DATA Warehouse LAB Manual

data warehousing lab (Anna University)

EX.NO:1 Data exploration and integration with WEKA

AIM:
Work with Weka tool in Data exploration and integration using classification,clustering algorithms in the given training data and test with the unknown sample. Also experiment with different scenarios and large data set.

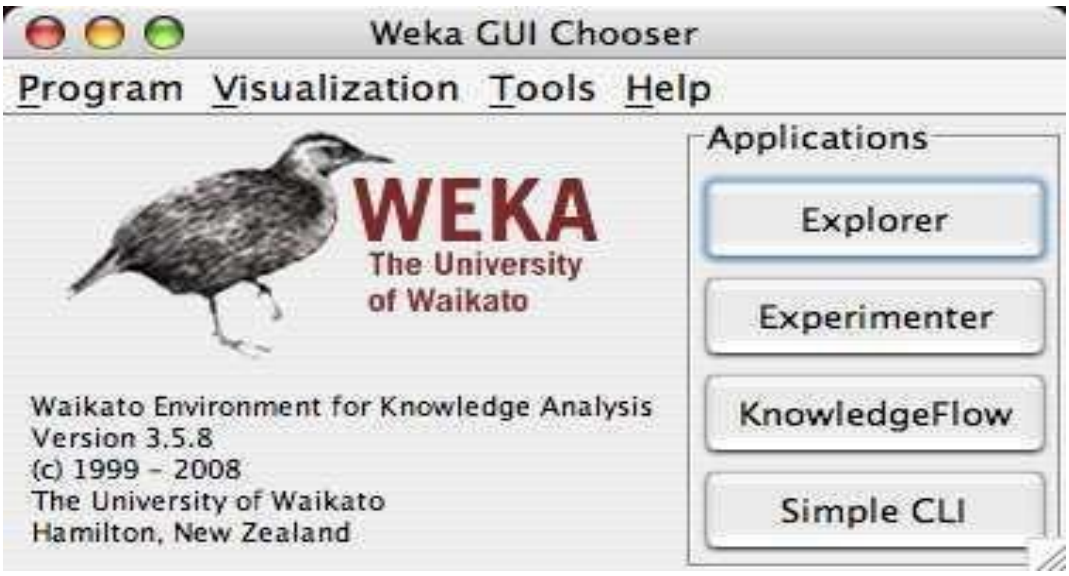
INTRODUCTION:

WEKA is open source java code created by researchers at the University of Waikato in New Zealand. It provides many different machine learning algorithms, including the following classifiers:

- Decision tree (j4.8, an extension of C4.5)
- MLP, aka multiple layer perceptron (a type of neural net)
- Naïve bayes
- Rule induction algorithms such as JRip
- Support vector machine
- And many more…

The GUI WEKA:

The Weka GUI Chooser (class weka.gui.GUIChooser) provides a starting point for launching Weka's main GUI applications and supporting tools. If one prefers a MDI ("multiple document interface") appearance, then this is provided by an alternative launcher called "Main" (class weka.gui.Main).The GUI Chooser consists of four buttons—one for each of the four major Weka applications—and four menus.



The buttons can be used to start the following applications:

• ***Explorer:***
An environment for exploring data with WEKA (the rest of this documentation deals With this application in more detail).

• ***Experimenter*:**
An environment for performing experiments and conducting statistical tests between learning schemes.
• ***Knowledge Flow:***
This environment supports essentially the same functions as the Explorer but with a drag-and-drop interface. One advantage is that it supports incremental learning.
• ***Simple CLI:***
Provides a simple command-line interface that allows direct execution of WEKA commands for operating systems that do not provide their own command line interface.

