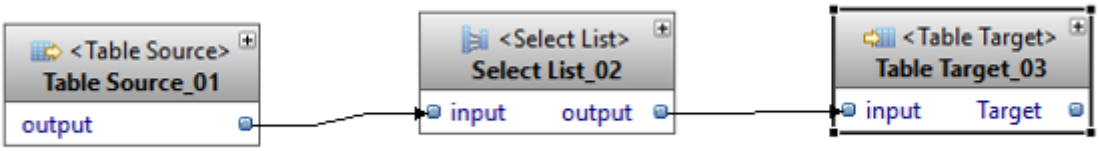

Figure 2.4: Data flow to populate BRANCH_COUNTRY_SCHEMA. COUNTRY

Configurations for Table Source_01:
Double click on the operator, the window shown in Figure 2.5 will appear.

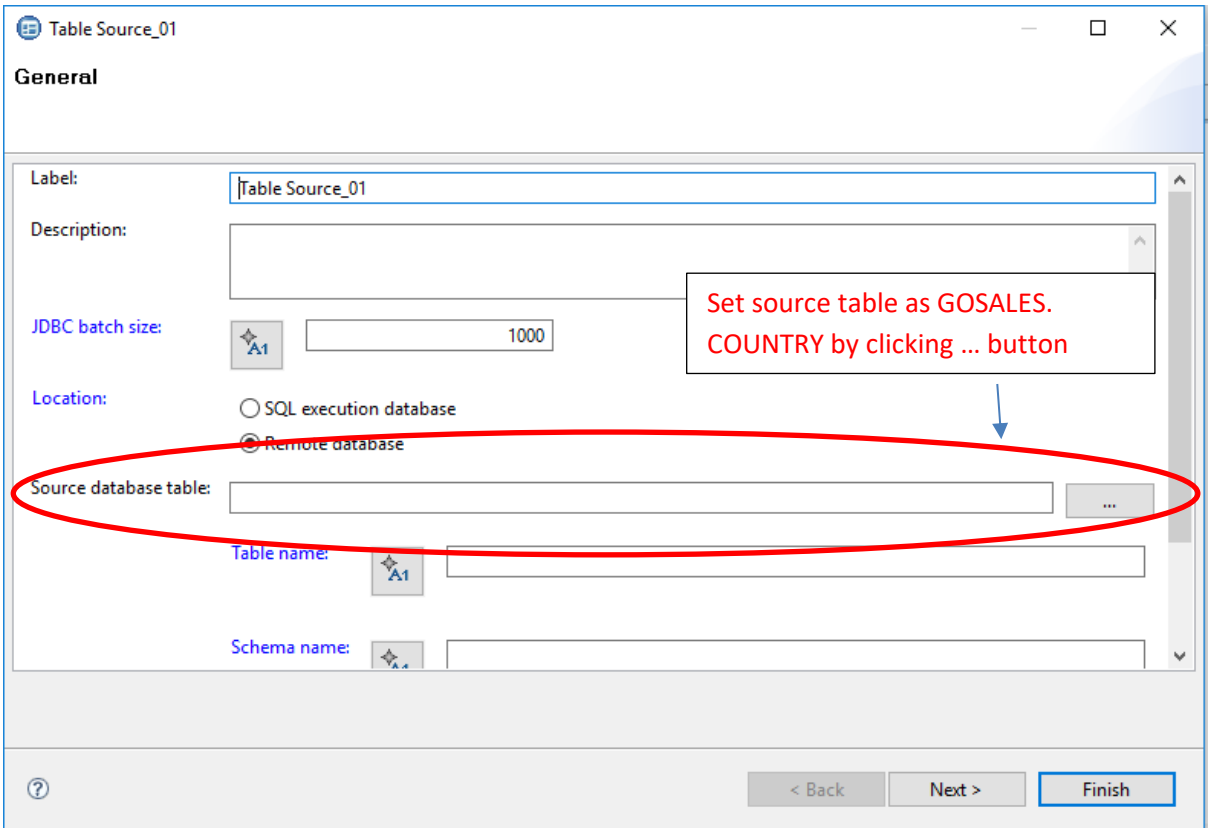

Figure 2.5: Setting source table for data flow

Select **SQL execution database**. On click of **…** button, the window shown in Figure 2.6 appears.

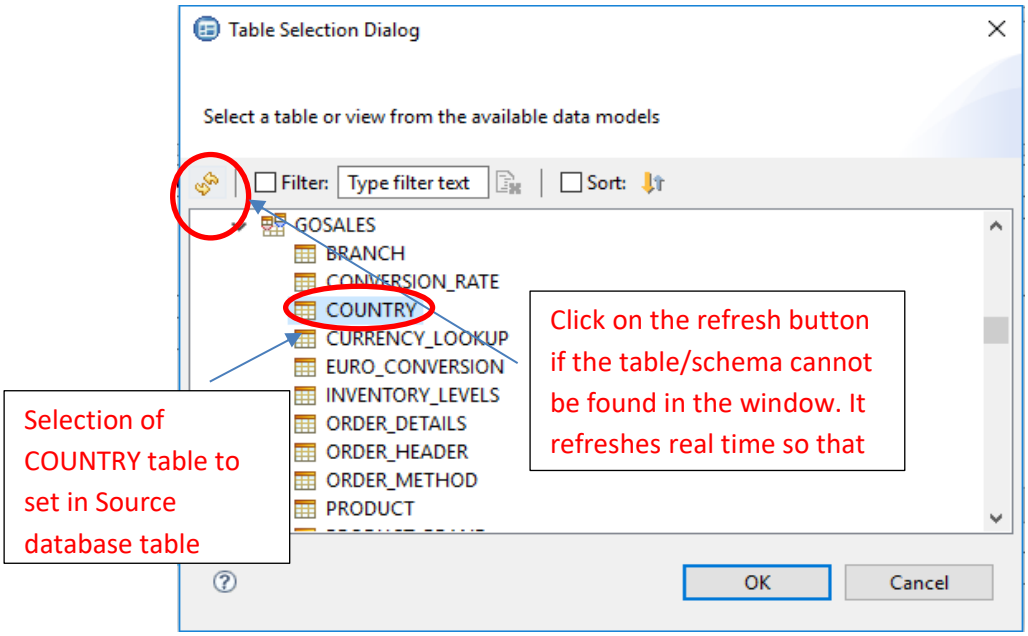

Figure 2.6: Setting source table for data flow

Once the GOSALES.COUNTRY table is set as source, Click on **finish**.

Configurations for Select List_02:

Double click on Select List_02, following figures shows sequence of steps to be followed:.

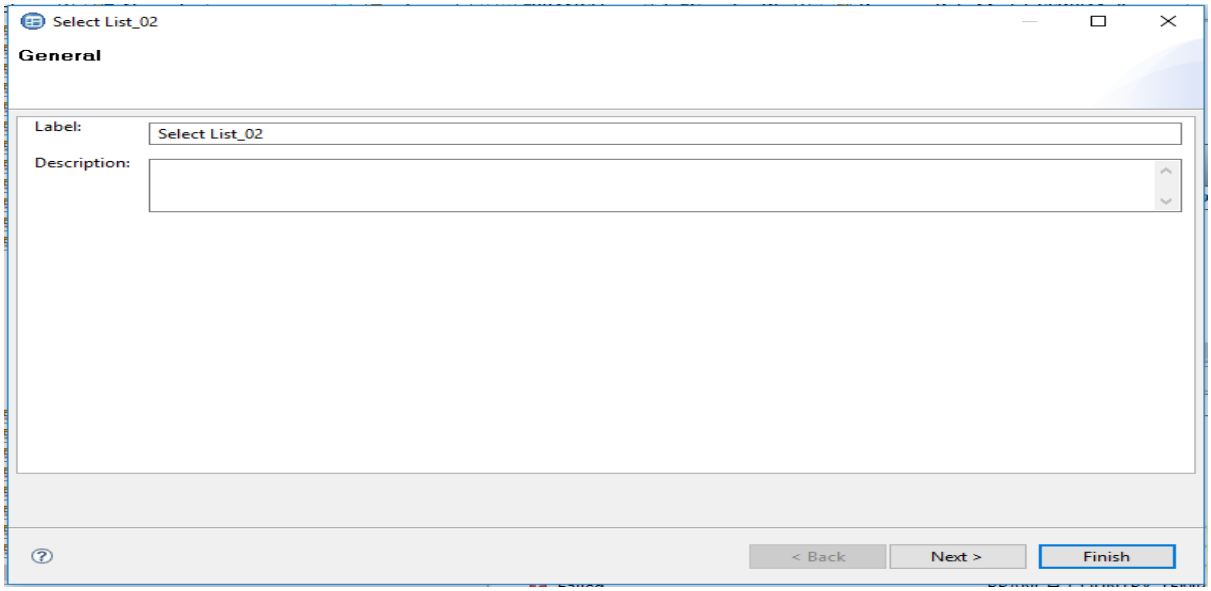

Figure 2.7: Configurations for Select List operator

Click on **Next.** In the **Select List** window, under **result** columns retain only those attributes which is required in Target table i.e BRANCH_COUNTRY_SCHEMA. COUNTRY by deleting all other attributes as shown in Figure 2.8

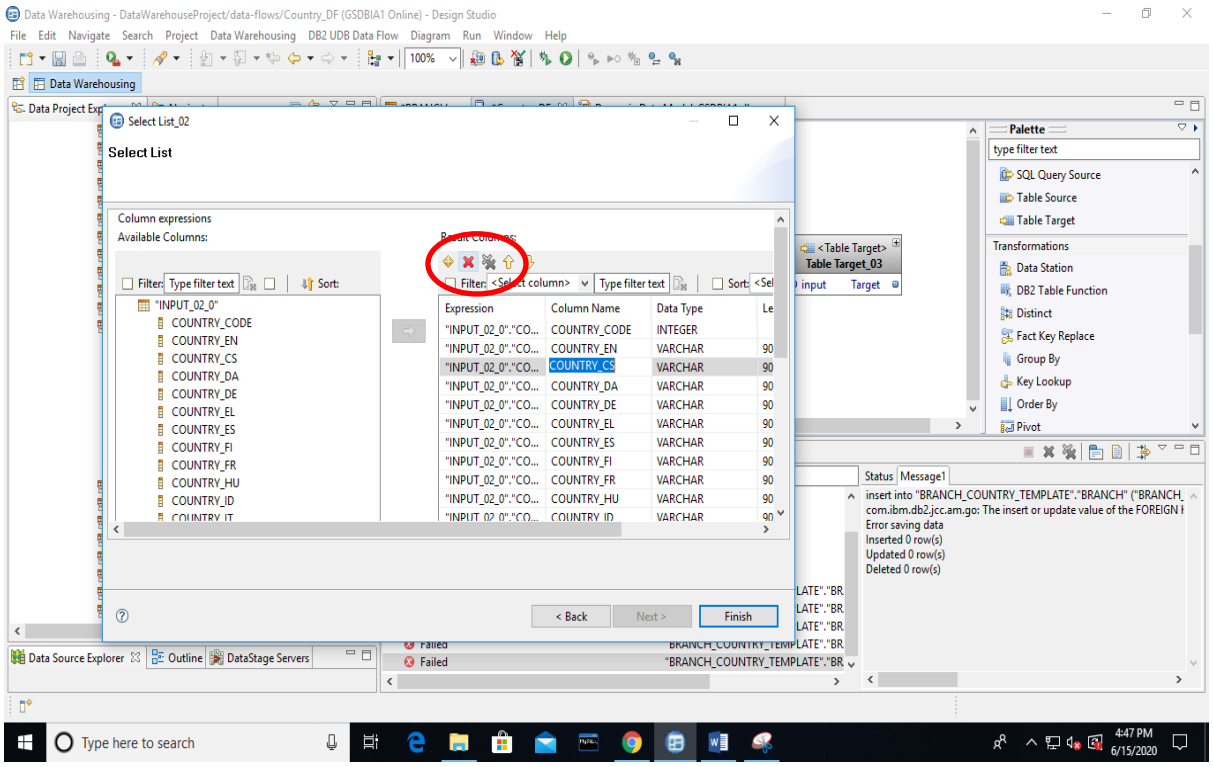

Figure 2.8 shows only the required attributes after deletion of other attributes

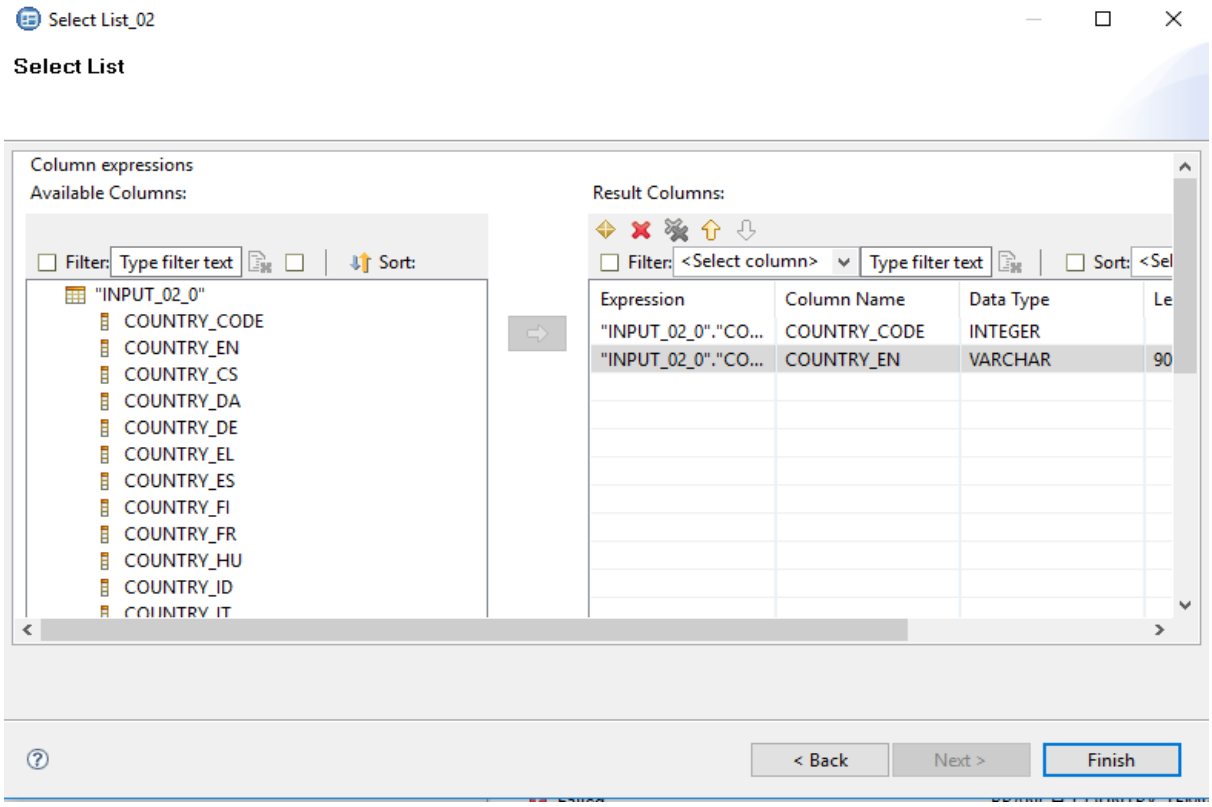

Figure 2.9 shows only the required attributes after deletion of other attributes

Click on **Finish.** Follow the same procedure for Target Table configurations as shown for source table. After setting up all the configurations for Target Table, mapping of attributes to their input ports have to be verified. Figure 2.10 shows window where attributes are mapped to their respective input port.

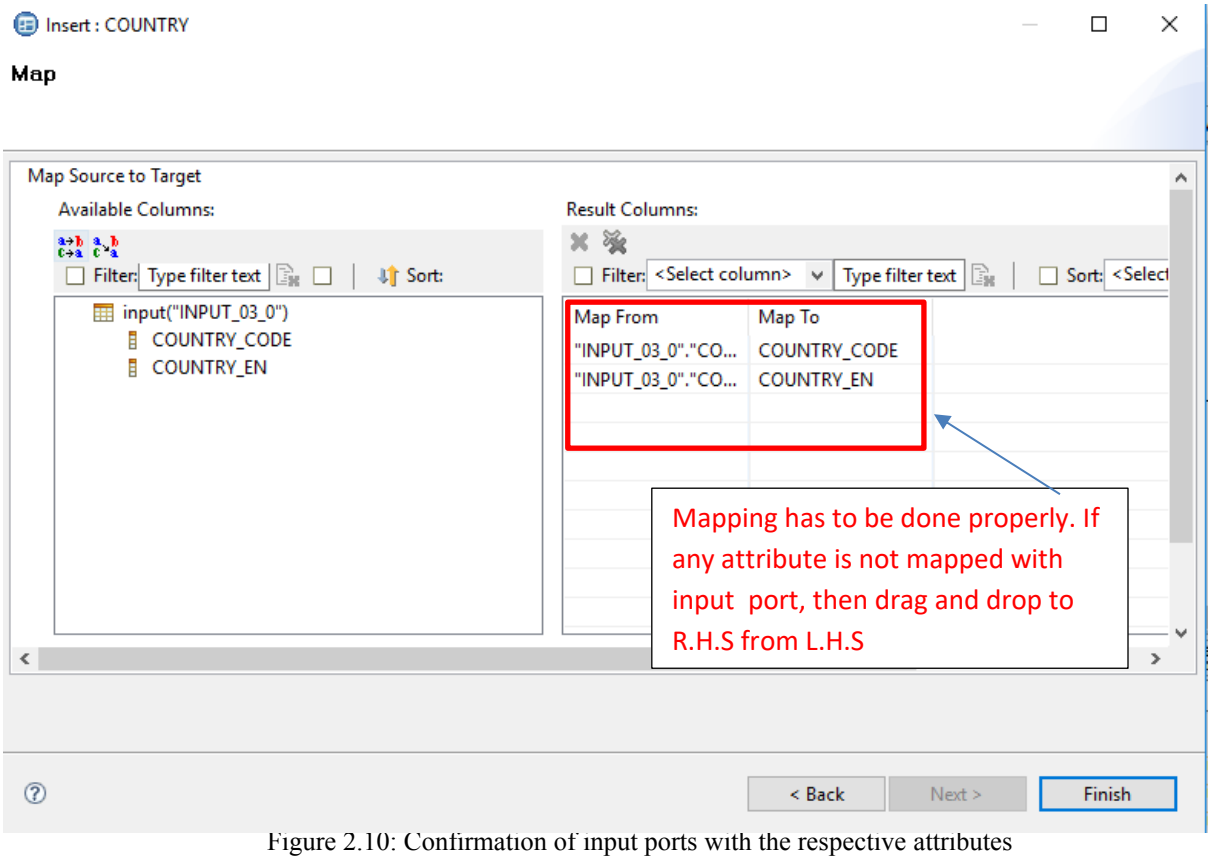

Figure 2.10: Confirmation of input ports with the respective attributes

Figure 2.11 shows final data flow after setting up of all configurations in the respective operators.

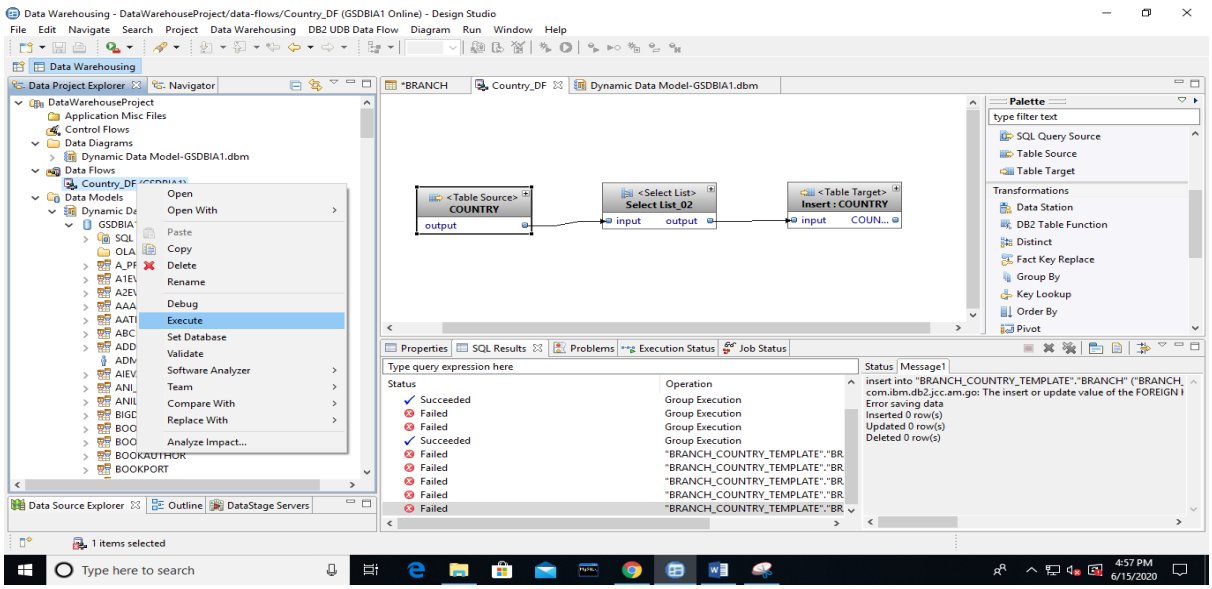

Figure 2.11: Final data flow to load COUNTRY table

Once the data flow is ready, it can be executed by clicking the green execute button on the top as shown in Figure 2.12

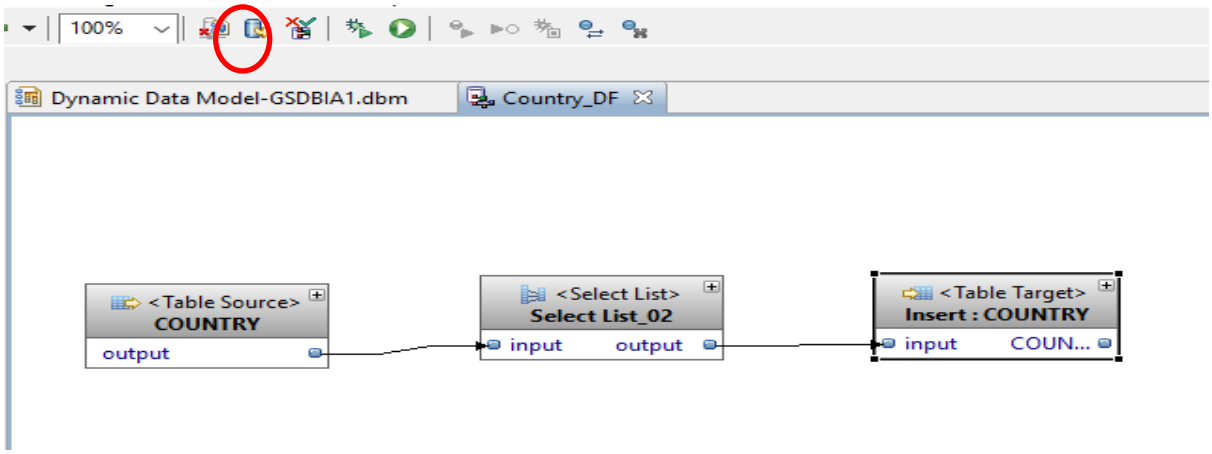

Figure 2.12: To execute data flow

The contents loaded into the BRANCH_COUNTRY_SCHEMA. COUNTRY table can be viewed through **Sample Contents** option on the table on server side.

Similarly, load the data for BRANCH table. The data for COUNTRY table is loaded first here is because BRANCH refers to COUNTRY with COUNTRY_CODE. Therefore, COUNTRY table has to be loaded first.

Natural Join of Two Tables

Here, Table Join based on a joining condition is explored. The process is learnt by join of BRANCH and COUNTRY using Country_code as the attribute to join. Since there is no table to hold the result of Join, a new table to hold the result of join needs to be created. Let JOIN_RESULT be the name of the new target table.

A new table can be created inside the already existing schema i.e BRANCH_COUNTRY_SCHEMA and only generate DDL for JOIN_RESULT. Figure 2.13 shows creation of JOIN_RESULT.

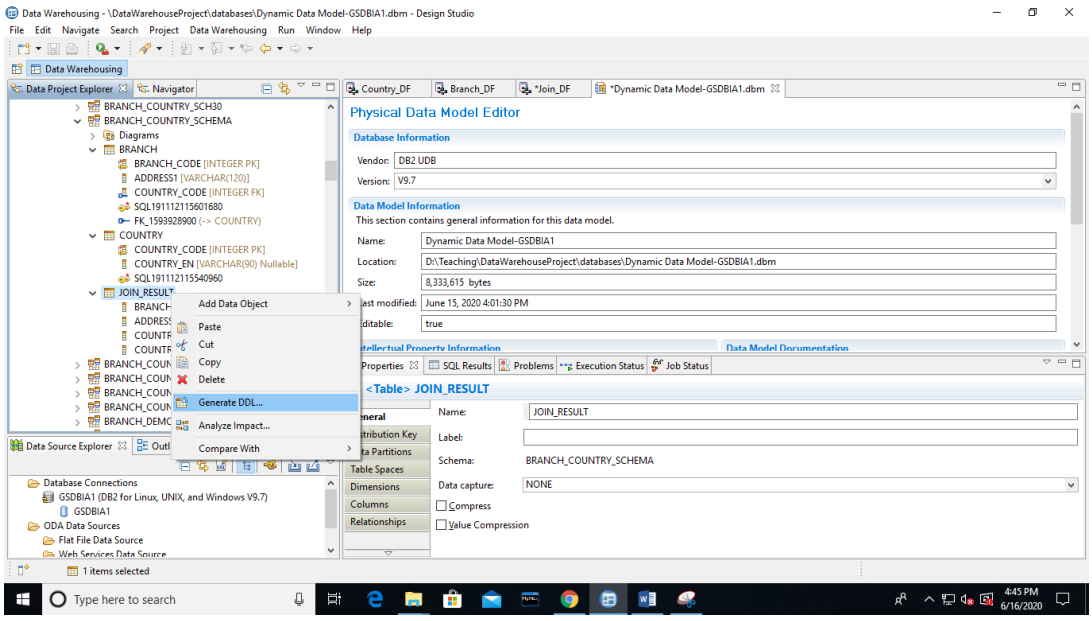

Figure 2.13: Generation of JOIN_RESULT table

The dataflow for Table Join is shown in Figure 2.14

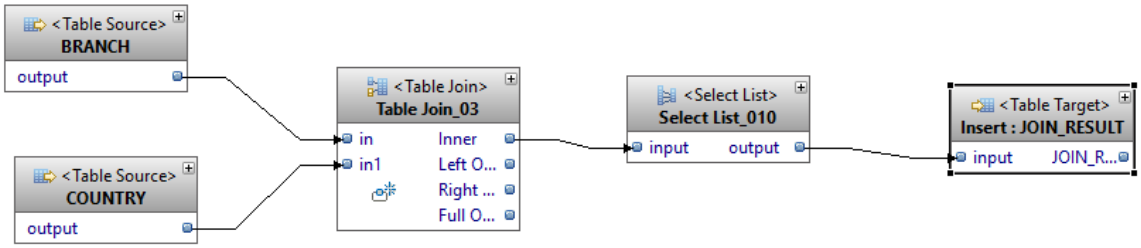

Figure 2.14: Data flow for Joining BRANCH and COUNTRY table

Table sources should be set as shown in the above sections.

Double click on the Table join operator and the window in Figure 2.15 will appear.

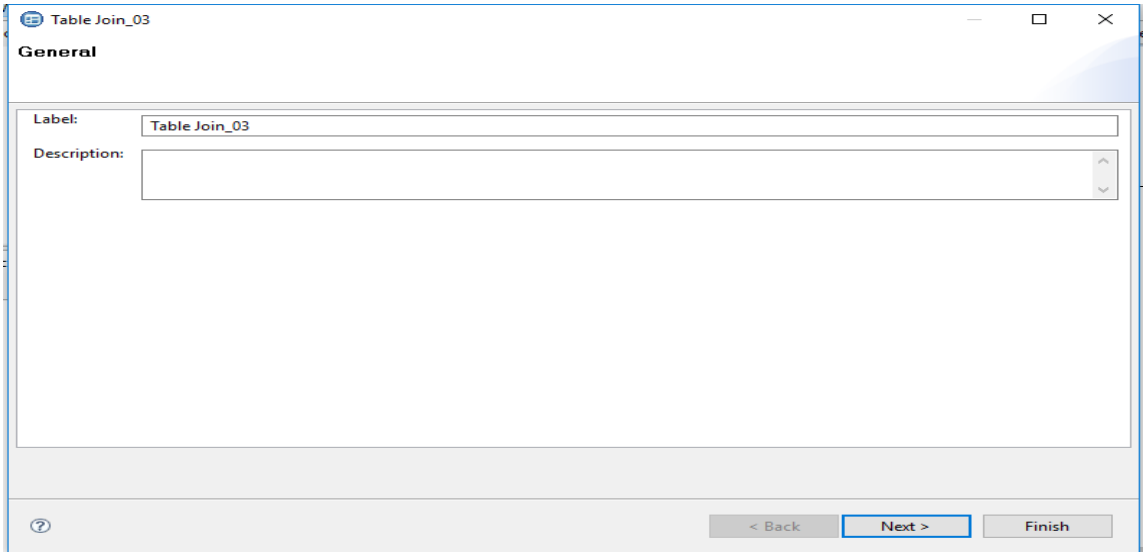

Figure 2.15: Configurations for Table Join operator

Click on **Next.** Figure 2.16 shows the window where joining condition appears after it has been set.

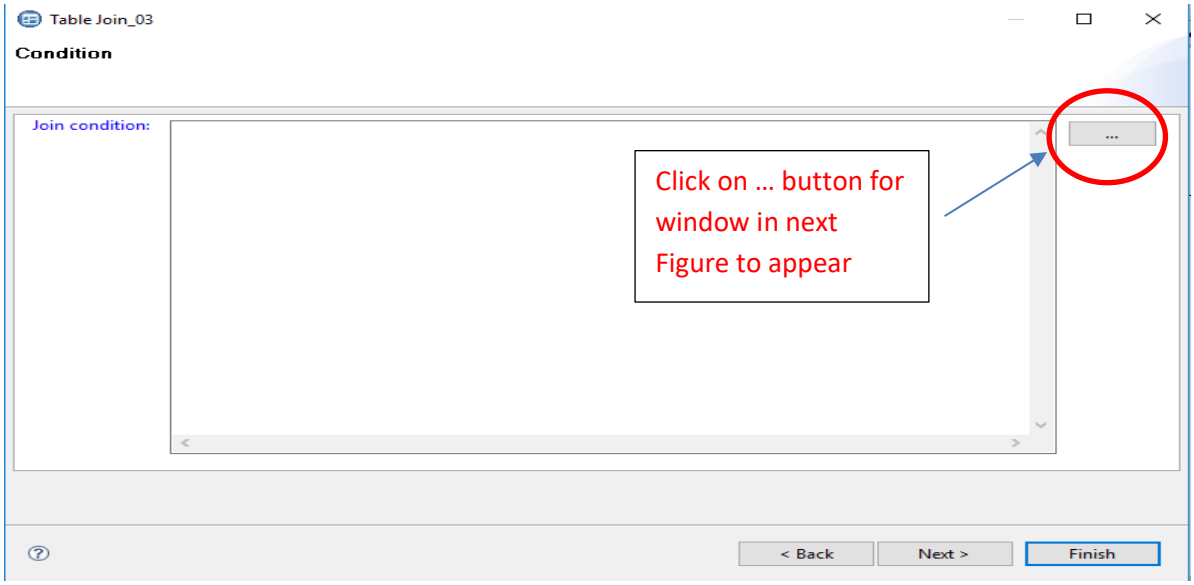

Figure 2.16: Window where joining condition appears

Figure 2.17 shows window where condition has to be set.

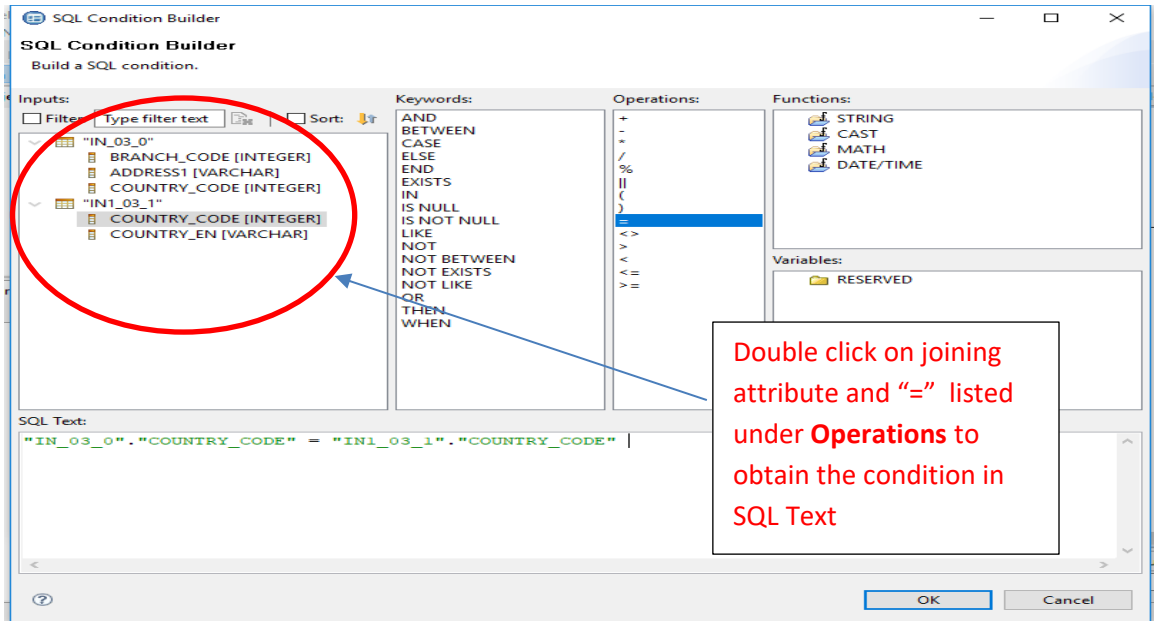

Figure 2.17: Window where joining condition is set

Click on **OK**. Figure 2.18 shows the configuration for Select List operator

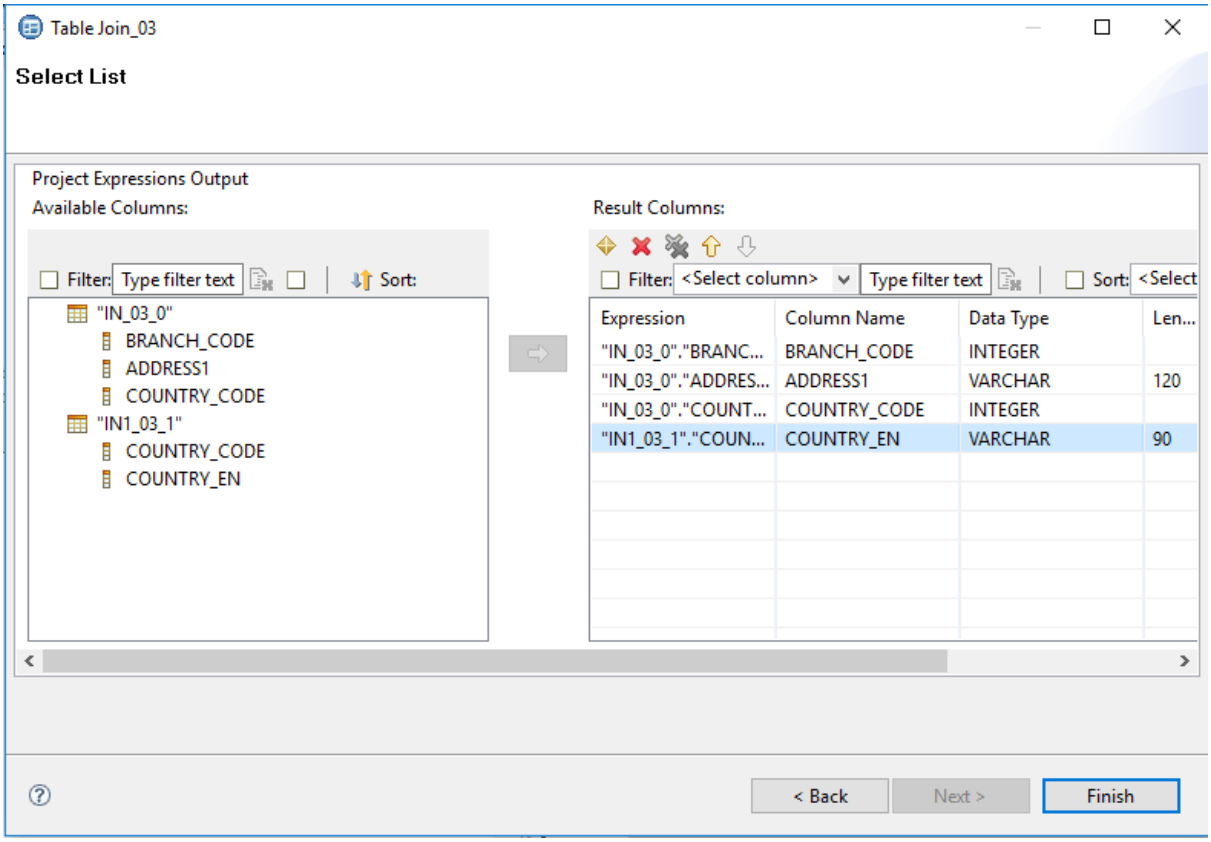

Figure 2.18: Configuration settings for Select List

Once the data flow is complete, it can be executed and result of the execution can be viewed in JOIN_RESULT table on server side.

Where Condition Operator

Design studio offers **Where Conditional Operator** to apply condition on individual tuples. Figure 2.19 shows **where operator.**

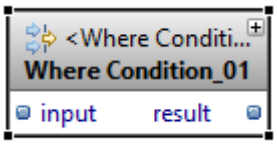

Figure 2.19: Where operator

Double click on the **Where operator** for the window in Figure 2.20 to appear, where description about the condition being set is provided. Figure 2.21, 4.22 provides the configuration settings for **Where Operator.**

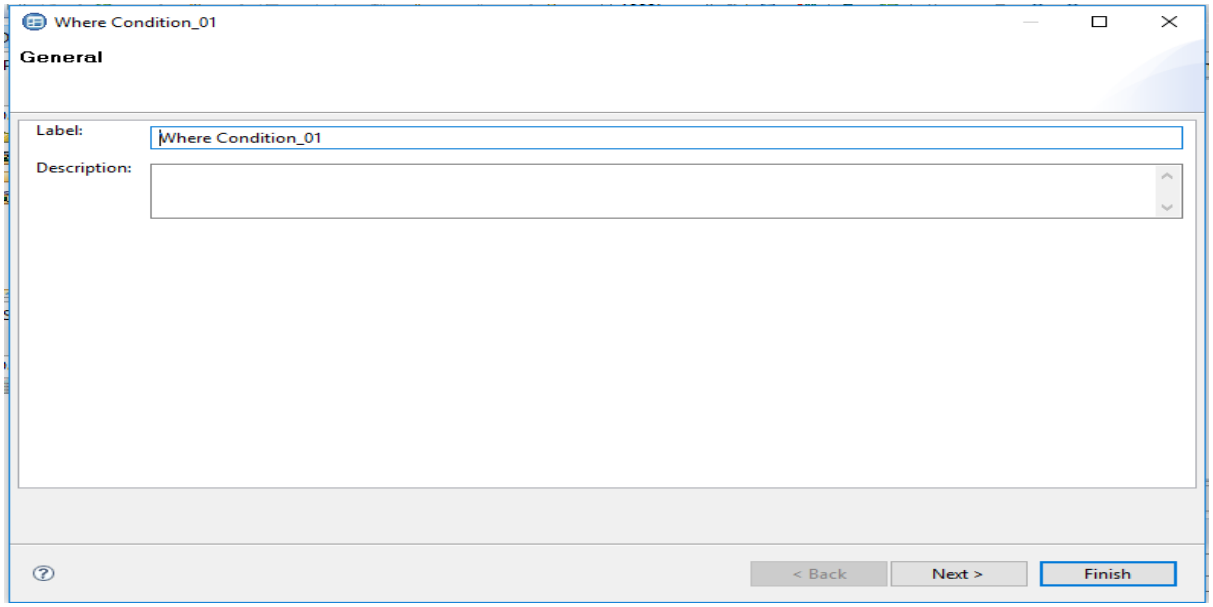

Figure 2.20: Window to provide description about condition being set.

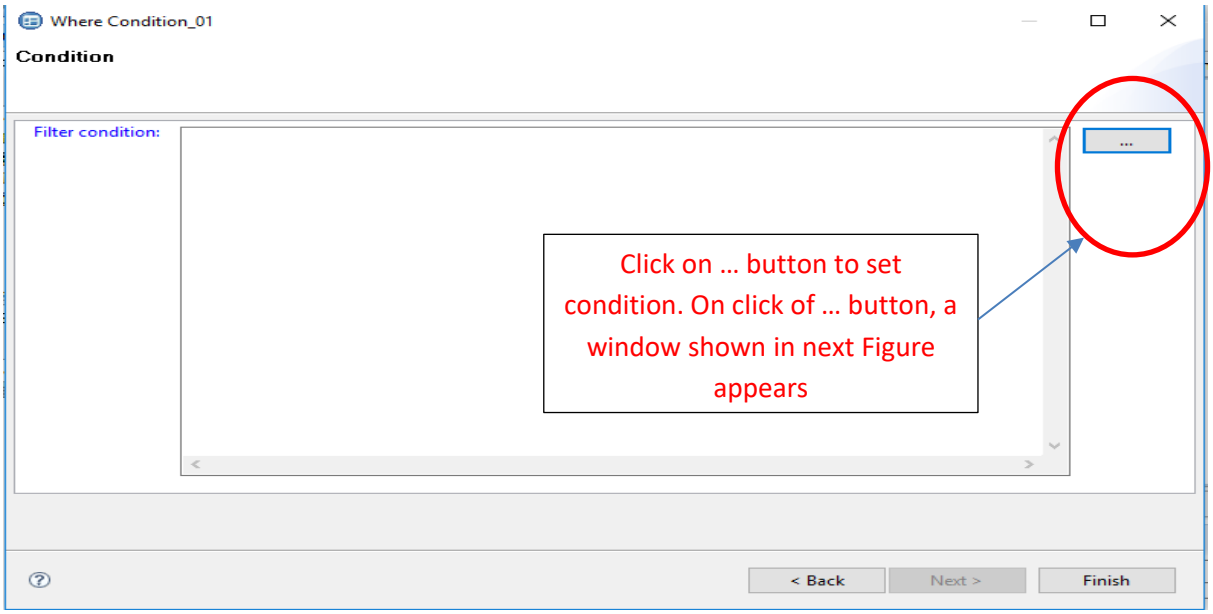

Figure 2.21: Configurations for Where Operator.

In Figure 2.22, Under **Input** field, various attributes will appear on **Where operator's** connection with table source. Choose attributes on which condition has to be set. The set condition will appear in **SQL Text field**.