WEKA Explorer:

The user interface Section Tabs

At the very top of the window, just below the title bar, is a row of tabs. When the Explorer is first started only the first tab is active; the others are grayed out. This is because it is necessary to open (and potentially pre-process) a data set before starting to explore the data.
The tabs are as follows:
1. *Preprocess.* Choose and modify the data being acted on.
2. *Classify.* Train and test learning schemes that classify or perform regression.
3. *Cluster.* Learn clusters for the data.
4. *Associate.* Learn association rules for the data.
5. *Select attributes*. Select the most relevant attributes in the data.
6. *Visualize.* View an interactive 2D plot of the data.
Once the tabs are active, clicking on them flicks between different screens, on which the respective actions can be performed. The bottom area of the window (including the status box, the log button, and the Weka bird) stays visible regardless of which section you are in. The Explorer can be easily extended with custom tabs.

The buttons can be used to start the following applications:

• ***Explorer:***
An environment for exploring data with WEKA (the rest of this documentation deals With this application in more detail).
• ***Experimenter*:**
An environment for performing experiments and conducting statistical tests between learning schemes.
• ***Knowledge Flow:***
This environment supports essentially the same functions as the Explorer but with a drag-and-drop interface. One advantage is that it supports incremental learning.
• ***Simple CLI:***
Provides a simple command-line interface that allows direct execution of WEKA commands for operating systems that do not provide their own command line interface.