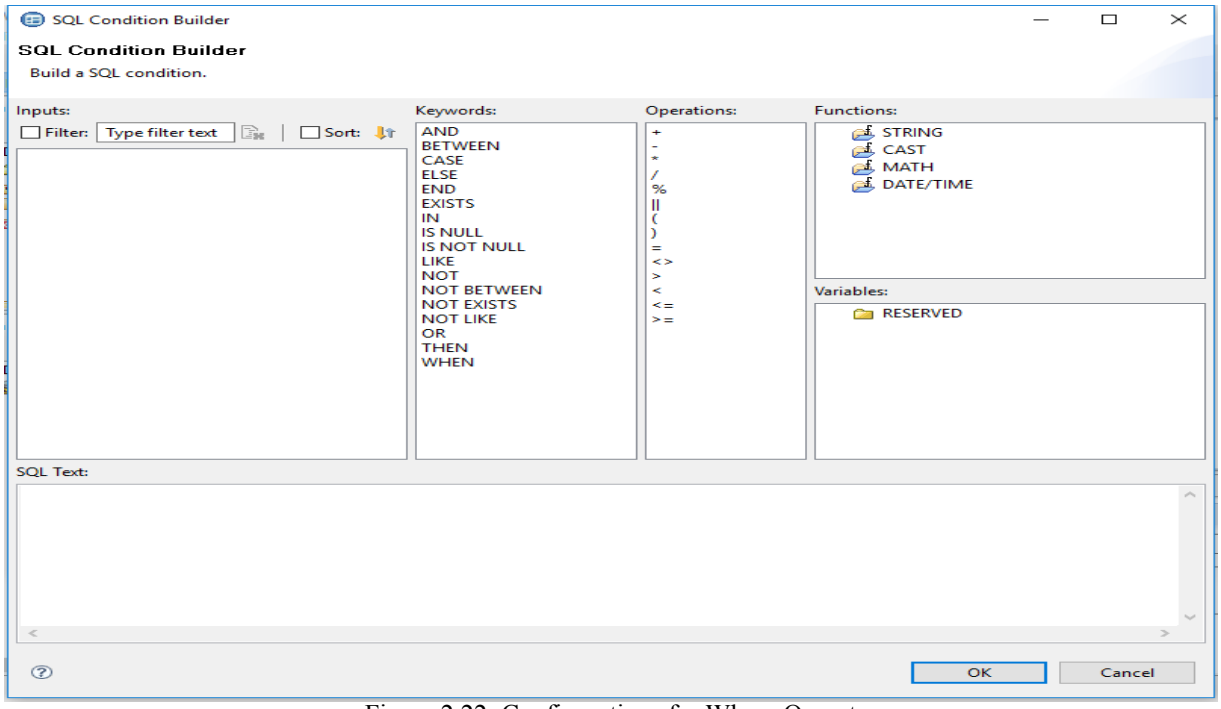

Figure 2.22: Configurations for Where Operator.

Group by Operator and Having Clause

Group By Operator is found under SQL Warehousing operator. The Group By operator in turn has **Having clause** in it. Figure 2.23 Shows the Group By operator.

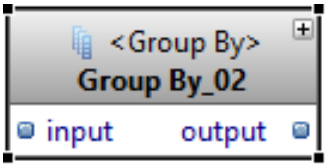

Figure 2.23: Group By operator

On double click of Group By operator window in Figure 2.24 will appear where description about group by operation can be given.

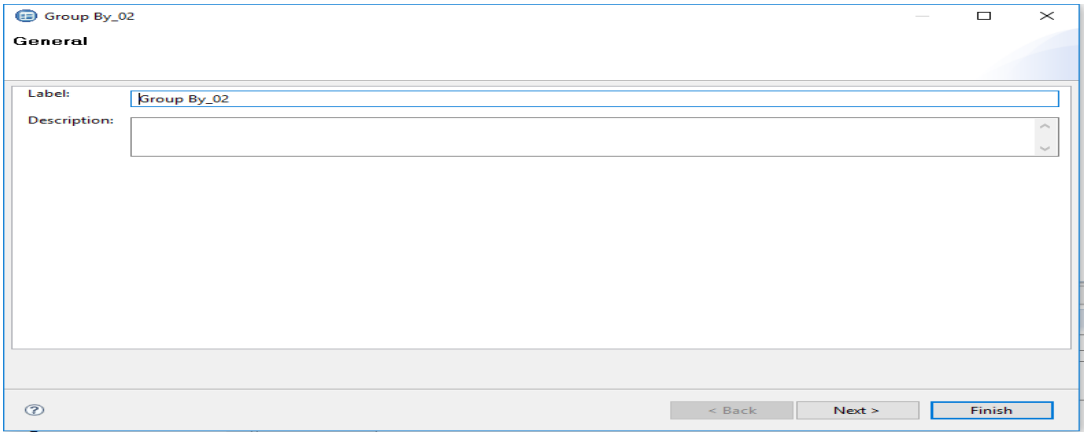

Figure 2.24: Description window for Group By operator

On click of **Next,** window in Figure 2.25 will appear where attributes which have to appear in select list must be listed under Result Columns. Click on **…** button for SQL Expression Builder window (Figure 2.25) to apply aggregate function on any attribute in select list. Figure 2.26 shows application of aggregate function.

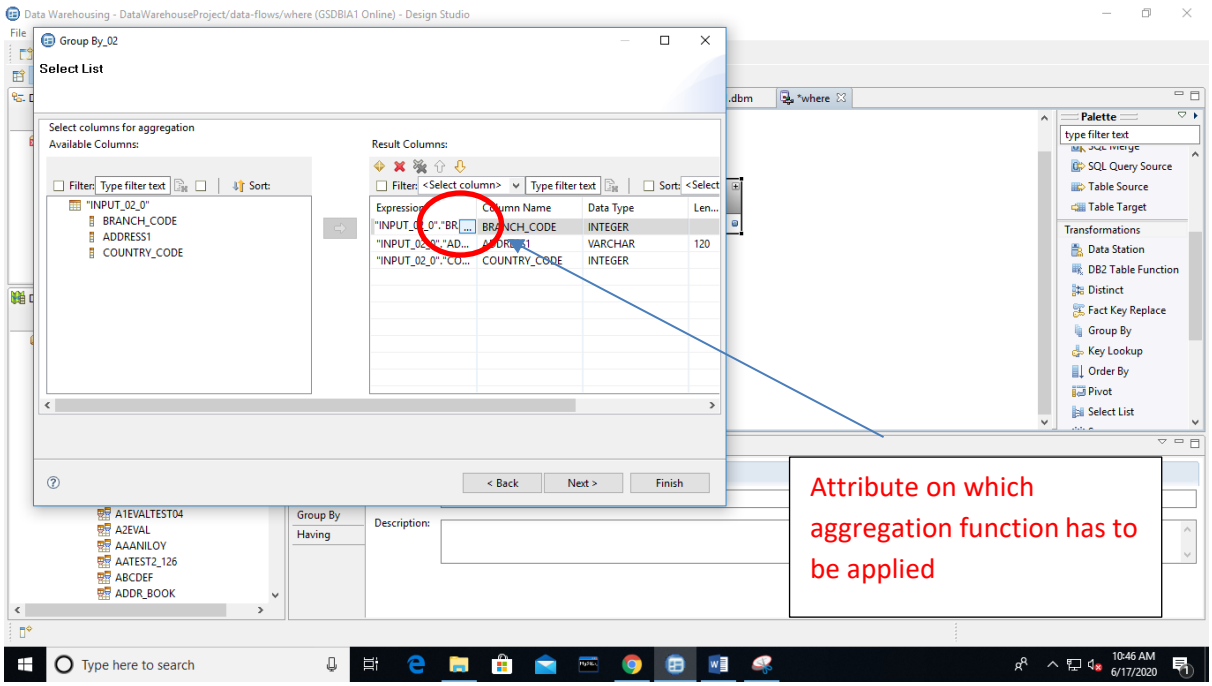

Figure 2.25: Configurations for Group By Operator in Select List window

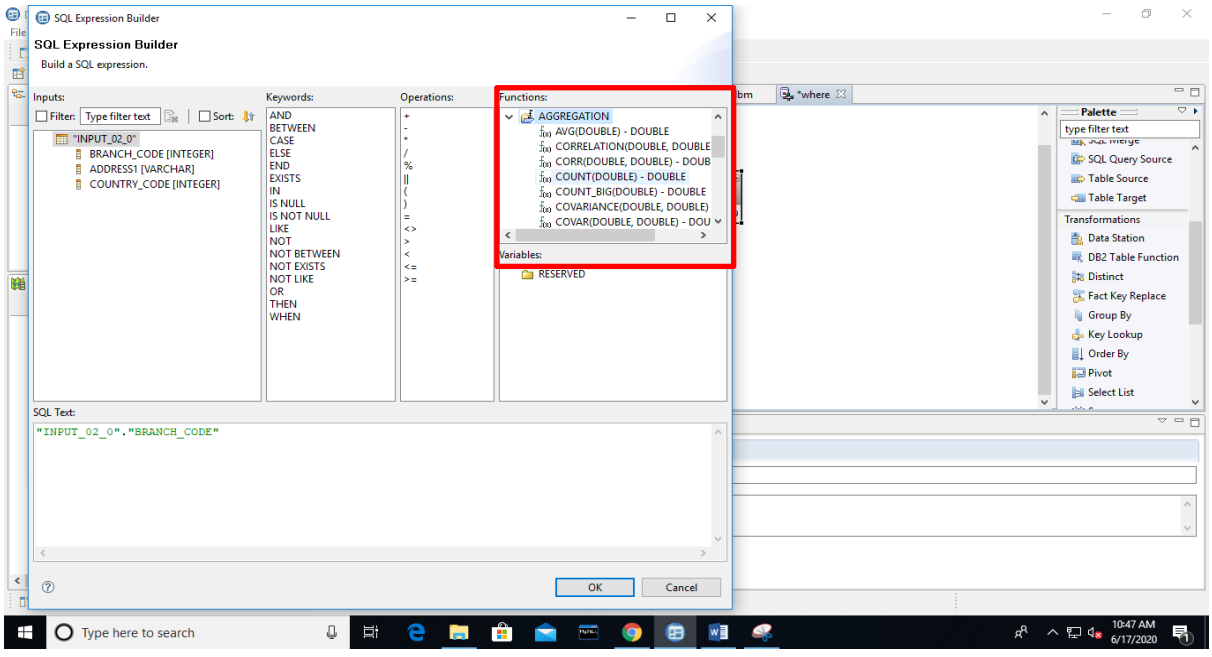

Figure 2.26: Configurations for Group By Operator to apply aggregate functions

On click of **Aggregations** under **Functions**, gives many functions to build SQL expressions that appears in select list as shown in Figure 2.27.

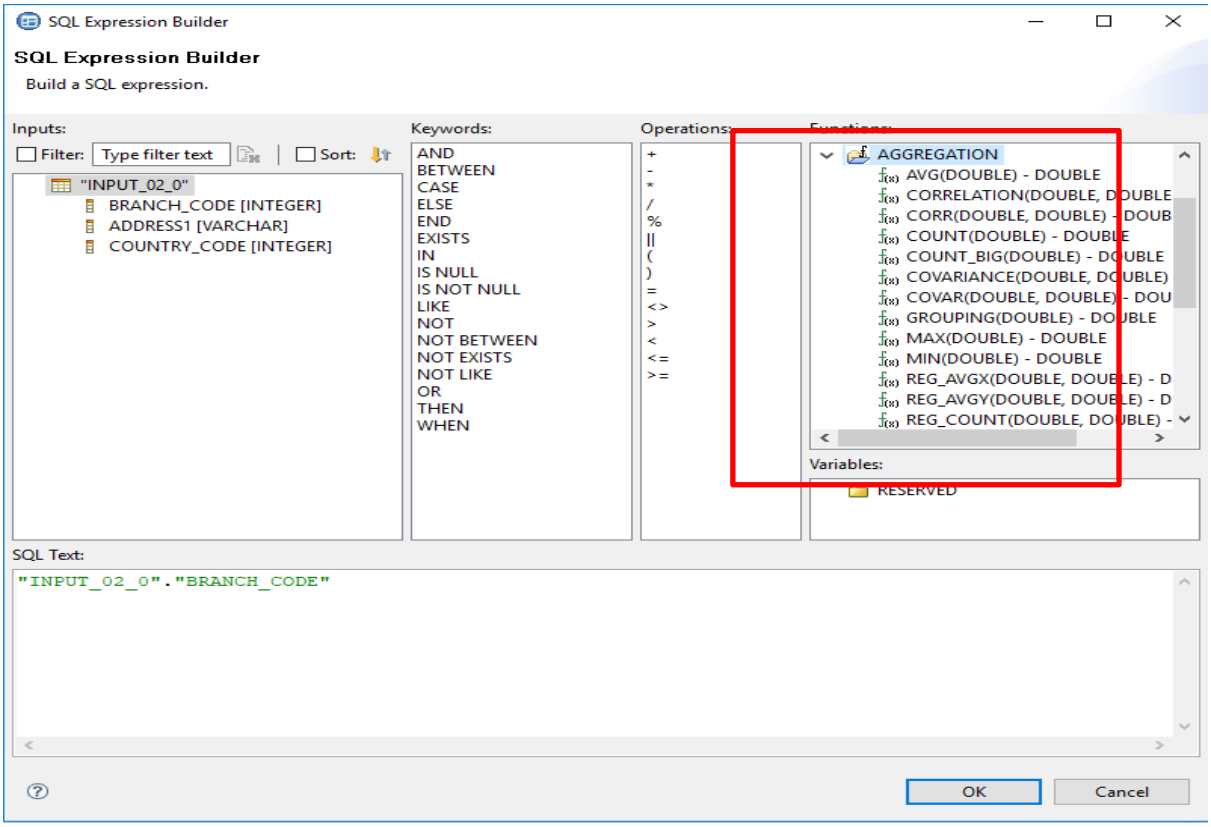

Figure 2.27: Aggregation functions in Group By operator

The attribute on which Group By has to be applied must be mentioned in Group By window as shown in Figure 2.28.

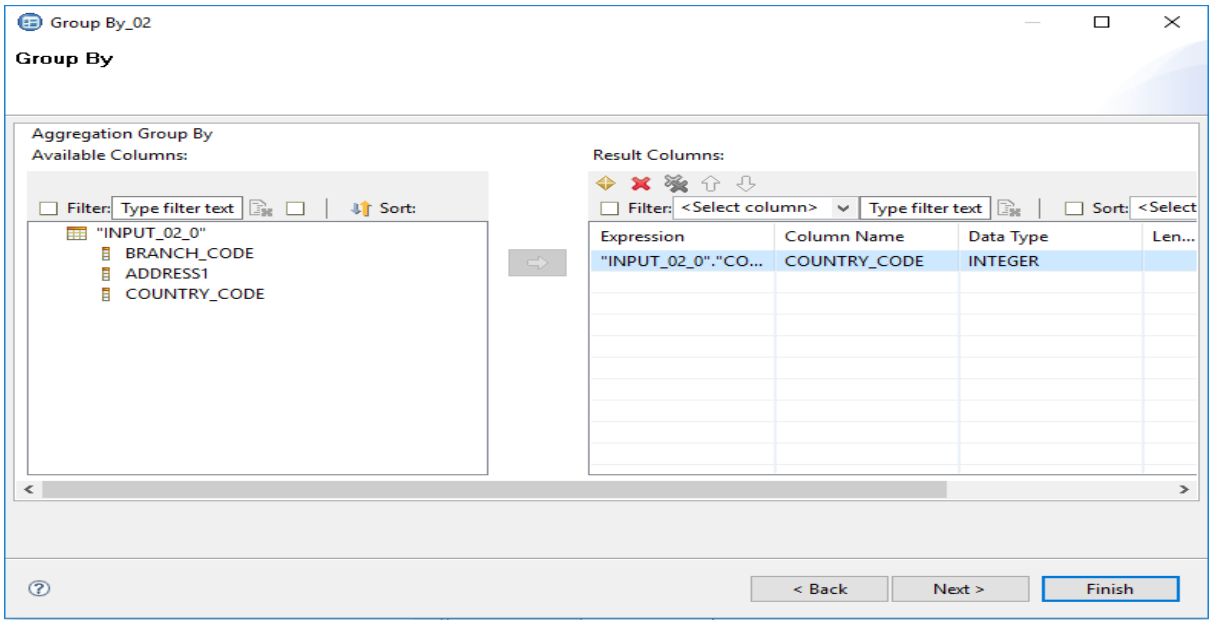

Figure 2.28: Group By window in Group By operator

If at all, any condition has to be given on a group of tuples, then such conditions can be set in **Having clause** as shown in Figure 2.29**.**

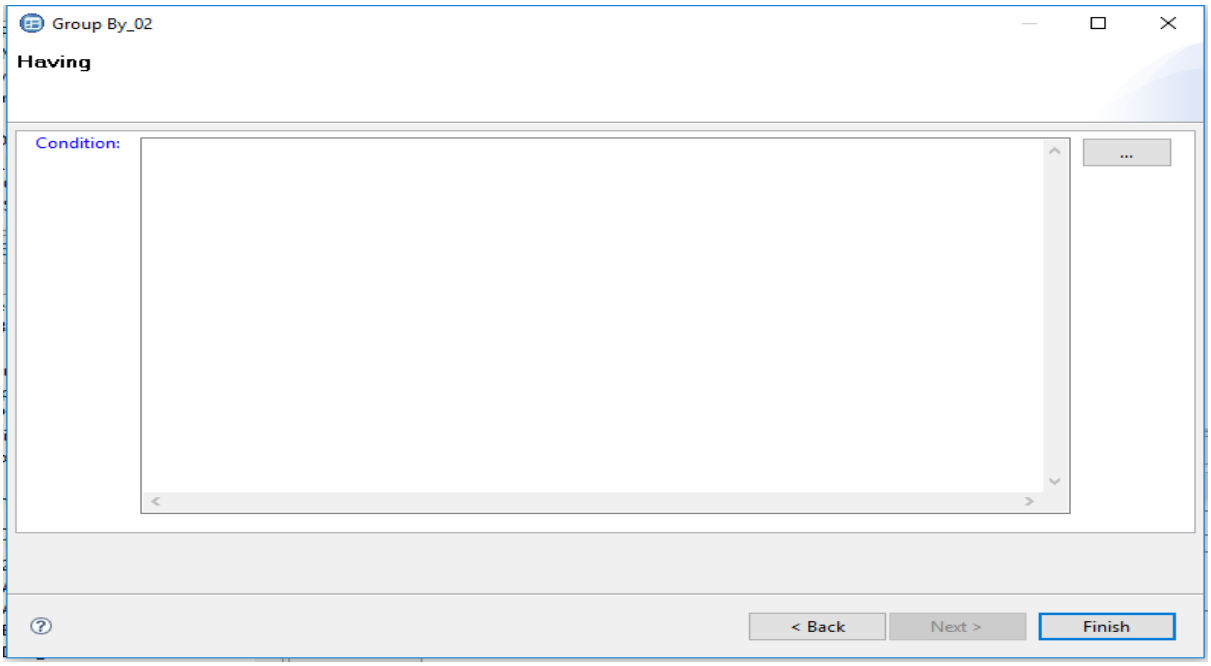

Figure 2.29: Having clause window in Group By operator

Apart from the operator mentioned, there are many other SQL operators such as Union, Distinct, Order By which may be explored by Students.

Exercises

Execute the following SQL queries using data flows in Infosphere

1.Product(pid,pname)
Supplier(sid,sname)
Supply(pid,sid,qty)

i. Display the name of the product with Product ID = 3266
ii. Display the product ID of 'Eureka Forbes vacuum' and 'Aquaguard water purifier'.
iii. Display the supplier name of the product with the ID = 3266.
iv. Display the quantity of 'Eureka Forbes vacuum' products sold by the seller 'Coud9'
v. Find the PID of products whose names start with 'AI'
vi. Find number of sellers for each product.
vii. Find sellers who sell more than two products.
viii. Find seller names who sell more than 100 pieces of products.

2.Student (Regno, Name, Major, Bdate)
Course (Course_ID, Cname, dept)
Enroll (RegNo, Course_ID, marks)
Book_Adoption (Course_ID, sem, ISBN)
Text (book_ISBN, title, publisher, author)

i. Display the course names taken by students of ICT department.
ii. Find whether ICT department has taken up any course which needs books with ISBN=1111.
iii. Find names of students who have scored more than 90 in the course of 'Compilers'
iv. Find the departments which provides Major.
v. Find the departments which does not use textbooks of 'Wiley' publisher.
vi. Display names of the students who have registered for courses for which more than 10 students have registered.
vii. Display the departments which uses more than 2 books of author 'Ken'
viii. Find the department which has incorporated textbook by a single publisher.
ix. Find course which has maximum number of students.
x. Find the total marks of each student in all courses.

Additional Exercise

Execute the following SQL queries using data flows in Infosphere

1.PERSON (driver _ id , name, address)
CAR (Regno, model, year)
ACCIDENT (report_number, date, location)
OWNS (driver-id, Regno)
PARTICIPATED (driver-id, Regno , report_number, damage)

i. Select registration number of cars which do not come in between the model year 2000 and 2010.
ii. Find driver names whose sum of damage amount of all his accidents is less than average
iii. damage amount of all accidents in the database.
iv. Find driver names whose sum of damage amount of all his accidents is less than average damage amount of all accidents in database.
v. Find driver ids' who have met with accident more than once but with different cars owned by him and damage amount is greater than Rs.50000.
vi. Find driver id of persons with name 'john'.

2.Employee (Name, SSN, Salary, DoJoin)
Project (Pname, Pnumber, Mgr_ssn, PAddress)
Project_Domain(Dnumber, Pno)
Domain(Dnumber, Dname, Description)
Works_On (Essn, Pno, hrs)

i. Find names of employees who have joined department between 2010 and 2015.
ii. Find the domain with more than 3 projects under it.
iii. Find employee names who work on all projects running under domain of "Banking".

LAB NO. 3 **Date**

CREATING CONTROL FLOWS WITH IBM-INFOSPHERE

Objective

1. To create and execute Control Flows for simultaneous execution of multiple data flows.

A control flow sequences and manages the flow of activities. Activities are independent units-of-work, and could be, as examples, a data flow, a database utility, an operating system script, or a stored procedure. A control flow is the unit of execution in the runtime environment. A control flow can have one or more data flow as an activity and these data flows can be organized in sequential or parallel. If two data flows do not depend on one another for the data, then they may be organized parallel in the control flow. If the data flows have a dependency, then they have to be organized in sequence so that output of one dataflow may be taken as input by another dataflow. The need for control flow is that when a business has a very complex query to run for their analytical propose, such a complex query will result in a huge data flow (if implemented in a single go) or may be a number of data flows for one single complex query. When there are multiple data flows for a single complex query, the data flows have to be run individually as when data changes in real time to get the facts and figures. This will result in overhead for the individual. Such an overhead can be overcome using control flows where in data flows can be organized sequentially or paralleled and can be executed only once to get the final result.

Working of Control Flows

To create a new control flow, (Right click) control flow folder in Data warehousing Project → New → Control Flow as shown in Figure 3.1. In **New Control Flow** window, give a name to the control flow followed by which the work bench appears as shown in Figure 3.2.

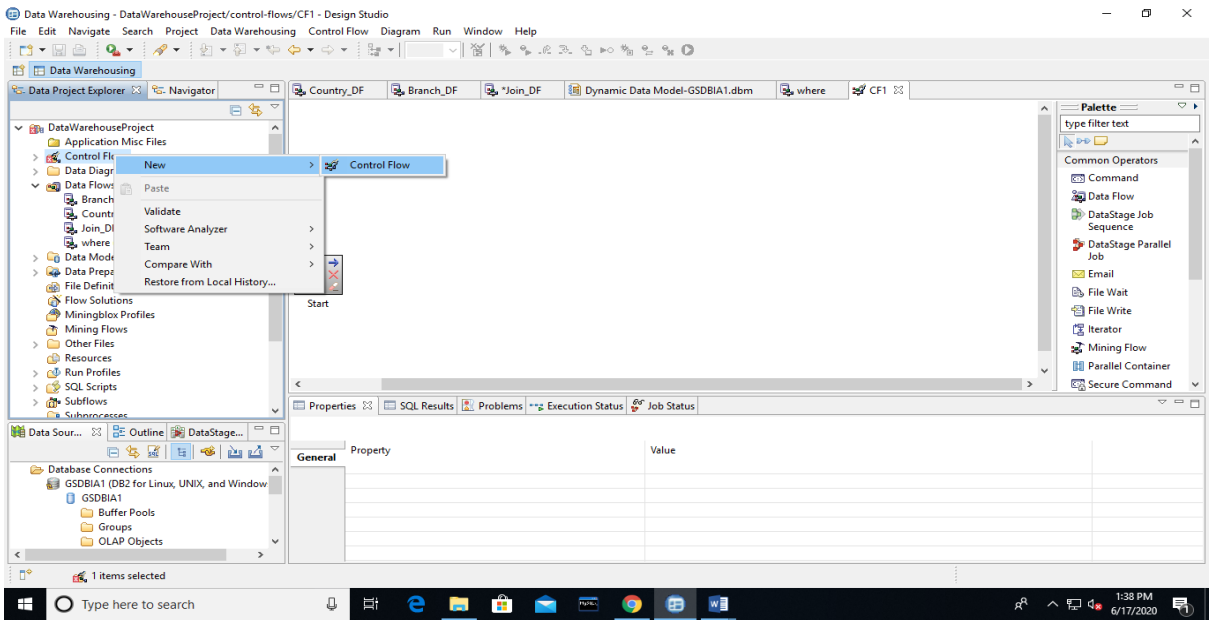

Figure 3.1: Creation of new Control Flow

As shown in Figure 3.2, on creation of a new control flow, there appears a **start icon** which has to be connected to the activities. The Palette on right provides various control flow operators.

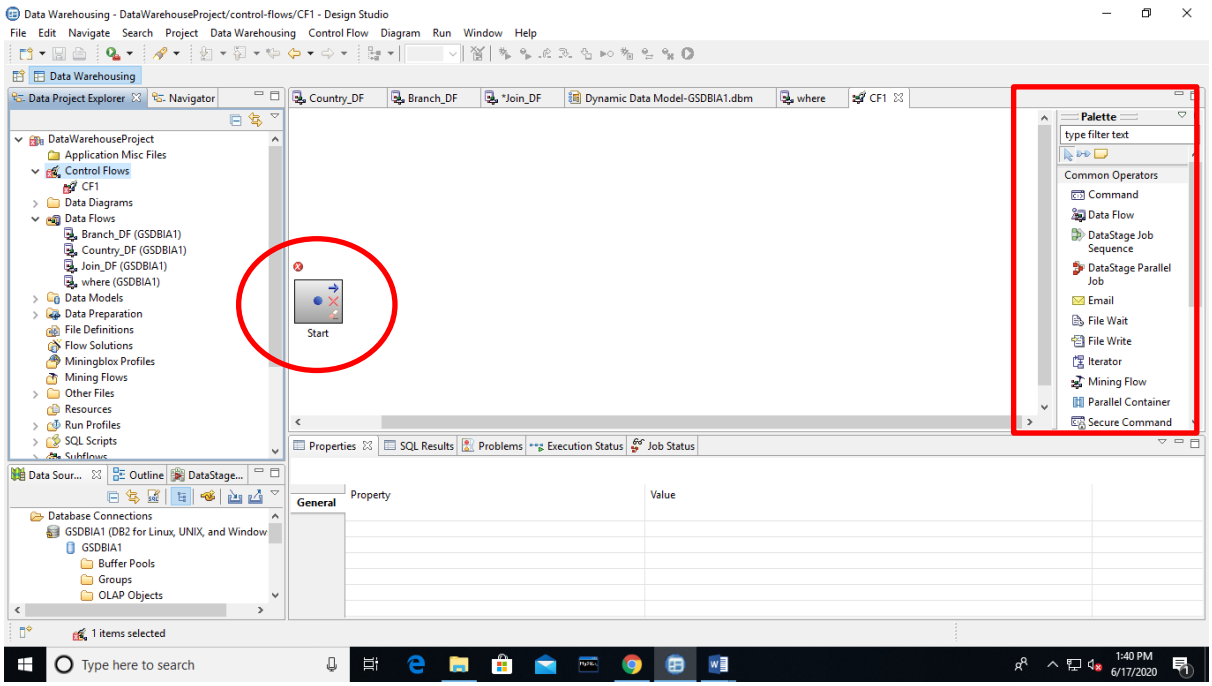

Figure 3.2: Work bench after creating a new control flow

Control flow using Data Flows Sequentially

Considering the data flows constructed to populate BRANCH and COUNTRY tables, Figure 3.3 shows the control flow to load BRANCH and COUNTRY table with data. The control flow has two data flows i.e data flow for BRANCH and data flow for COUNTRY. Since there is a dependency between COUNTRY and BRANCH through foreign key, the data flows need to be organized sequentially. The data flow for COUNTRY is set first followed by data flow for BRANCH.

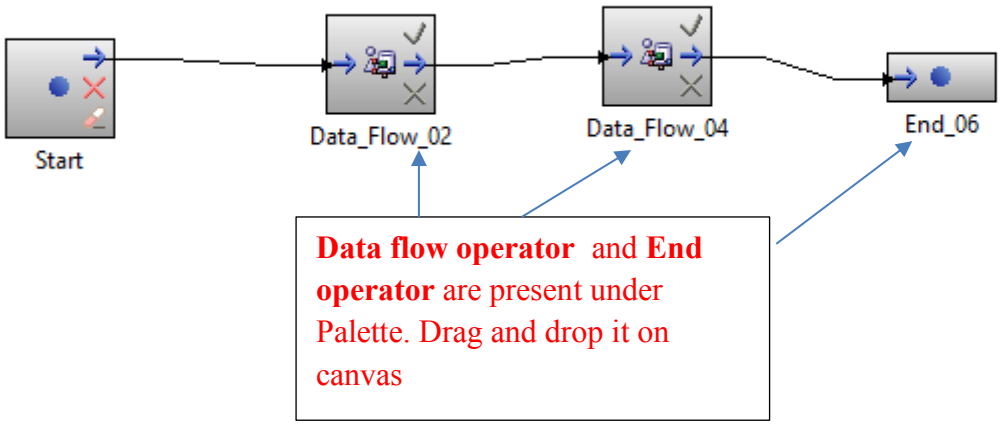

Figure 3.3: Control flow (Sequential) to populate BRANCH and COUNTRY

Click on Data_Flow_02 to set the data flow for it. On click of Data_Flow_02, the properties tab will appear as shown in Figure 3.4.

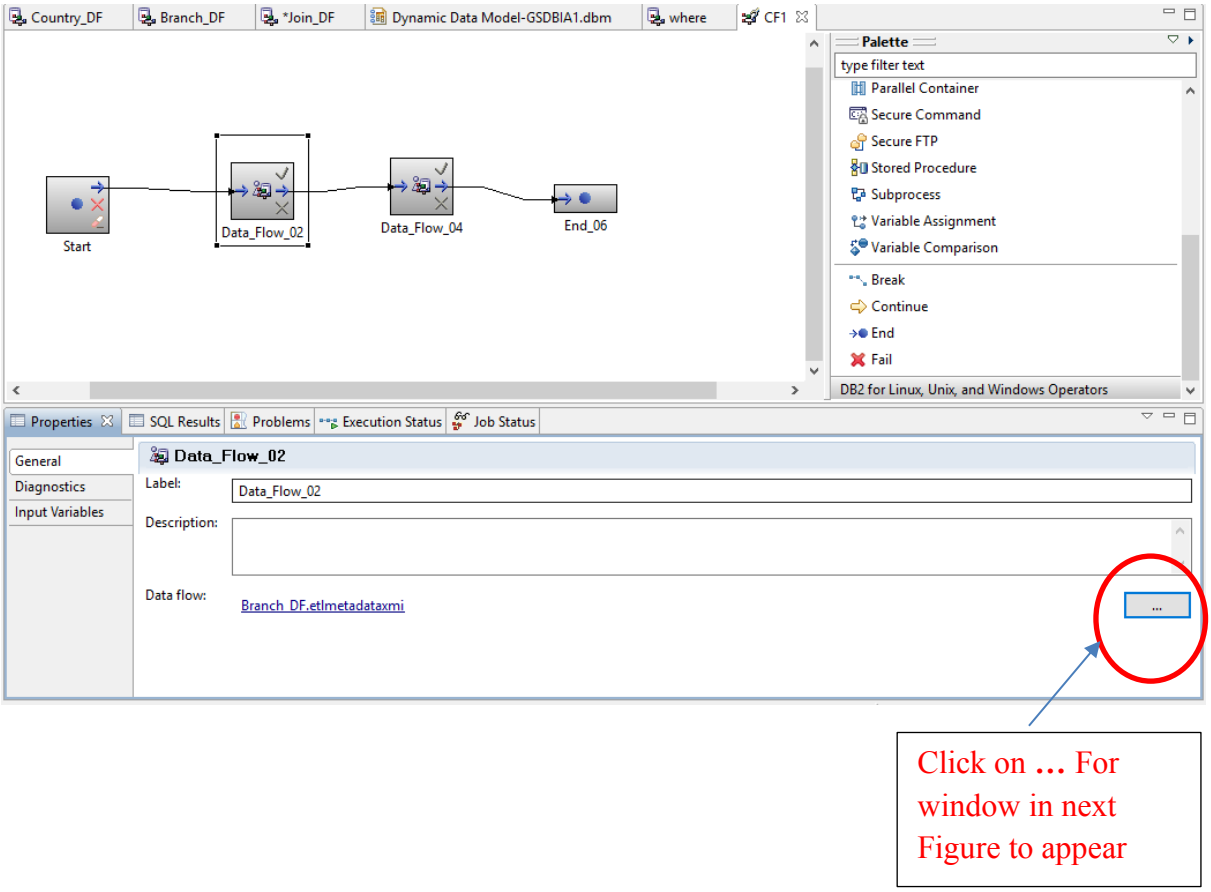

Figure 3.4 Properties tab of Data_Flow_02 to set the data flow of BRANCH table

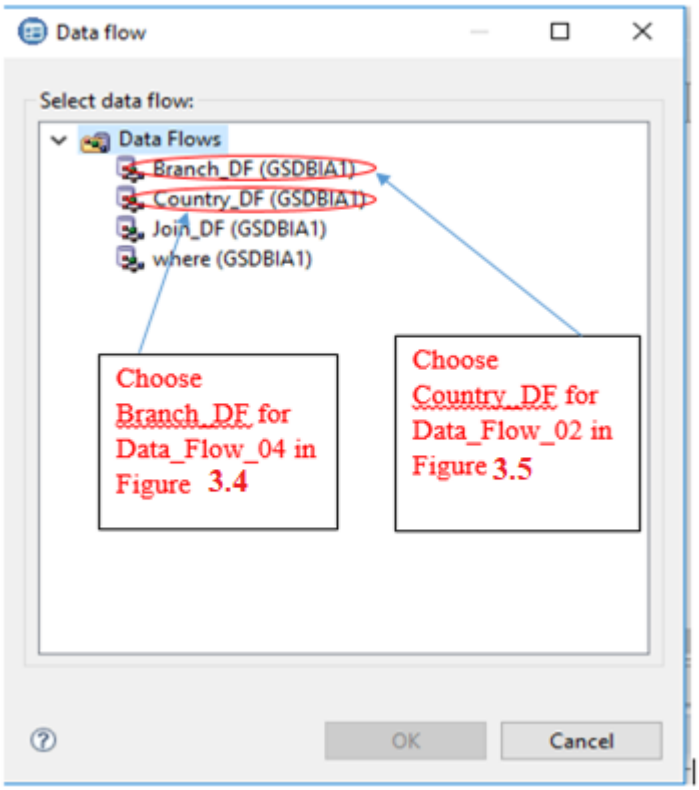

Figure 3.5 Choosing data flows for the Data_Flow operators

NOTE: Branch_DF is the data flow for populating BRANCH table and Country_DF is the data flow for COUNTRY table.

Once the control flow is completed, it can be executed using the **green execute button** present on the top as shown in Figure 3.6.

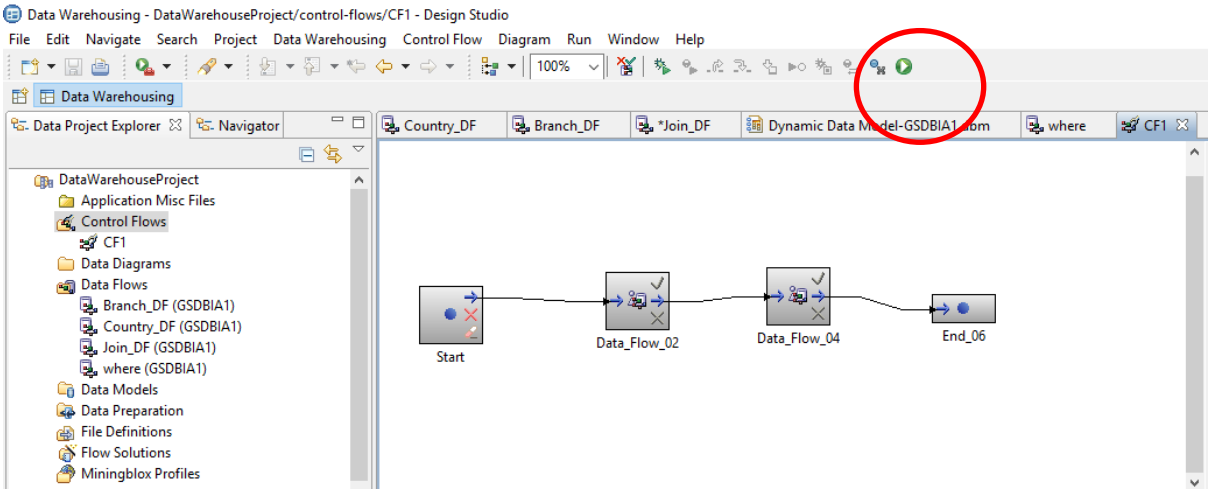

Figure 3.6: Execute button

Control Flow using Data Flows Parallel:

Two or more dataflow can be organized in parallel in a control flow when there is no dependency between the data flows. Consider two dataflows; one for **Joining tables** and other data flow with **where operator,** to be organized in the control flow (These two dataflows have no dependency of data on one another). Figure 3.7 shows the control flow with parallel execution of data flows.

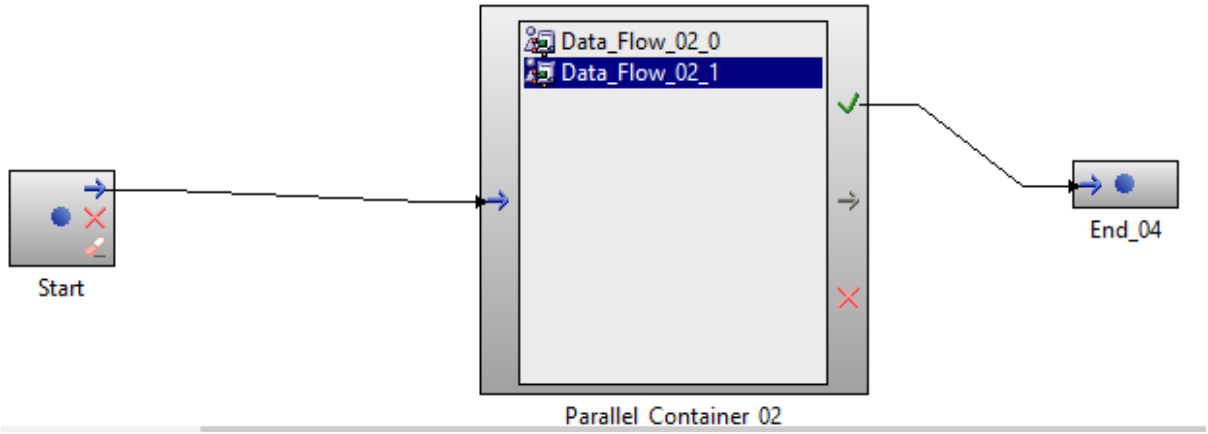

Figure 3.7: Control flow with parallel execution of data flows

Drag and drop the **parallel container** operator present in Palette. Also, drag and drop two Data flow operators on the Parallel Container as shown in Figure 3.7. Connections must be done as shown in Figure 3.7. Data_Flow_02_0 in Figure 3.7 must be set to data flow for **joining two tables** and Data_Flow_02_1 must be set to data flow for **where operator** in property tab of Data_Flow_02_0 and Data_Flow_02_1 respectively. Once the construction of control flow is complete, it may be executed. The control flow also provides many other operators such as file write to log messages on success or failure of a process.

Lab Exercise

1. Do the following for Lab 2 Exercises:

    i. Figure out the SQL queries which needs nesting of queries.
    ii. Draw one data flow for each level of nesting.
    iii. Obtain the final result by executing the control flow having data flows from Question (ii).
    iv. Log suitable messages into a file on success or failure of a data flow.

Additional Exercise

1. Do the following for Lab 2 Additional Exercises:

    i. Figure out the SQL queries which needs nesting of queries.
    ii. Draw one data flow for each level of nesting.
    iii. Obtain the final result by executing the control flow having data flows from Question (ii).
    iv. Log suitable messages into a file on success or failure of a data flow.

LAB NO: 4 **Date:**

RAPID MINER OPERATORS

Objectives:

1. To understand the different operators for data access in Rapid miner.
2. To understand the different operators for data preprocessing in Rapid miner.

Introduction:

Rapid Miner Studio is open source and in process view of the Studio there are five panels. They are Repository, Operators, Process, Parameters and Help. Figure 4.1 shows the overall panels in process view of the tool. Rapid Miner Studio provides operators for Data Access, Blending, Cleansing, Modeling, Scoring, Validation and Utility.

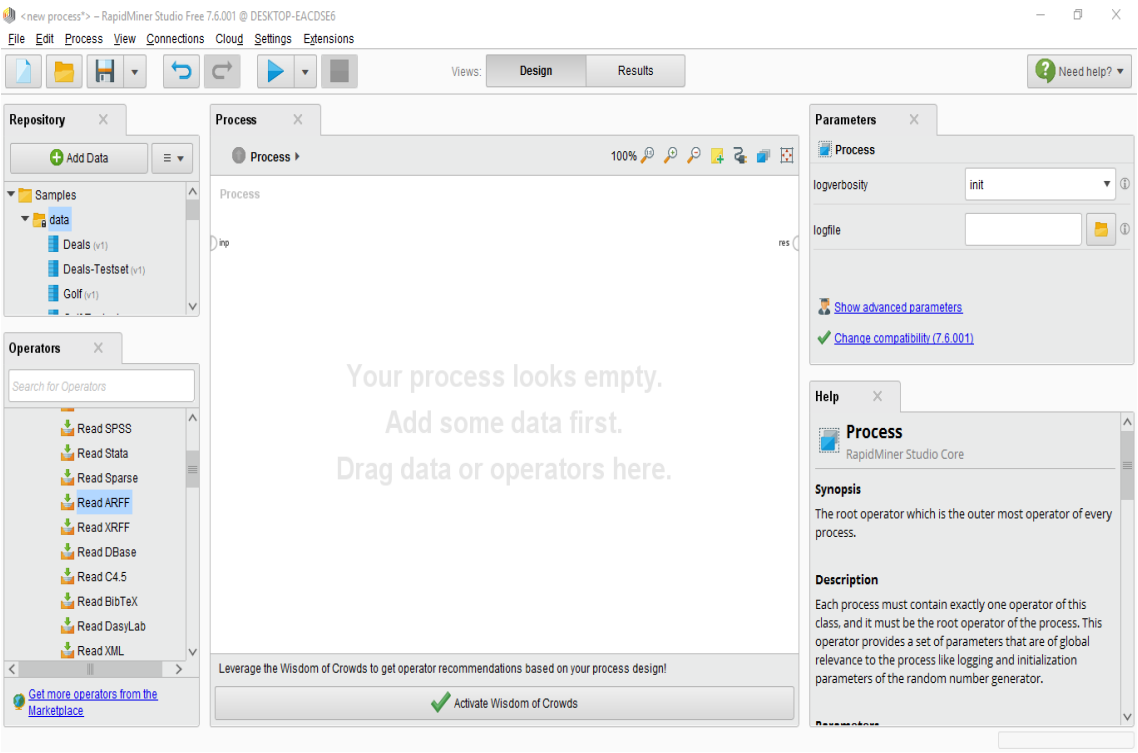

Figure 4.1. Process View

This section provides details about various operators in Rapid Miner Studio such as Data Access operators and Data preprocessing operators.

1. Data Access Operators

This section has operators to

i. read and write data from and into File respectively.
ii. read, write and update Database
iii. obtain data from Web Applications

i. Reading from Files

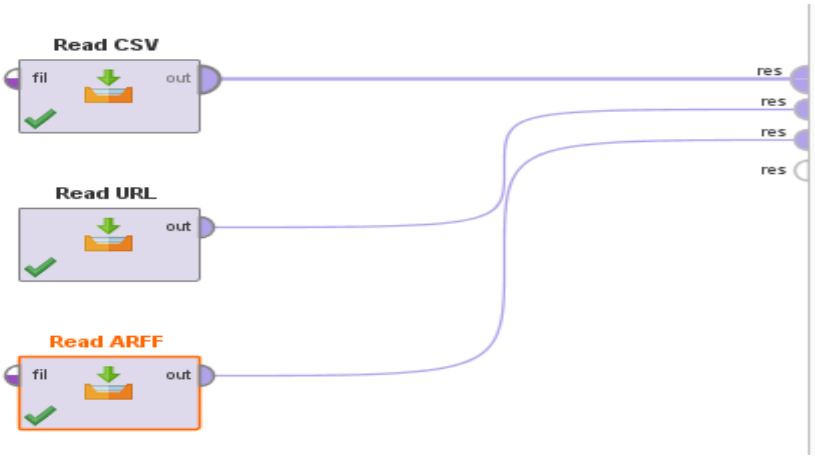

Figure 4.2. Example operators for reading from file

As shown in Figure 4.2 drag and drop Read CSV, Read URL and Read ARFF operators to the process panel and connect their out ports to res port. Set the configuration for each operator in parameter panel by clicking on respective operator icon. In order to see the result click on Execute button in top panel.

ii. Read-Write operation

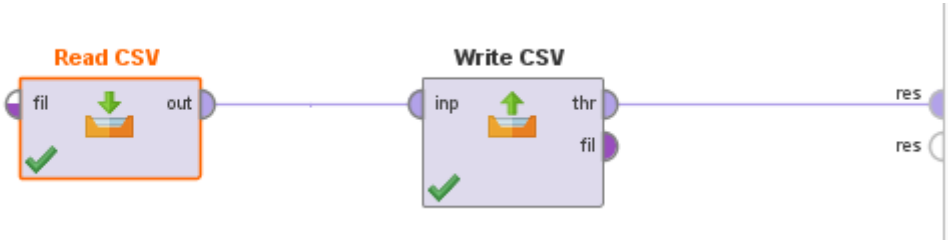

Figure 4.3. Read Write Operation

As shown in Figure 4.3, in order to write into a CSV file drag and drop a Read CSV and Write CSV operators to the Process Panel and then connect out port of Read CSV operator icon to inp port of Write CSV operator. The thr port of Write CSV should be connected to res port. The path of the file into which content has to be written should be given in parameter panel. Using repeated Write operator, it is possible to write into multiple files. Rapid miner also provides sample data set whose operators can be dragged and dropped on to process panel. Table 4.1 provides various port abbreviations and their meaning and description in each operator icon.

Table 4.1. Port Abbreviations

Port Abbreviation	Meaning	Description
Ass	*Association*	Association rules that have been discovered in a frequent item set
Att	*Attribute*	Attribute weights (in and out)

Ave	*Average*	Performance measures; estimate of performance using the model built on the complete delivered data set
Clu	*Cluster model*	Cluster model created when clustering an example set
Exa	*Example set*	Example set
Fil	*File*	File object
For	*Formula*	Formula result
Fre	*Frequent*	Frequent item or item sets for association rule learning
Gro	*Grouped*	Grouped models, attributes, items
Hie	*Hierarchical*	Hierarchical clustering model
Inp	*Input*	Input source, can take various objects
Ite	*Item sets*	Frequent item sets (groups of items that often appear together in the data)
Joi	*Join*	Join of the left and right example sets
Lab	*Labeled data*	Model that was given in input is applied on the example set and the updated example set is delivered from this port
Lef	*Left*	Left input port expecting an example set, which is used as the left example set for a join
Mat	*Matrix*	Correlations matrix of all attributes of the input example set
Mer	*Merged*	Merged example set
Mod	*Model*	Default model from this output port
Obj	*Object*	IO object
Ori	*Original*	Input example set is passed without changing to this port
Out	*Output*	Output port

Adding Twitter Data

Drag and drop the twitter icon on process panel. Connect the out port to res port as depicted in figure 4.4.

Figure 4.4. Adding Twitter data

In the parameter window, do the following to set the connection:

1. Create an access token for Rapid Miner to request twitter feed.
2. Click on twitter symbol against connection field.
3. Click Add Connection - Give a connection name and select connection type as twitter connection from the drop down. Click on create button.
4. Against access token field, paste the access token received (can be created in https://apps.twitter.com/app/) and click on button, Request access token as shown in Figure 4.5.

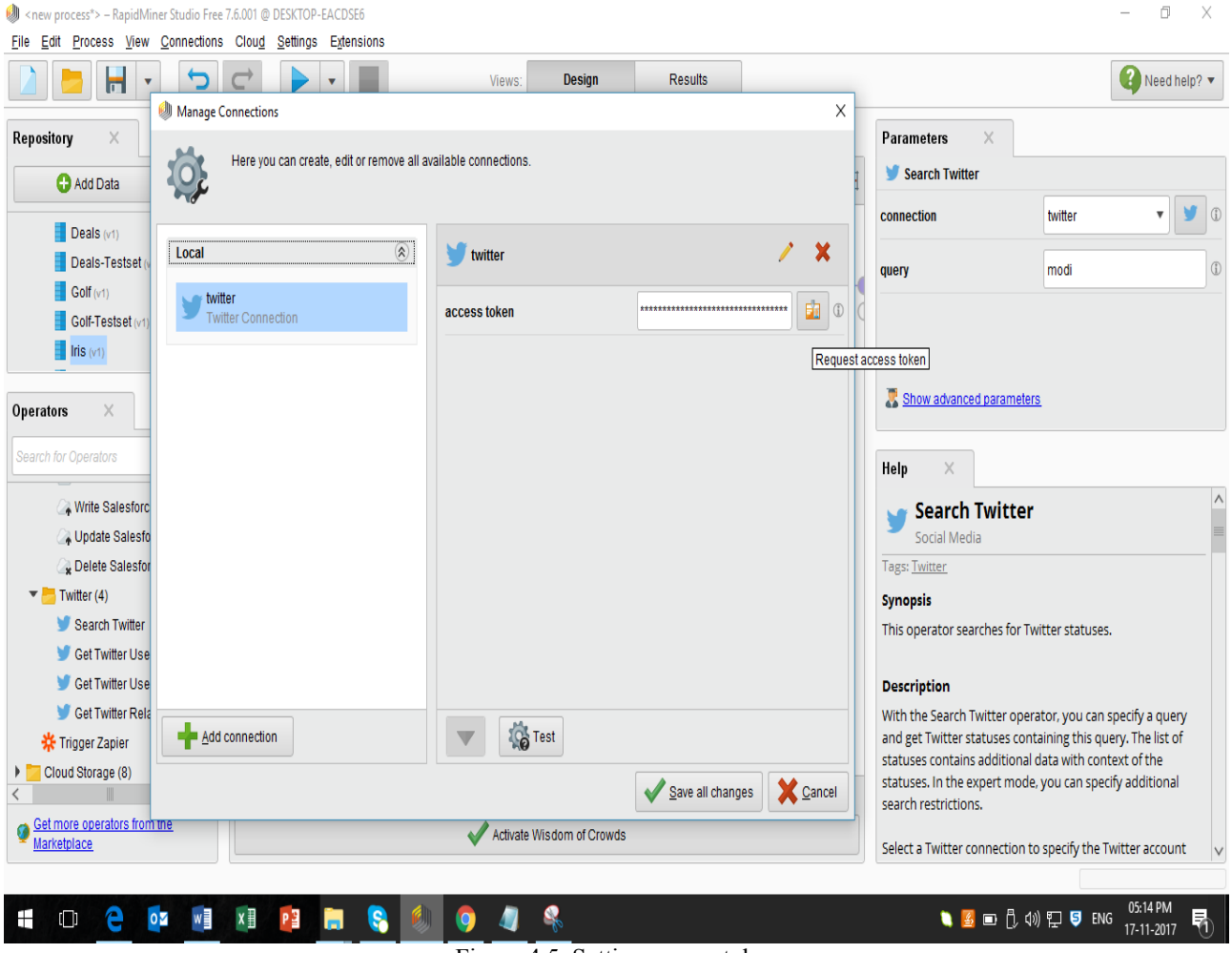

Figure 4.5. Setting access token

5. On click of the Request access token button the window shown in Figure 4.6 appears.

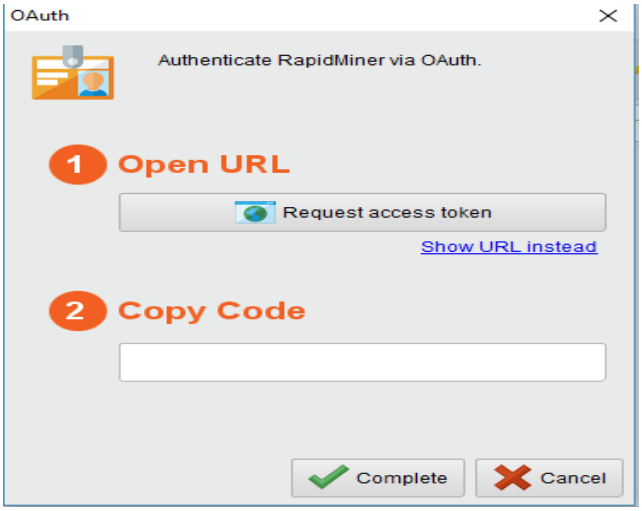

Figure 4.6. Authentication

6. Click on Request access token the web page shown in Figure 4.7 appears.

Figure 4.7. Authorization

7. Click on Authorize app.
8. Enter the PIN received in “COPY CODE” field. Click on complete.
9. Click on Test and followed by Save all changes.
10. Enter a query in query field. Example “Modi”
11. Execute.

The result of these steps is shown in Figure 4.8

	Id	Created-At	From-User	From-User-Id	To-User	To-User-Id	Language	Source	Text
	9309942641...	Nov 16, 2017 ...	Office of RG	3171712086	?	-1	en	<a href="http:/...	Modi ji - nice t...
	9310547262...	Nov 16, 2017 ...	Amit Shah	1447949844	?	-1	en	<a href="http:/...	The findings ...
	9310835361...	Nov 16, 2017 ...	BJP	207809313	?	-1	en	<a href="http:/...	Yet another e...
	9314834123...	Nov 17, 2017 ...	Shivanna Ma...	9271721907...	INCIndia	1153045459	in	<a href="http:/...	@INCIndia M...
	9314834102...	Nov 17, 2017 ...	Ankit Srivasta...	70166896	?	-1	en	<a href="http:/...	RT @sharma...
	9314834057...	Nov 17, 2017 ...	smitha060444	9049645668...	?	-1	en	<a href="http:/...	RT @Paperb...
	9314834043...	Nov 17, 2017 ...	Raviraj Virkar	2402557338	?	-1	en	<a href="http:/...	RT @muglika...
	9314833977...	Nov 17, 2017 ...	GLOBALCITI...	621030151	Knkinjal_	9106516548...	in	<a href="http:/...	@Knkinjal_ ...
	9314833971...	Nov 17, 2017 ...	Rajesh Kumar	8963371510...	awasthis	71815077	hi	<a href="http:/...	@awasthis ...
	9314833664...	Nov 17, 2017 ...	Shrimant Mane	120393444	?	-1	en	<a href="http:/...	Moody's Upgr...
	9314833656...	Nov 17, 2017 ...	Sheikh Muha...	8590654530...	?	-1	en	<a href="http:/...	RT @Jagrati...
	9314833630...	Nov 17, 2017 ...	Capt.Nilesh	4188258740	?	-1	en	<a href="http:/...	RT @ggiiittiikk...
	9314833618...	Nov 17, 2017 ...	True Fight	3092961320	?	-1	en	<a href="http:/...	RT @rahulro...
	9314833590...	Nov 17, 2017 ...	Retweet Wal...	1366646221	?	-1	en	<a href="http:/...	RT @rahulro...
	9314833577...	Nov 17, 2017 ...	Al Humbert	305060948	?	-1	en	<a href="http:/...	RT @AmarA...

Figure 4.8. Result of Twitter data access

Database Connection in Rapid Miner

1. Drag and drop the Read Database operator on the process panel as shown in Figure 4.9.

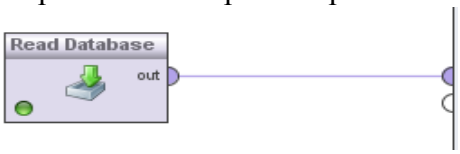

Figure 4.9. Data base access

2. Do the following configuration in Property window:
   a) Set the define connection to “predefined”.
   b) Click on the button “create, edit, delete database connections, beside connection as shown in the screenshot below in Figure 4.10.

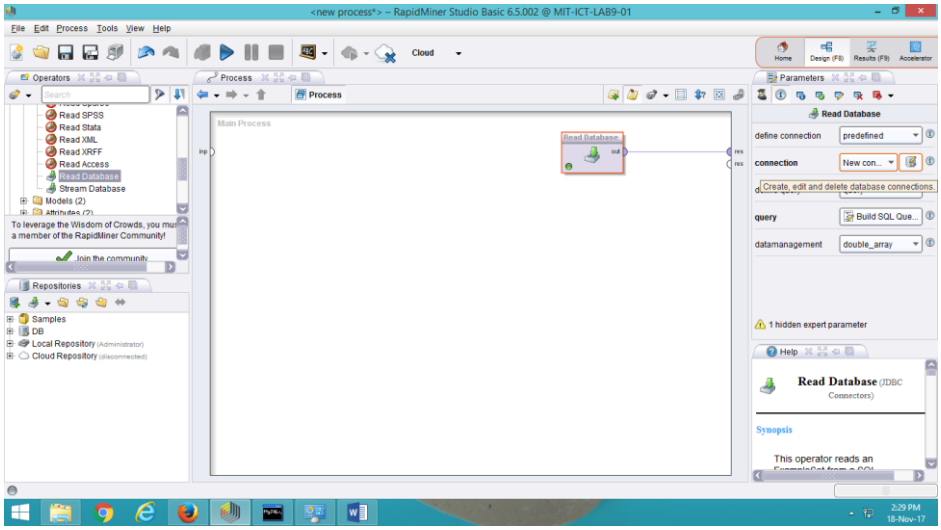

Figure 4.10. Screen shot for setting parameters for DB connection

   c) A window (Figure 4.11) will appear in which the following information must be entered:
      - Connection name
      - Database system: MySql
      - Host: Localhost, Port:3306
      - Database scheme (The database name created in the backend)
      - User:root
      - Password:student

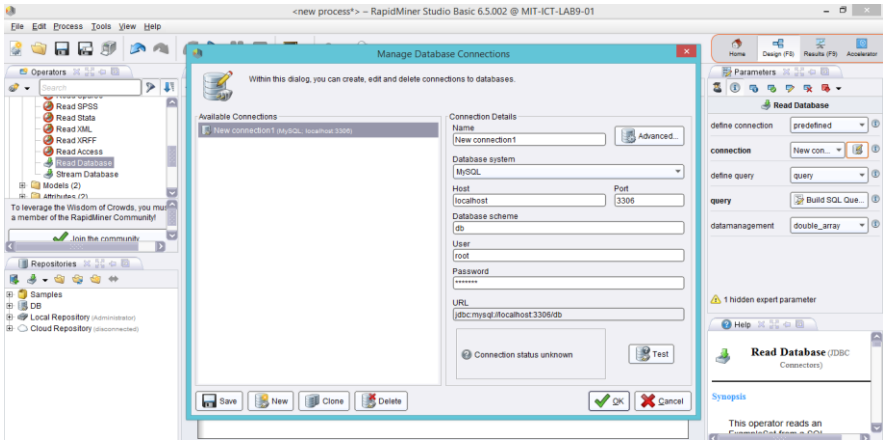

Figure 4.11. Setting parameters for DB access

d) Click on "Test" button.

- Click on the query button in the property window. The window as shown in Figure 4.12 will appear where the query is to be entered.

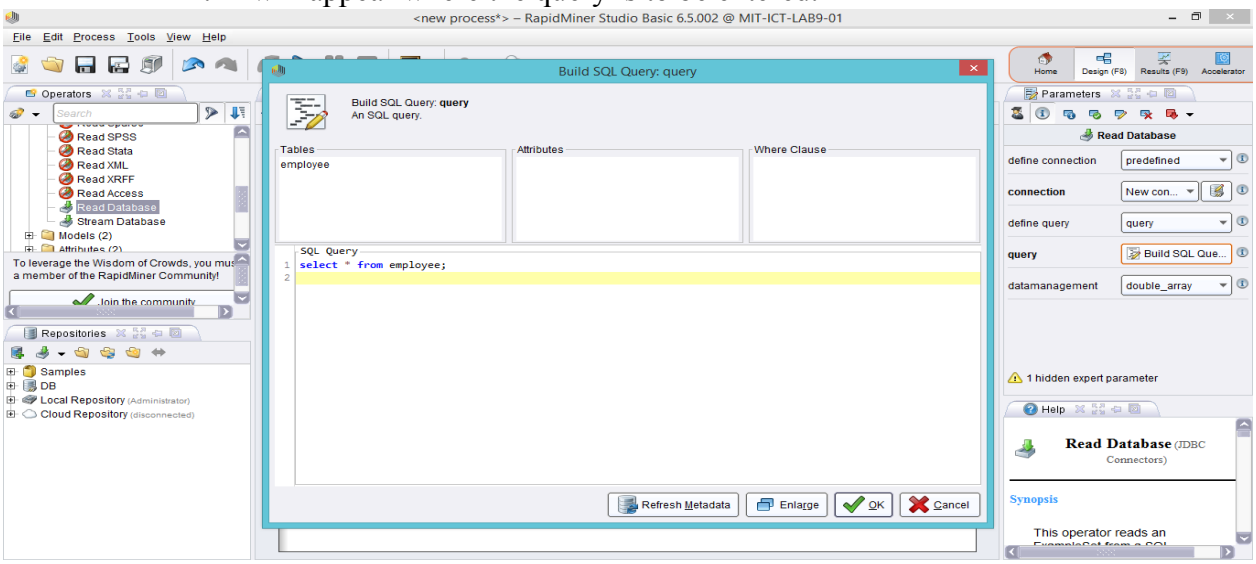

Figure 4.12. Entering SQL query for DB Access

3.Execute the process to obtain the data required by the query.

2. Data Preprocessing Operators

I. Blending Operators

A. Attributes

i. Names & Roles

- **Rename:** Renames the attribute names.

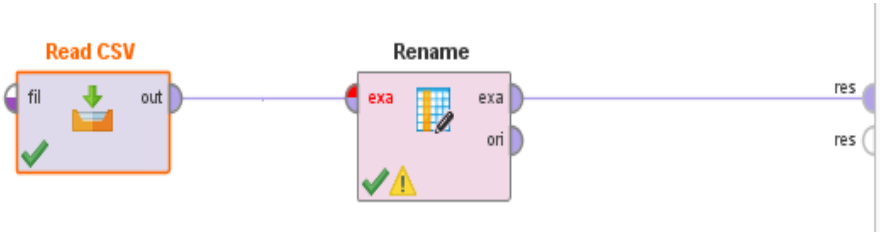

Figure 4.13. Renaming file

Figure 4.13 depicts the process diagram for renaming a file. In the parameter window of Rename operator specify the old attribute name in dataset file and the new name which user requires.

- **Rename by Generic Names**: This operator renames the selected attributes of the given ExampleSet to a set of generic names like att1, att2, att3 etc. Figure 4.14 shows the process diagram for usage of the Rename by Generic Names operator.

The Write Constructions operator writes all attributes of the ExampleSet given at the input port into the specified file. The path of the file is specified through the *attribute constructions*

file parameter. Each line in the file holds the construction description of one attribute. The Read Constructions operator can be used for reading this file.

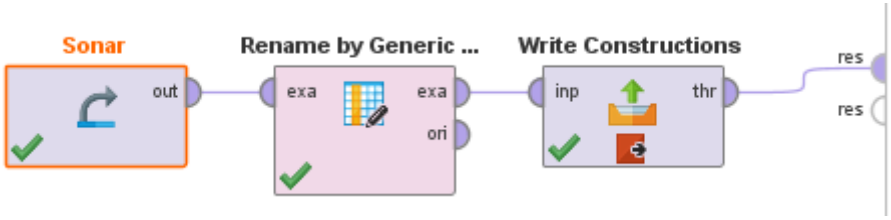

Figure 4.14. Rename by generic names

ii. **Types:** This operator changes the type of the selected numeric attributes to a binominal type. It also maps all values of these attributes to corresponding binominal values. Figure 4.15 shows the process diagram for numeric to binomial operator.

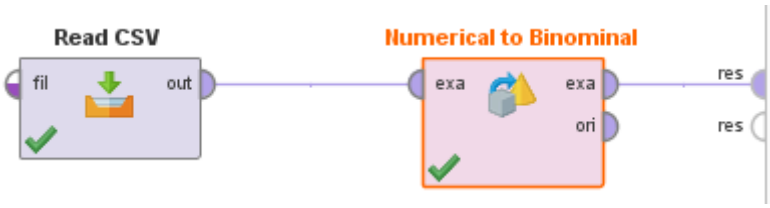

Figure 4.15. Types

iii. Selection

- **Select Attributes**: This Operator selects a subset of Attributes of an ExampleSet and removes the other Attributes. The Figure 4.16 demonstrates that a subset of attribute is selected in the parameter window and attributes which need to appear is selected in the "Select Attribute button" against "attributes" field.

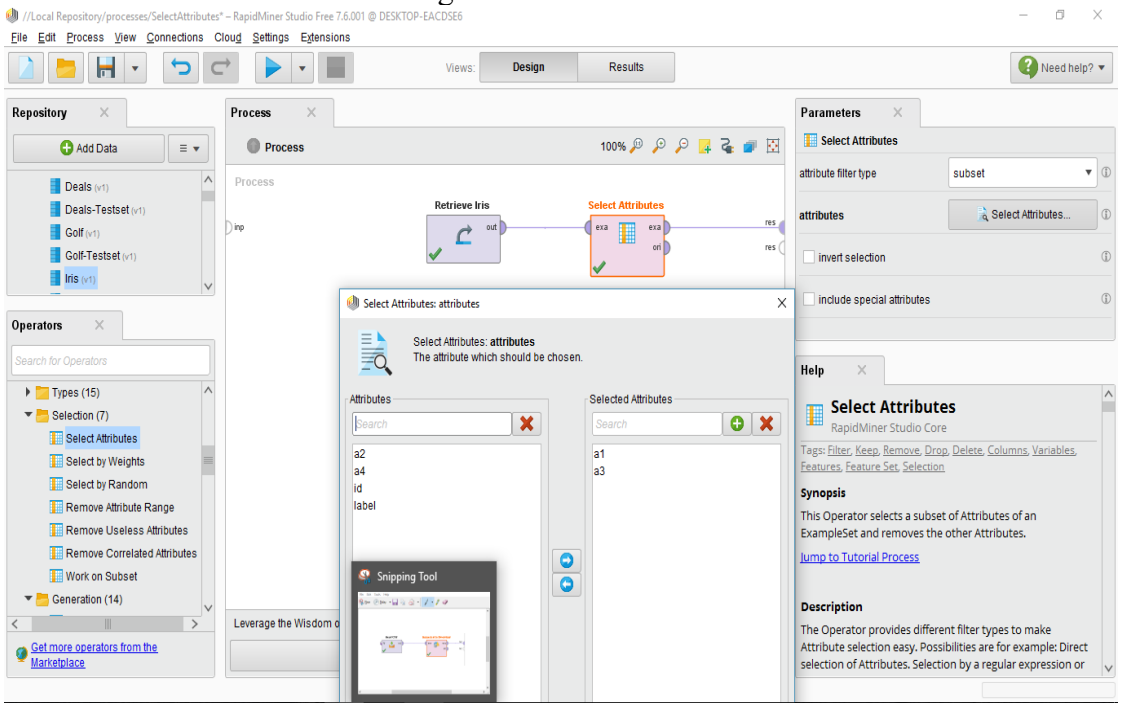

Figure 4.16. Setting parameters for selection

- **Remove correlated attributes:** This operator removes correlated attributes from an ExampleSet. The correlation threshold is specified by the user in the correlation field of parameter window. Figure 4.17 shows the process diagram for the operator.

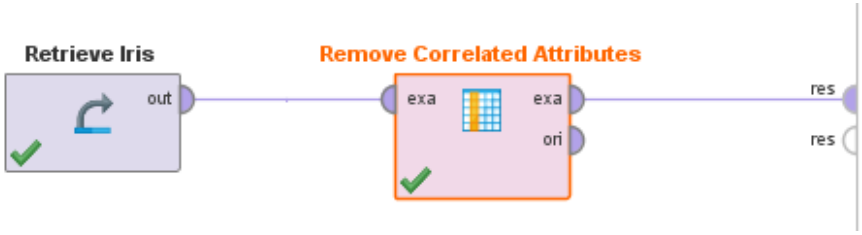

Figure 4.17. Removing correlated attributes

iv.Genertation

- **tf-idf:** This operator performs a TF-IDF filtering of the given ExampleSet. The usage of the operator is shown in process diagram of Figure 4.18.

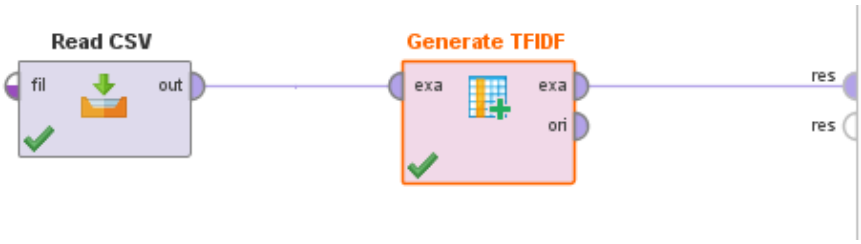

Figure 4.18. Generation of TFIDF

B. Examples

i) Filter Example operator

Filters examples based on a condition applied on an attribute. Figure 4.19 shows the parameter setting window where the user needs to enter attribute and values based on which dataset must be filtered.

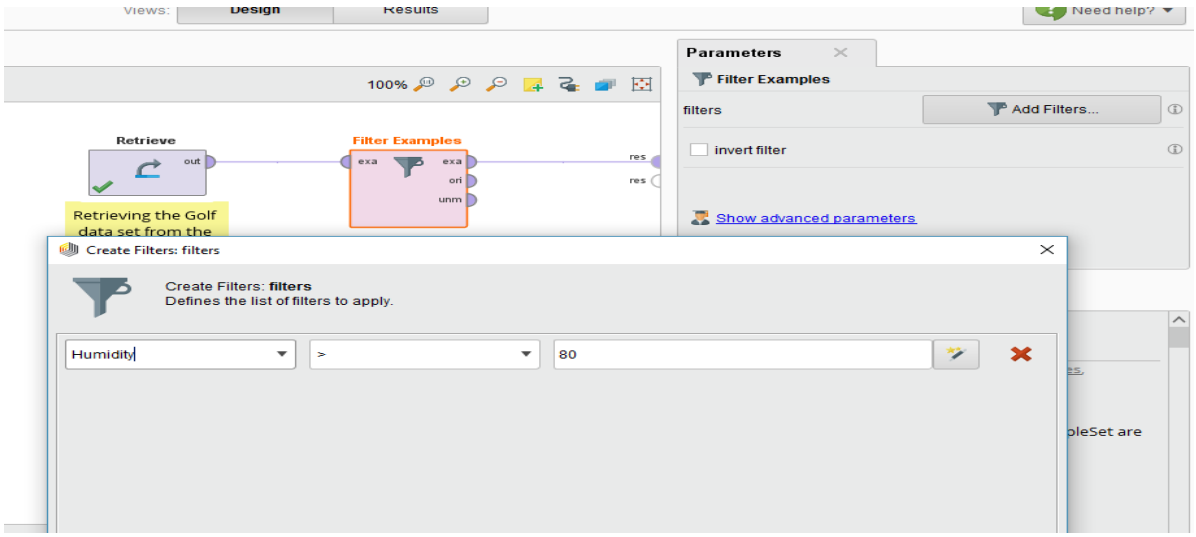

Figure 4.19. Filter operator parameter setting

ii) Sampling operator

- **split data:**

Splitting data set according to the ratio provided in the parameter panel. Figure 4.20 shows the parameter setting window where the ratios add to 1 and linear sampling is selected.

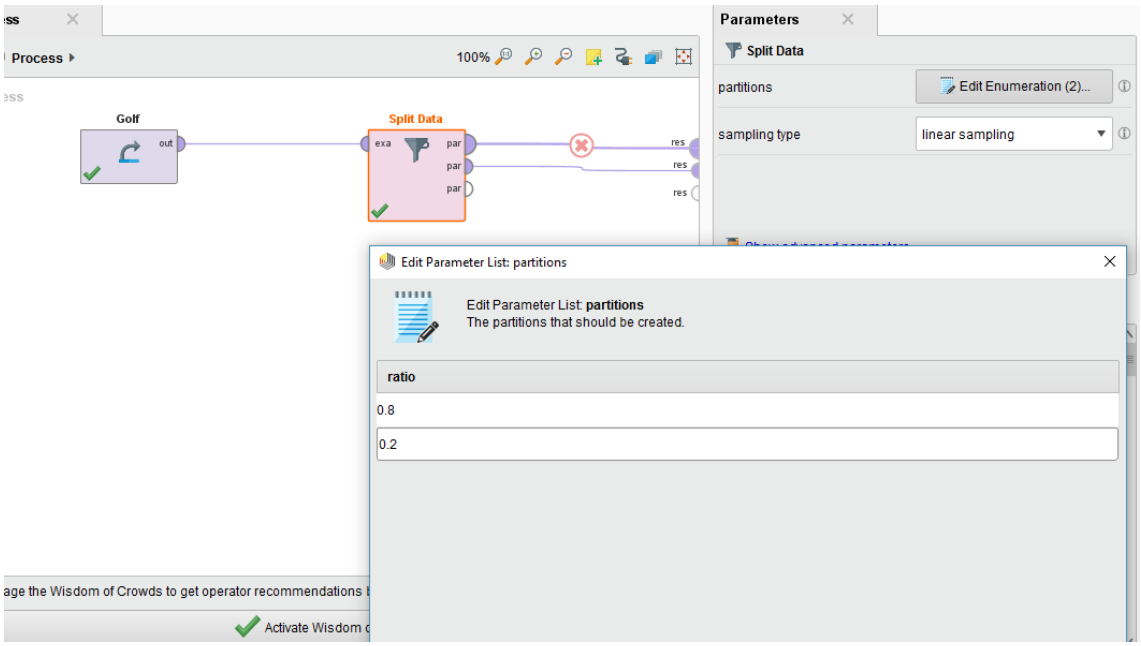

Figure 4.20. Sampling input data set.

ii)Sorting: Figure 4.21 depicts the process and parameter setting for the sort operator.

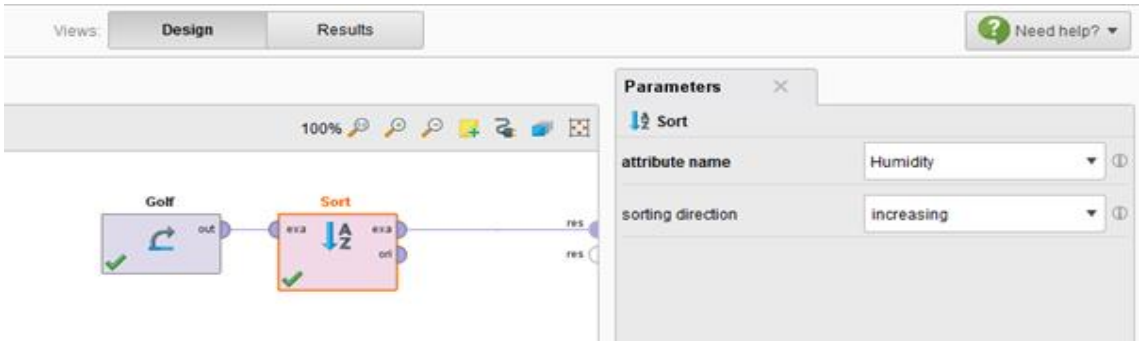

Figure 4.21. Sorting

C. Tables

i)Rotation - Transpose**:** Figure 4.22 depicts the process diagram for transpose operator.

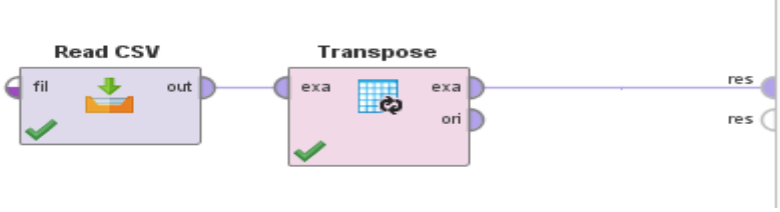

Figure 4.22. Finding Transpose

ii)Joins - Set minus operator: Figure 4.23 depicts the process diagram for Set Minus operator

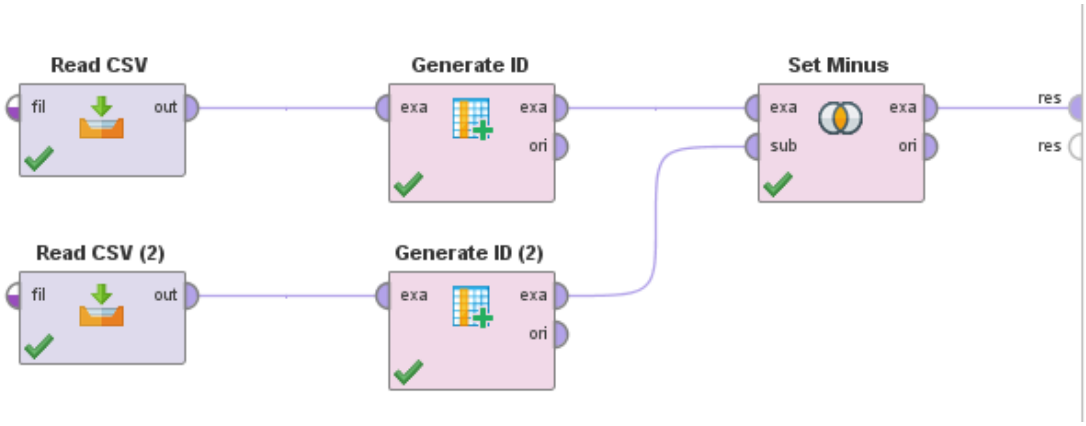

Figure 4.23. Subtraction of Data set.

1. Data Cleansing Operators:

i)Normalize: This operator normalizes the attribute values of the selected attributes.

ii)Replace Missing Values: This operator replaces missing values in examples of selected attributes by a specified replacement. Figure 4.24 shows the parameter setting where all missing values would be filled with zeros.

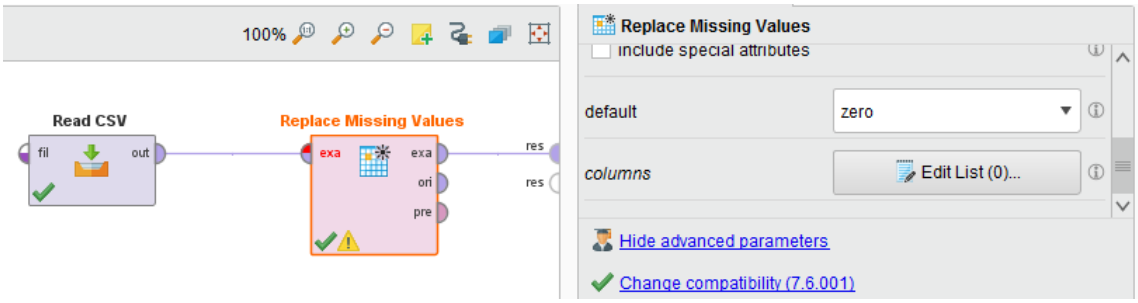

Figure 4.24. Replacing missing values

iii)Remove Duplicates: This operator removes duplicate examples from an ExampleSet by comparing all examples with each other on the basis of the specified attributes. Two examples are considered duplicate if the selected attributes have the same values in them. Figure 4.25 shows the process diagram for remove duplicates.

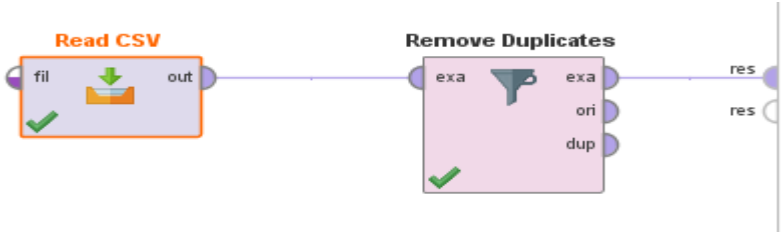

Figure 4.25. Removing duplicates

iv)Detect Outlier : This operator identifies *n* outliers in the given ExampleSet based on the distance to their *k* nearest neighbors. Figure 4.26 shows the variables *n* and *k* that can be specified through parameters as 200 and 2 respectively. The output of the outlier detection is shown in Figure 4.27..

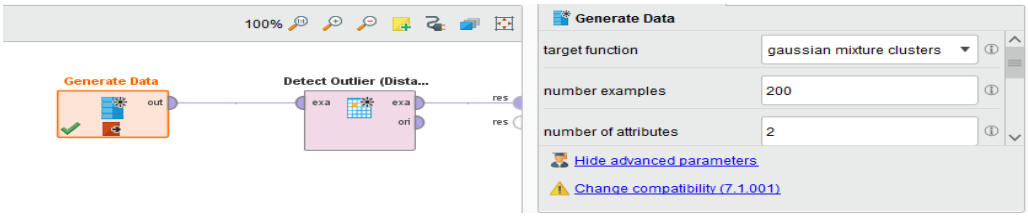

Figure 4.26. Detection of outliers

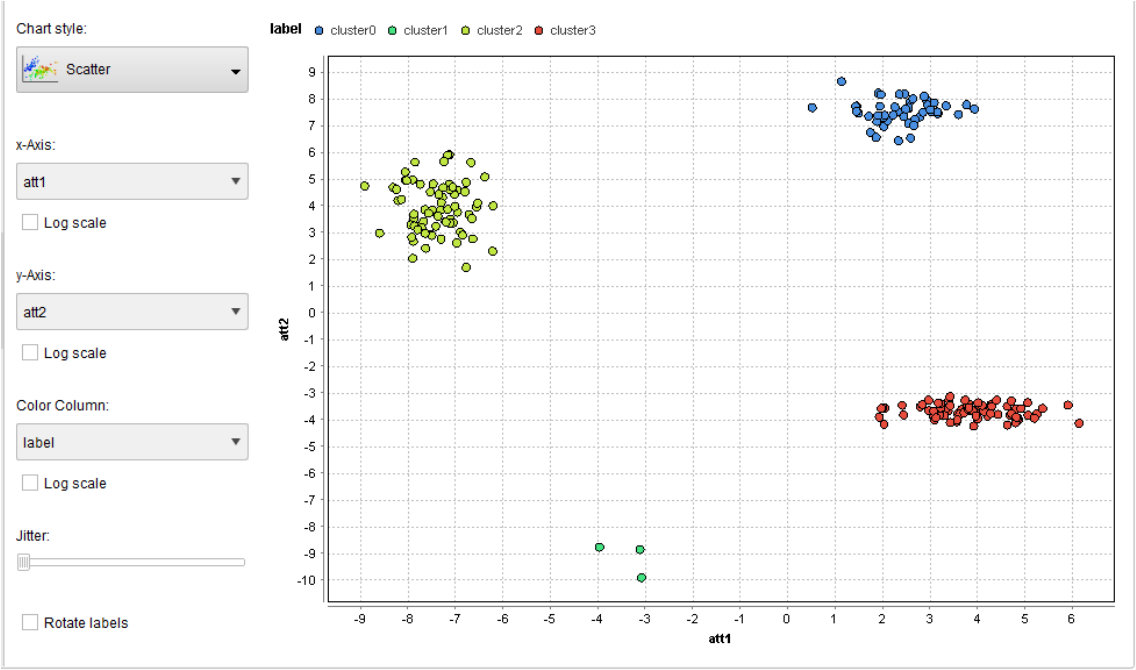

Figure 4.27. Result of outlier detection

v)Dimesionality Reduction(PCA) : This operator performs a Principal Component Analysis (PCA) using the covariance matrix. The user can specify the amount of variance to cover in the original data while retaining the best number of principal components. The user can also specify manually the number of principal components. Figure 4.28 shows the process diagram for PCA operator.

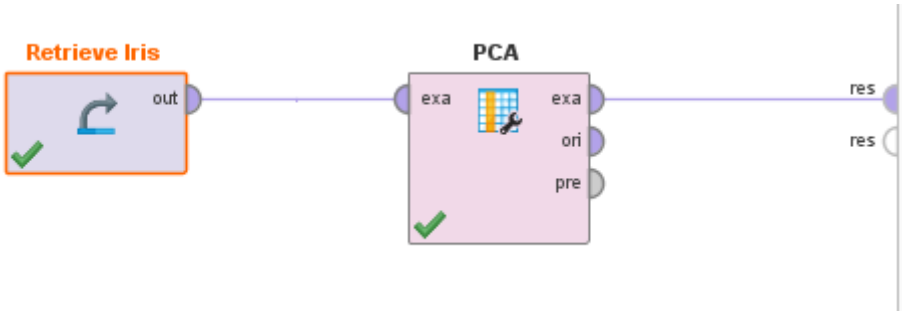

Figure 4.28. Dimensionality Reduction

Lab Exercises:

1. Explore all the other operators listed under operator panel for preprocessing of data. Draw the process diagram and list configuration parameters with sample input and output for the following.
   i. Read excel
   ii. Write Excel
   iii. Numerical to Binominal
   iv. Nominal to Binomial
   v. Select Attributes
   vi. Filter Example
   vii. Sample
   viii. Split Data
   ix. Sort
   x. Transpose
   xi. Intersect
   xii. Normalize
   xiii. Discretize by frequency
   xiv. Replace Missing Values
   xv. Remove Duplicates

2. Download the data set from UCI repository (https://archive.ics.uci.edu/ml/datasets/Car+Evaluation). The file is in csv format.
   i. Import the data to Rapid Miner with proper attribute names and attribute type.
   ii. Divide the dataset into 4 partitions based on the class label.
   iii. Considering the dataset with class label "unacc", What is the most common(maximum occurrence) value of attribute "buying".
   iv. Is there correlation between the attributes "doors" and "persons".
   v. Remove the attribute "maint".
   vi. Is it possible to apply transformation on attribute 'doors'? What can be done to apply transformation operator on this attribute?

3. Consider the dataset https://www2.stetson.edu/~jrasp/data/AMZN-KO.xls Which is daily returns for the stocks of two companies, Check whether the attributes are highly correlated.

Additional Exercise

1. Consider a data set with attributes RollNo, Score1, Score2. Perform the following operations.
   i. Select all attributes except RollNo and display the data.
   ii. How many missing values are there in each attribute?
   iii. Find statistics such as minimum and maximum in Score1 and Score2.
   iv. What is Z-transformation? Apply Z-transformation Score1 and Score2 and find the minimum and maximum value after transformation. Also find the mean and standard deviation.
   v. Check whether the distribution remains same for Score 1 and Score 2 after applying z-transformation
   vi. What is Range transformation? Apply range transformation and find minimum, maximum and average values.

LAB NO: 5 **Date:**

DATA VISUALIZATION AND MODELLING USING CLASSIFICATION

Objectives:

1. To visualize the dataset.
2. To explore association mining and clustering operators in rapid miner
3. To model the data set.

Data Visualization:

RapidMiner provides several plotters in order to visualize the properties of a data set. Each time an ExampleSet is part of the (intermediate) result, the user can select between data view, statistics view and charts view. The charts view provides several 2D and 3D charts, histogram charts, and other charts for high-dimensional data sets. Figure 5.1 shows pie chart for an example data set.

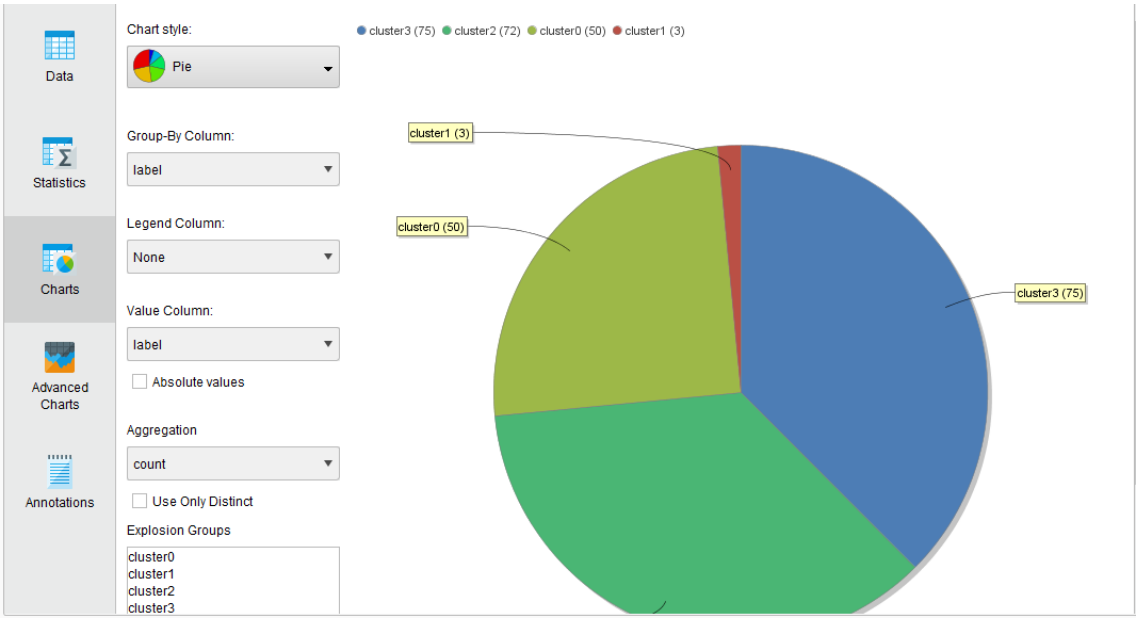

Figure 5.1. Pie chart for an example data set.

I. Modelling using Classification

1. Decision Tree Model creation: : Figure 5.2 shows the process diagram for modelling a decision tree and the Figure 5.3. shows the output obtained.

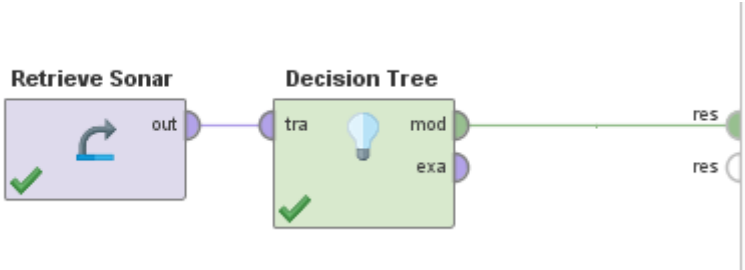

Figure 5.2. Decision tree model creation

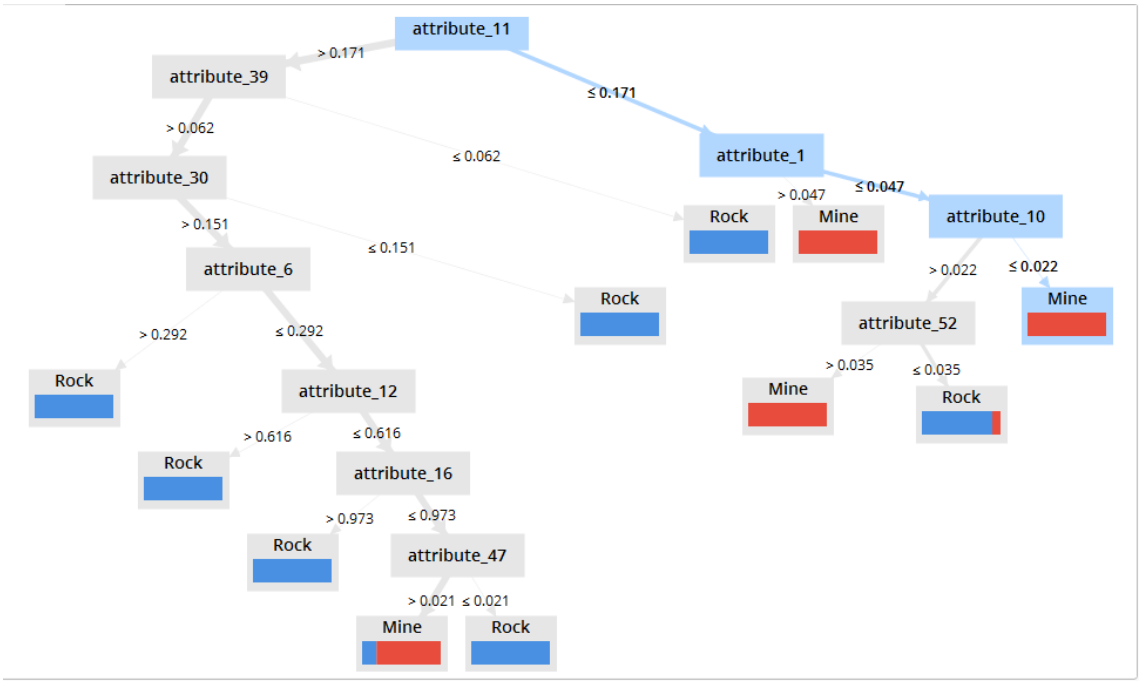

Figure 5.3. Decision Tree

1. **Criterion:** Selects the criterion on which Attributes will be selected for splitting.
2. **Pruning: P**runing the decision tree is optional which if selected confidence for pessimistic error calculation of pruning must be provided.
3. **minimal Maximal depth: gain** (optional) The gain of a node is calculated before splitting it. The node is split if its gain is greater than the minimal gain.
4. **minimal leaf size** (optional) The size of a leaf is the number of Examples in its subset. The tree is generated in such a way that every leaf has at least the *minimal leaf size* number of Examples.
5. **minimal size for split** (optional) The size of a node is the number of Examples in its subset. Only those nodes are split whose size is greater than or equal to the minimal size for split parameter.

Performance of the created model by applying on training data

Figure 5.4 shows the process diagram to create a model using Apply model operator. The performance of the model can be measured using Performance operator. Figure 5.5 shows sample output of Accuracy.

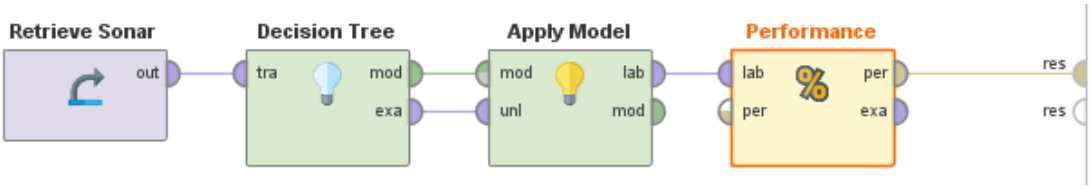

Figure 5.4. Applying model to training data

Performance on Test data

The Sonar dataset is split into two (80% for training and 20% for testing) subsets. Since the decision tree model has to be applied on training and test dataset, apply model operator is used on which Performance operator is applied as shown in the process diagram of Figure 5.6. Figure 5.7 shows the results, there would be two tabs, one for Performance on test data and other for Performance for Training Data.

accuracy: 86.06%

	true Rock	true Mine	class precision
pred. Rock	75	7	91.46%
pred. Mine	22	104	82.54%
class recall	77.32%	93.69%	

Figure 5.5. Accuracy of Training data

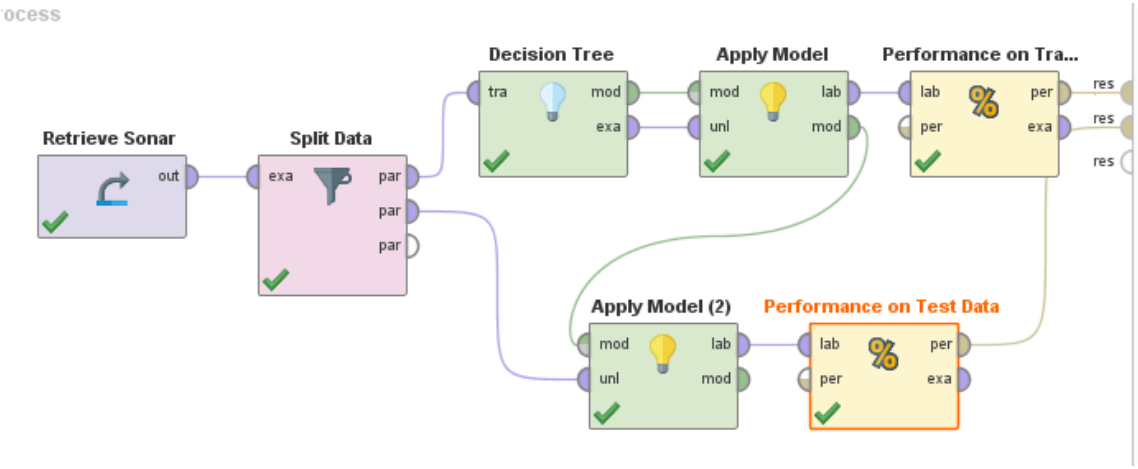

Figure 5.6. Performance on test data

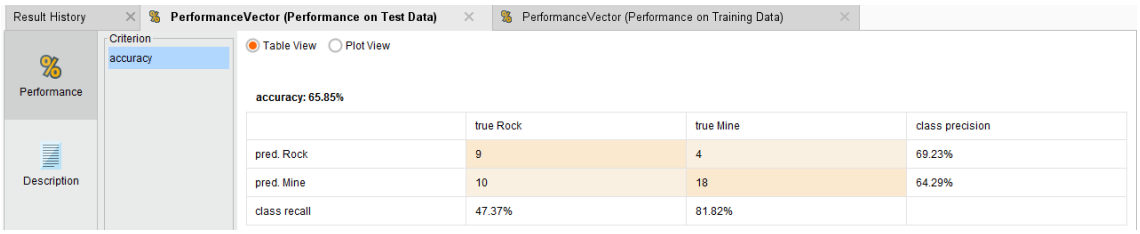

Figure 5.7. Accuracy on test data

Validation of the Result: Performance is obtained by applying cross validation operator as shown in process diagram of Figure 5.8. The results of validation are depicted in Figure 5.9.

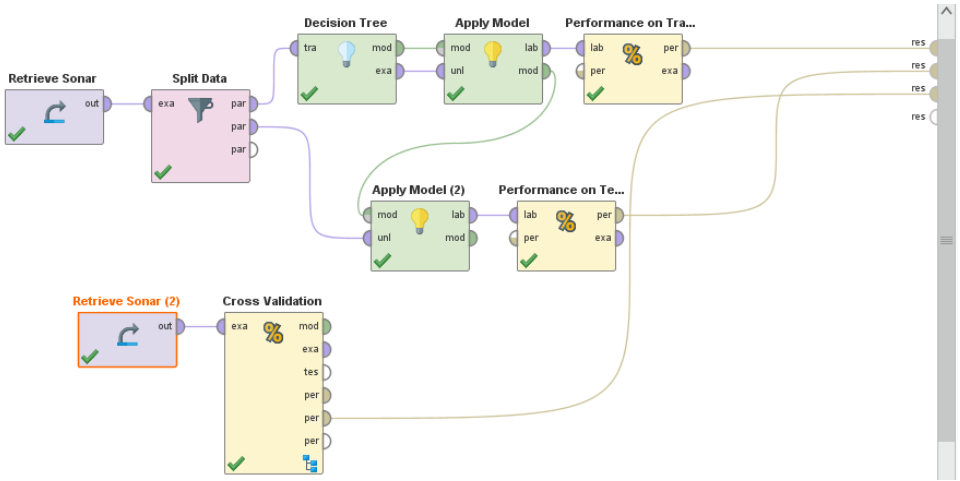

Figure 5.8. Cross Validation operator

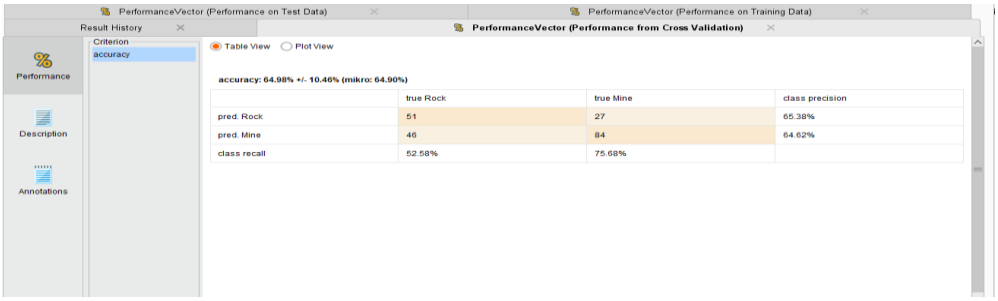

Figure 5.9. Result after cross validation

Prediction

The example for which the class has be predicted should be given in csv file as shown in process diagram of Figure 5.10.

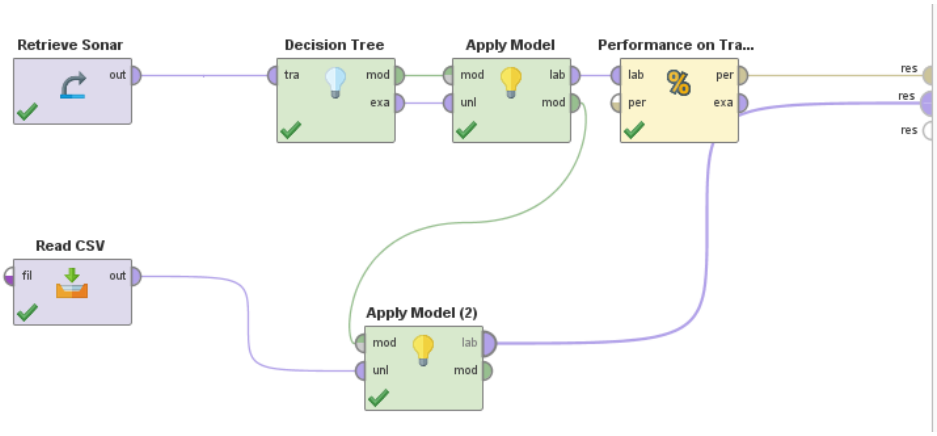

Figure 5.10. Prediction

II. Association Mining

FP-Growth:

This operator efficiently calculates all frequent itemsets from the given ExampleSet using the FP-tree data structure. It is compulsory that all attributes of the input ExampleSet should be binominal. In simple words, frequent itemsets are groups of items that often appear together in the data. It is important to know the basics of market-basket analysis
for understanding frequent itemsets.

This operator has two basic working modes:

- Finding at least the specified number of itemsets with highest support without taking the 'min support' into account. This mode is available when the find min number of itemsets parameter is set to true. Then this operator finds the number of itemsets specified in the min number of itemsets parameter. The min support parameter is ignored in this case.
- Finding all itemsets with a support larger than the specified minimum support. The minimum support is specified through the min support parameter. This mode is available when the find min number of itemsets parameter is set to false.

Create Association Rules

This operator generates a set of association rules from the given set of frequent itemsets. Association rules are created by analyzing data for frequent if/then patterns and using the criteria support and confidence to identify the most important relationships. Support is an indication of how frequently the items appear in the database. Confidence indicates the number of times the if/then statements have been found to be true. The

frequent if/then patterns are mined using the operators like the FP-Growth operator as shown in process diagram of Figure 5.11. The Create Association Rules operator takes these frequent itemsets and generates association rules.

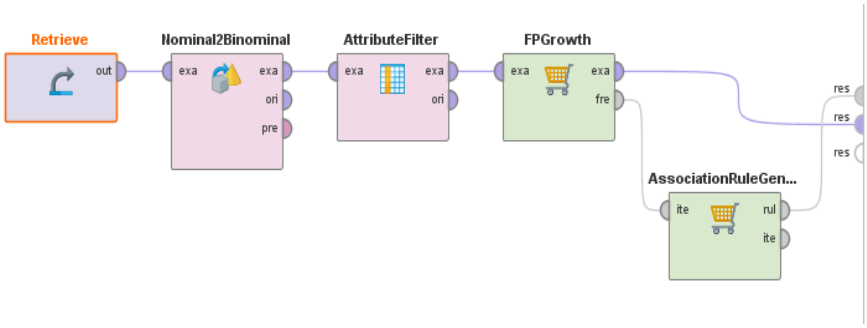

Figure 5.11. Association rule generation

Apply Association rules:

This operator creates a new confidence attribute for each item occurring in at least one conclusion of an association rule. Then it checks for each example and for each rule, if the example fulfills the premise of the rule, which it does, if it covers all items in the premise. An example covers an item, if the attribute representing the item contains the positive value. If the check is positive, a confidence value for each item in the conclusion is derived. Which value is used, depends on the selected confidence aggregation method. There are two types: The binary choice will set a 1, for any item contained inside a fulfilled rule's conclusion. This is independent of how confident the rule was. Any aggregation choice will select the maximum of the previous and the new value of the selected confidence function. Figure 5.12 and 5.13 shows the output of association rules in data and description tab respectively.

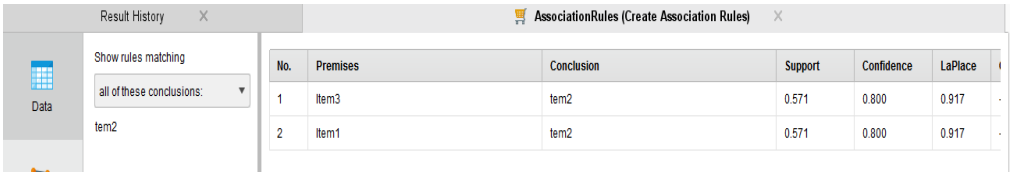

Figure 5.12. Frequent items in data tab

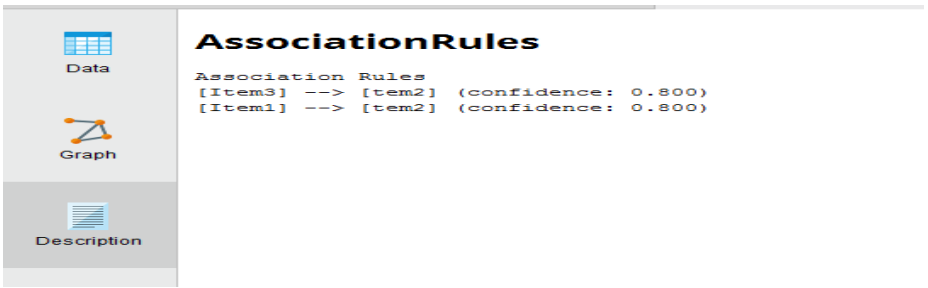

Figure 5.13. Association Rules in Description tab

Clustering

K-Means: This operator performs clustering using the k-means algorithm as shown in process diagram of Figure 5.14. Clustering is concerned with grouping objects together that are similar to each other and dissimilar to the objects belonging to other clusters. Clustering is a technique for extracting information from unlabeled data. Figure 5.15 gives the overview of the cluster by specifying the number of items in each cluster. K-means clustering is an exclusive clustering algorithm i.e. each object is assigned to precisely one of the set of clusters as shown in Figure 5.16.

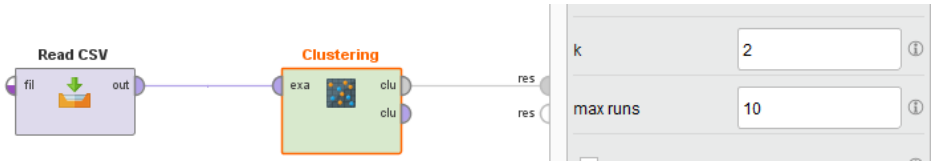

Figure 5.14. K-Means operator applied on example data set

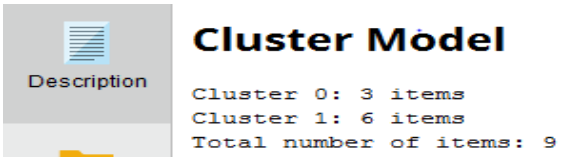

Figure 5.15. Resulting cluster details

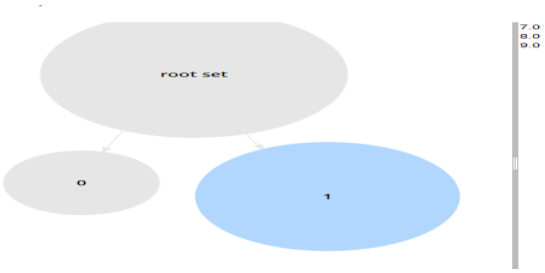

Figure 5.16. Clusters visualized on click of each circle data points

Lab Exercise

1. Apply the following visualizations on any two datasets and write the sample outputs and benefits of Scatter plot, Quartile plot, Deviation, Series, and Distribution plots
2. Retrieve iris data set from the Samples repository in rapid miner. Use different plots to visualize and explore the data set. Which attribute and value ranges(approximate) determine the type of iris flower?
3. Build a process to classify the iris dataset.
   i. Discretize all attributes of the iris data set by frequency into three bins.
   ii. Use split validation operator to generate training and test data set.
4. As inner operator of split validation use the ID3 to learn a decision tree and Performance (classification) operator to evaluate the accuracy of the model.
5. Use Decision Tree and k-NN operators for classifying iris dataset. Which gives best accuracy for the given dataset?

Additional Exercise

1. Apply Naïve Bayes operator for classification. Validate and predict the result as outlined above in manual.
2. Generate association rules for frequent itemsets obtained using FP-Growth operator.
3. Cluster the iris dataset using different algorithms and different parameter settings.
   a. Which algorithm and parameter setting reproduces the original division into three different species?
   b. If the data is normalized before applying the clustering will there be any change in cluster formed?

LAB NO. 6 **Date:**

MINI PROJECT – SYNOPSIS SUBMISSION

Objective

1. **To finalize the title of miniproject**

Students are required to submit the synopsis of the mini project. They are encouraged to select the topic based on the Scopus/WoS indexed paper in data mining area. They can also develop any application based on the data mining algorithms.

LAB NO. 7 **Date:**

APRIORI ALGORITHM

Objectives

2. To implement Apriori algorithm
3. To generate association rules based on frequent item sets

Apriori Algorithm

Agarwal and Srikant proposed the Apriori algorithm in 1994. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as *candidate generation)*, and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found. Apriori uses breadth-first search to count candidate item sets efficiently. This algorithm uses downward closure property, which states that, "Any subset of a frequent itemset must be frequent". It is called as apriori because it uses prior knowledge of frequent item set properties.

It uses level-wise search, where k-itemsets (an itemset containing k number of items is called as a k-itemset) are used to explore (k+1) itemsets to mine frequent itemsets from transactional database. First, the set of frequent 1-temset (L_1) is found. L_1 is used to find L_2, which is used to find L_3 and so on, until no more frequent k-itemsets can be found.

The candidate-gen function takes L_{k-1} and returns a superset (called the candidates) of the set of all frequent k-itemsets. It has two steps

- *join* step: Generate all possible candidate itemsets C_k of length k
- *prune* step: Remove those candidates in C_k that cannot be frequent.

Figure 7.1 shows the pseudocode of the algorithm.

prune(C_k)

for all $c \in C_k$
for all $(k-1)$-subsets d of c ***do***
    if $d \notin L_{k-1}$
    then $C_k = C_k \setminus \{c\}$

gen_candidate_itemsets with the given L_{k-1} as follows:

$C_k = \varnothing$
for all itemsets $l_1 \in L_{k-1}$ ***do***
for all itemsets $l_2 \in L_{k-1}$ ***do***
    if $l_1[1] = l_2[1] \wedge l_1[2] = l_2[2] \wedge \ldots \wedge l_1[k-1] < l_2[k-1]$
    then $c = l_1[1], l_1[2] \ldots l_1[k-1], l_2[k-1]$
    $C_k = C_k \cup \{c\}$

Initialize: $k := 1$, C_1 = all the 1-itemsets;
read the database to count the support of C_1 to determine L_1.
L_1 := {frequent 1-itemsets};
$k := 2$; *// k represents the pass number//*
while $(L_{k-1} \neq \emptyset)$ ***do***
begin
$C_k :=$ ***gen_candidate_itemsets*** *with the given* L_{k-1}
prune(C_k)
for all transactions $t \in T$ ***do***
increment the count of all candidates in C_k that are contained in t;
$L_k :=$ All candidates in C_k with minimum support ;
$k := k + 1$;
end
Answer := $\cup_k L_k$;

Figure 7.1 Apriori algorithm

Example

Consider a database consisting of 9 transactions as given in Table 7.1. Find the frequent itemsets using apriori algorithm. Assume the minimum support count as 2.

- Scan the database to get the support of each candidate item
  C_1 = { {A} – 6, {B} – 7, {C} – 6, {D} – 2, {E} – 2 }
- Determine L_1 from C_1
  L_1 = { {A}, {B}, {C}, {D}, {E} }

Table 7.1 Database

Transaction ID	List of Items
1	A, B, E
2	B, D
3	B, C
4	A, B, D
5	A, C
6	B, C
7	A, C
8	A, B, C, E
9	A, B, C

- Generate C_2 from L_1 using apriori join step
  C_2 = { {A,B}, {A,C}, {A,D}, {A,E}, {B,C}, {B,D}, {B,E}, {C,D}, {C,E}, {D,E}}
- Scan the database to get the support of each candidate item of C_2
  C_2 = { {A,B} – 4, {A,C} – 4, {A,D} – 1, {A,E} – 2, {B,C} – 4, {B,D} – 2, {B,E} – 2, {C,D} – 0, {C,E} – 1, {D,E} – 0 }

- Determine L_2 from C_2
  L_2 = { {A,B}, {A,C}, {A,E}, {B,C}, {B,D}, {B,E} }
- Generate C_3 from L_2 using apriori join step
  C_3 = { {A,B,C}, {A,B,E}, {A,C,E},{ B,C,D}, {B,C,E}, {B,D,E} }
- Prune C_3 using apriori prune step
  C_3 = { {A,B,C}, {A,B,E} }
- Scan the database to get the support of each candidate item of C3
  C_3 = { {A,B,C} – 2, {A,B,E} – 2 }
- Determine L_3 from C_3
  L_3 = { {A,B,C}, {A,B,E} }
- Generate C_4 from L_3 using apriori join step
  C_4 = { {A,B,C, E} }
- Prune C4 using apriori prune step
  C_4 = { }
  L_4 = { }

The algorithm stops, as L_4 is empty.
Answer: Frequent itemsets: {{A}, {B}, {C}, {D}, {E}, {A, B}, {A, C}, {A, E}, {B, C}, {B,D}, {B,E}, {A,B,C}, {A, B, E}}

Association Rules

An association rule is an implication of the form X $\Rightarrow$ Y, where X & Y are transactions with set of items from a transactional database 'D'.

- The rule X $\Rightarrow$ Y holds in 'D' with confidence c if c% of transactions in D that contain X also contain Y
- The rule X $\Rightarrow$ Y has support s in D if s% of transactions in D contain X U Y

Find all rules that have support and confidence greater than user-specified minimum support and minimum confidence. The steps to generate association rules is as given below:

- For each frequent itemset *l*, generate all nonempty subsets of *l*.
- For every nonempty subset *s* of *l*, output the rule "$s \Rightarrow l\text{-}s$ " if (*support count*(*l*) / *support count*(*s*)) $\geq$ *min conf*, where *min conf* is the minimum confidence threshold.
- As the rules are generated from the frequent itemsets, each one automatically satisfies minimum support.

Example: Consider one of the frequent itemset {A, B, C} for the database given in Table 7.1 to generate association rule by considering the minimum confidence threshold as 75%

- *l* ={A, B, E} , Subsets of *l* ={{A, B},{A, E},{B, E},{A},{B},{E}}
- s = {A, B}, {A, B} $\Rightarrow$ {E} , conf({A, B}$\Rightarrow$ {E})=2/4=50%
- s = {A, E}, {A, E} $\Rightarrow$ {B} , conf({A, C} $\Rightarrow$ {B})=2/2=100%
- s = {B, E}, {B, E} $\Rightarrow$ {A} , conf({B, E} $\Rightarrow$ {A})=2/2=100%
- s = {A}, {A}$\Rightarrow$ {B, E} , conf({A} $\Rightarrow$ {B, E})=2/6=33%

- s = {B}, {B} $\Rightarrow$ {A,E} , conf({B} $\Rightarrow$ {A,E})=2/7=29%
- s = {E}, {E} $\Rightarrow$ {A,B} , conf({E} $\Rightarrow$ {A,B})=2/2=100%

Association rules satisfying the minimum support and threshold are as follows:
{A, E} $\Rightarrow$ {B}, {B, E} $\Rightarrow$ {A}, {E} $\Rightarrow$ {A, B}

Lab Exercise

1. Find the frequent itemsets by using the Apriori algorithm for a given transactional database and determine the association rules by considering suitable minimum support and confidence values

Additional Exercise

1. Implement partition based Apriori algorithm for a database stored as a file and find the association rules.

LAB NO: 8 **Date:**

K-MEANS ALGORITHM

Objective

1. To implement K-means algorithm for clustering

k-means algorithm

Clustering is a process of partitioning various objects into groups called as clusters, with the aim of having high intra-cluster similarity and low inter cluster similarity. It is an example of unsupervised learning. Clustering is a form of learning by observation, rather than learning by examples.

K-means is a well-known and commonly used partitioning method. The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intracluster similarity is high but the intercluster similarity is low. Cluster similarity is measured concerning the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or center of gravity.

The k-means algorithm proceeds as follows:
First, it randomly selects k of the objects, each of which initially represents a cluster mean or center. For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean. It then computes the new mean for each cluster. This process iterates until the criterion function converges. Typically, the square-error criterion is used, as given in equation 8.1.

$$E = \sum_{i=1}^{k} \sum_{p \in Ci} |p - mi|^2 \qquad (8.1)$$

where, E is the sum of the square error for all objects in the data set; p is the point in space representing a given object; and mi is the mean of cluster Ci (both p and mi are multidimensional). In other words, for each object in each cluster, the distance from the object to its cluster center is squared, and the distances are summed. This criterion tries to make the resulting k clusters as compact and as separate as possible. The k-means procedure is summarized in Figure 8.1
The algorithm attempts to determine k partitions that minimize the square-error function. It works well when the clusters are compact clouds that are well separated from one another. The method is relatively scalable and efficient in processing large data sets because the computational complexity of the algorithm is O(nkt), where n is the total number of objects, k is the number of clusters, and t is the number of iterations. Normally, k << n and t << n. The method often terminates at a local optimum.

The K-means method, however, can be applied only when the mean of a cluster is defined. The necessity for users to specify k, the number of clusters, in advance can be seen as a disadvantage. The K-means method is not suitable for discovering clusters with nonconvex shapes or clusters of very different size. It is sensitive to noise and outlier data points because a small number of such data can substantially influence the mean value.

Suppose that there is a set of objects located in space as depicted in the rectangle shown in Figure 8.2(a). Let k = 3; that is, the user would like the objects to be partitioned into three clusters. According to the algorithm in Figure 8.1, we arbitrarily choose three objects as the three initial cluster centers, where cluster centers are marked by a "+". Each object is distributed to a cluster based on the cluster center to which it is

the nearest. Such a distribution forms silhouettes encircled by dotted curves, as shown in Figure 8.2(a). Next, the cluster centers are updated. That is, the mean value of each cluster is recalculated based on the current objects in the cluster. Using the new cluster centers, the objects are redistributed to the clusters based on which cluster center is the nearest. Such a redistribution forms new silhouettes encircled by dashed curves, as shown in Figure 8.2(b).

Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;
(2) repeat
(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
(4) update the cluster means, i.e., calculate the mean value of the objects for each cluster;
(5) until no change;

Figure 8.1 K-means algorithm

.This process iterates, leading to Figure 8.2(c). The process of iteratively reassigning objects to clusters to improve the partitioning is referred to as iterative relocation. Eventually, no redistribution of the objects in any cluster occurs, and so the process terminates. The resulting clusters are returned by the clustering process.

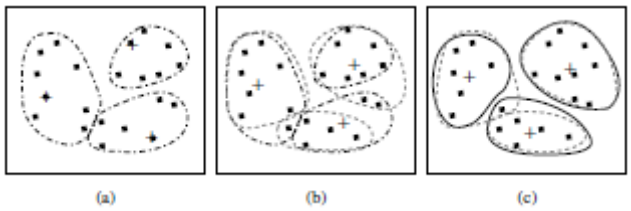

Figure 8.2 Clustering of a set of objects based on the k-means method

Lab Exercise

1. Implement K-means algorithm for a given dataset using Euclidean distance as a similarity measure**.**

Additional Exercise

1. Use Manhattan as a similarity measure to cluster the given dataset using K-means algorithm

Lab 9 **Date:**

DECISION TREE ID3 ALGORITHM FOR CLASSIFICATION

Objectives:

1. To understand the working of decision tree for classification
2. Implement ID3 algorithm to construct the decision tree.

Introduction:

ID3 builds a decision tree from a fixed set of examples. The resulting tree is used to classify future samples. The example has several attributes and belongs to a class (like yes or no). The leaf nodes of the decision tree contain the class name whereas a non-leaf node is a decision node. The decision node is an attribute test with each branch (to another decision tree) being a possible value of the attribute. ID3 uses information gain to help it decide which attribute goes into a decision node. The advantage of learning a decision tree is that a program, rather than a knowledge engineer, elicits knowledge from an expert.

How does ID3 decide which attribute is the best? A statistical property, called **information gain**, is used. Gain measures how well a given attribute separates training examples into targeted classes. The one with the highest information (information being the most useful for classification) is selected. In order to define gain, we first borrow an idea from information theory called entropy. Entropy measures the amount of information in an attribute.

Given a collection S of c outcomes, Entropy is calculated as given in equation 9.1

$$\text{Entropy}(S) = S\ -p(I)\ \log_2 p(I) \qquad (9.1)$$

where, p(I) is the proportion of S belonging to class I. S is over c. Log_2 is log base2. Note that S is not an attribute but the entire sample set. Gain(S, A) is information gain of example set S on attribute A is defined as given in equation 9.2.

$$\text{Gain}(S, A) = \text{Entropy}(S) - S\ ((|S_v| / |S|) * \text{Entropy}(S_v)) \qquad (9.2)$$

where,

S is each value v of all possible values of attribute A

S_v = subset of S for which attribute A has value v

$|S_v|$ = number of elements in S_v

|S| = number of elements in S

ID3 Algorithm:

ID3 (Examples, Target_Attribute, Attributes)

Create a root node for the tree

If all examples are positive, Return the single-node tree Root, with label = +.

If all examples are negative, Return the single-node tree Root, with label = -.

If number of predicting attributes is empty, then Return the single node tree Root, with label = most common value of the target attribute in the examples.

Otherwise Begin

A ← The Attribute that best classifies examples.

Decision Tree attribute for Root = A.

For each possible value, v_i, of A,

Add a new tree branch below Root, corresponding to the test A = v_i.

Let Examples(v_i) be the subset of examples that have the value v_i for A

If Examples(v_i) is empty

Then below this new branch add a leaf node with label = most common target value in the examples

Else below this new branch add the subtree ID3 (Examples(v_i), Target_Attribute, Attributes – {A})

End

Return Root

Consider the following example:

Table 9.1: Play ball dataset

Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Suppose Table 9.1 is a set of 14 examples in which one of the attributes is wind speed. The values of Wind can be *Weak* or *Strong*. The classification of these 14 examples are 9 YES and 5 NO. For attribute Wind, suppose there are 8 occurrences of Wind = Weak and 6 occurrences of Wind = Strong. For Wind = Weak, 6 of the examples are YES and 2 are NO. For Wind = Strong, 3 are YES and 3 are NO. Therefore,

$Gain(S,Wind) = Entropy(S) - (8/14)*Entropy(S_{weak}) - (6/14)*Entropy(S_{strong})$

$= 0.940 - (8/14)*0.811 - (6/14)*1.00$

$= 0.048$

$Entropy(S_{weak}) = - (6/8)*\log2(6/8) - (2/8)*\log2(2/8) = 0.811$

$Entropy(S_{strong}) = - (3/6)*\log2(3/6) - (3/6)*\log2(3/6) = 1.00$

For each attribute, the gain is calculated and the highest gain is used in the decision node. This process goes on until all data is classified perfectly or we run out of attributes.

Lab Exercises

1. Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use data set given in table 1 for building the decision tree.
2. Classify new samples using the rules obtained from the decision tree in exercise1.

Additional Exercises

1. Construct the decision tree for the following dataset of mushroom

   https://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/agaricus-lepiota.data

Lab 10 **Date:**

NAIVE BAYES CLASSIFIER

Objectives:

1. To understand the Naïve Bayesian classifier.
2. To build Naïve Bayes Gaussian classifier to classify data.
3. To build Multinomial Naïve Bayes Gaussian classifier to classify text.

Naïve Bayesian Gaussian Classifier

Introduction:

Naive Bayes is a supervised learning algorithm used for classification tasks. Hence, it is called as Naive Bayes Classifier. As other supervised learning algorithms, Naive Bayes uses features to make a prediction on a target variable. The key difference is that Naive Bayes assumes that features are independent of each other and there is no correlation between features

Bayes comes from the famous Bayes' Theorem of Thomas Bayes as shown in equation 10.1. To get a comprehensive understanding of Bayes' Theorem, we should talk about probability and conditional probability first.

$$P(A|B)=\frac{P(A).P(B)}{P(B)} \qquad (10.1)$$

Naive Bayes classifier calculates the probability of a class given a set of feature values (i.e. p(yi | x1, x2 , … , xn)). Input this into Bayes' theorem as given in equation 10.2.

$$P(yi|x1,x2,\ldots.xn)=\frac{P(x1,x2,\ldots.xn|yi).P(yi)}{P(x1,x2,\ldots.xn)} \qquad (10.2)$$

p(x1, x2 , … , xn | yi) means the probability of a specific combination of features given a class label. To be able to calculate this, we need extremely large datasets to have an estimate on the probability distribution for all different combinations of feature values. To overcome this issue, **naive bayes algorithm assumes that all features are independent of each other.** Furthermore, denominator (p(x1,x2, … , xn)) can be removed to simplify the equation because it only normalizes the value of conditional probability of a class given an observation (p(yi | x1,x2,… … , xn)).The probability of a class (p(yi)) is very simple to calculate as shown in equation 10.3.

$$P(yi)=\frac{number\ of\ observations\ with\ class\ yi}{number\ of\ all\ observations} \qquad (10.3)$$

Under the assumption of features being independent, **p(x1, x2 ,.. , xn | yi)** can be written as given in equation 10.4.

$$x1,x2,\ldots.xn|yi)=P(x1|yi).P(x2|yi)\ldots\ldots\ldots P(xn|yi) \qquad (10.4)$$

The conditional probability for a single feature given the class label (i.e. p(x1 | yi)) can be more easily estimated from the data. The algorithm needs to store probability distributions of features for each class independently. The type of distributions depends on the characteristics of features:

1. For binary features (Y/N, True/False, 0/1): Bernoulli distribution
2. For discrete features (i.e. word counts): Multinomial distribution
3. For continuous features: Gaussian (Normal) distribution

Multinomial Naive Bayes Classifier to Classify Text

Introduction:

Multinomial naive Bayes works similar to Gaussian naive Bayes, however the features are assumed to be multinomial distributed. In practice, this means that this classifier is commonly used when we have discrete data (e.g. movie ratings ranging 1 and 5).

For sentiment analysis, a Naive Bayes classifier is one of the easiest and most effective ways to hit the ground running for sentiment analysis.

Deriving the prior probability of a class is rather trivial, as it is simply the sum of all words in *doc* that are assigned to *c* divided by the number of words in *doc* as shown in equation 10.5.

$$\hat{P}(c) = \frac{N_c}{N_{doc}} \qquad (10.5)$$

How do we learn all of the probabilities of that make up each feature? The solution is again rather simple: for a given word *w* in words *W* from *d* we count how many of *w* belong in class *c*. We then divide this by all the words in *d* that belong to *c*. This gives us a probability for a word *w* given *c* as in equation 10.6.

$$\hat{P}(w_i|c) = \frac{count(w_i,c)+1}{\sum_{w\in V}(count(w,c)+1)} = \frac{count(w_i,c)+1}{(\sum_{w\in V} count(w,c))+|V|} \qquad (10.6)$$

Computing Error Rate, Accuracy, Precision and Recall from confusion matrix:

A confusion matrix is a technique for summarizing the performance of a classification algorithm. The number of correct and incorrect predictions are summarized with count values and broken down by each class. These numbers are then organized into a table, or a matrix as shown in Table 10.1.

Table 10.1: Confusion matrix for 2-class problem

		PREDICTED	
		Positive	**negative**
ACTUAL	**Positive**	True Positive	False negative
	negative	False Positive	True Negative

From confusion matrix one can get the following measures.

1. Accuracy (ACC) is calculated as the number of all correct predictions divided by the total number of the dataset as in equation 10.7.

Accuracy=(TP+TN)/(TP+TN+FP+FN) (10.7)

2. Error rate (ERR) is calculated as the number of all incorrect predictions divided by the total number of the dataset as in equation 10.8.

$$\text{Error rate} = (FP+FN)/(TP+TN+FP+FN) \quad (10.8)$$

3. Precision evaluates the fraction of correct classified instances among the ones classified as positive as in equation 10.9.

$$\text{Precision} = TP/(TP+FP) \quad (10.9)$$

4. Recall is a metric that quantifies the number of correct positive predictions made out of all positive predictions that could have been made as in equation 10.10.

$$\text{Recall} = TP/(TP+FN) \quad (10.10)$$

sklearn to fetch measures

sklearn can be used to fetch these measures using the following codes:

```
from sklearn import metrics
metrics.confusion_matrix
metrics.classification_report(predicted, expected)
```

Lab Exercises

1. Write a program to implement the naïve Bayesian classifier(Gaussian) for *Pima indians diabetes* training data set. Compute the accuracy of the classifier, considering few test data sets.
2. Use the naïve Bayesian(Multinomial) Classifier model to perform text classification task on 20newsgroups dataset.
3. Calculate the accuracy, precision, and recall for your data set on the confusion matrix obtained.

Additional Exercises

1. Write a program to implement the naïve Bayesian classifier(Gaussian) for scikit-learn wine training data set. Compute the accuracy of the classifier, considering few test data sets.
2. Use the naïve Bayesian(Multinomial) Classifier model to perform text classification task on Reuters News Dataset.

LAB NO. 11 **Date:**

MINI PROJECT – IMPLEMENTATION

Objective

1. **To implement the miniproject**

Students are required to implement the mini project.

LAB NO. 12 **Date:**

MINI PROJECT – PROGRESS

Objective

1. **To discuss the progress of miniproject implementation**

Students are required to show the progress of mini project implementation and discuss with faculty

REFERENCES

1. Jiawei Han and Micheline Kamber, “Data Mining- Concepts and Techniques”, 3rd Edition, Morgan Kaufmann Publishers, 2011.

2. Arun K. Pujari, ”Data Mining Techniques”, University press, 2006.

3. G.K. Gupta,”Introduction to data mining with Case Studies”, Easter Economy edition, Prentice Hall of India,2006.

4. Pang-Ning Tan,Michael Steinbach and Vipin Kumar,”Introduction to Data Mining” Pearson Education,2007.

5. Infosphere Documentation.

6. Rapid miner Documentation.