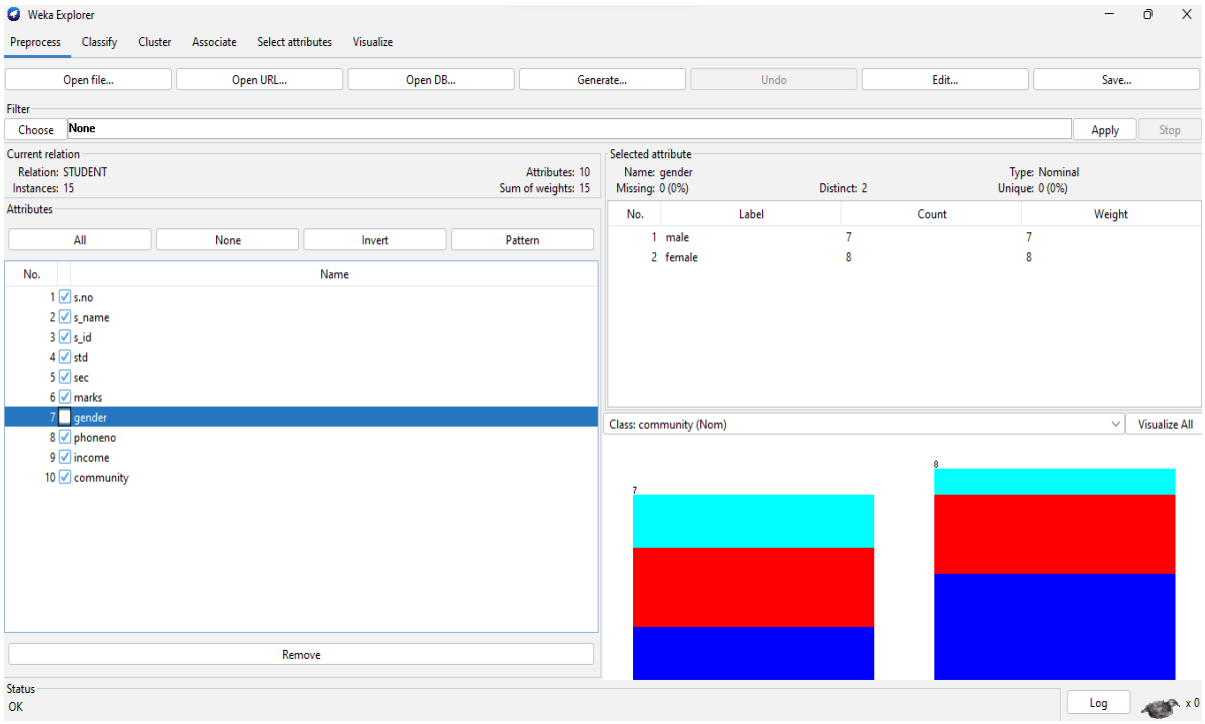
PROGRAM

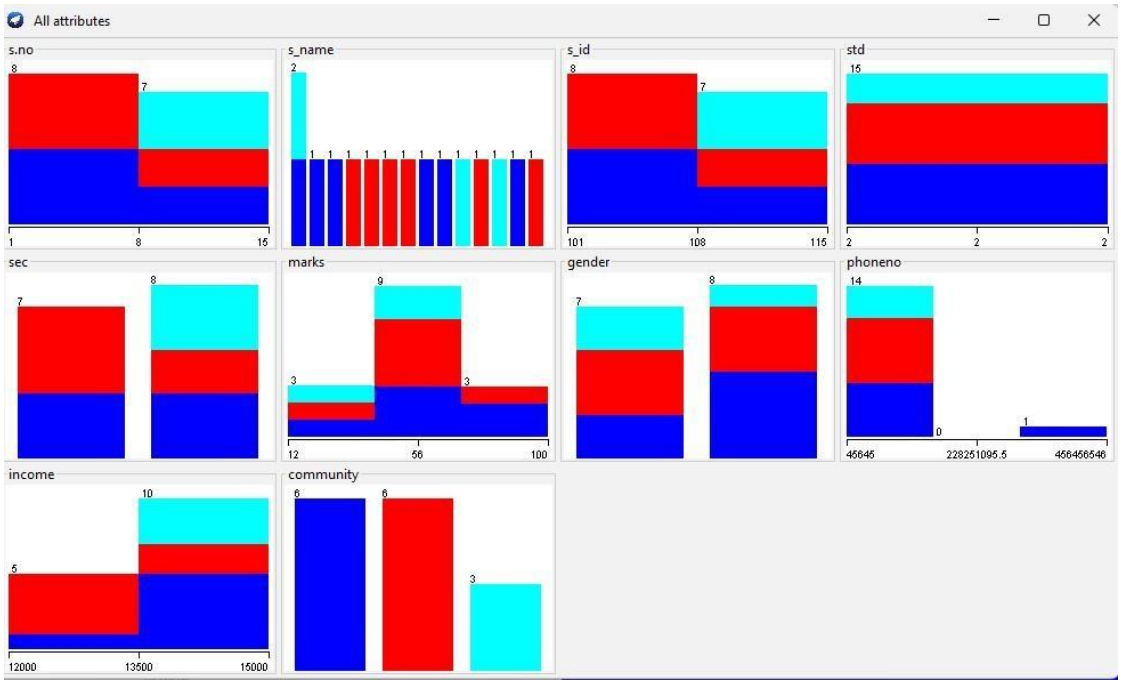
```
@RELATION STUDENT


@ATTRIBUTE s.no numeric
```

@ATTRIBUTE s_name {barath,gayathri,laya,suganya,harish,minion,prdhi,kiruthi,ashvi,bansa,rajesh,elakkiya,palani, anu}
@ATTRIBUTE s_id numeric
@ATTRIBUTE std numeric
@ATTRIBUTE sec {A,B}
@ATTRIBUTE marks numeric
@ATTRIBUTE gender {male,female}
@ATTRIBUTE phoneno numeric
@ATTRIBUTE income numeric
@ATTRIBUTE community {bc,mbc,sc}

@DATA
1,barath,101,2,A,100,male,456546,12000,bc
2,gayathri,102,2,A,50,female,456456,14000,bc
3,laya,103,2,A,45,female,123213,15000,bc
4,suganya,104,2,A,78,female,1321321,12000,mbc
5,harish,105,2,A,58,male,12312321,12000,mbc
6,minion,106,2,A,65,male,21665456,12000,mbc
7,prdhi,107,2,A,12,female,53546456,14000,mbc
8,ashvi,108,2,B,100,female,456456546,15000,bc
9,bansa,109,2,B,65,male,54645645,15000,sc
10,barath,110,2,B,35,male,54654654,15000,sc
11,kiruthi,111,2,B,26,female,456456,14000,bc
12,rajesh,112,2,B,55,male,45645654,12000,mbc
13,elakkiya,113,2,B,55,female,54654654,14000,sc
14,palani,114,2,B,48,male,45645645,15000,bc
15,anu,115,2,B,52,female,45645,14000,mbc

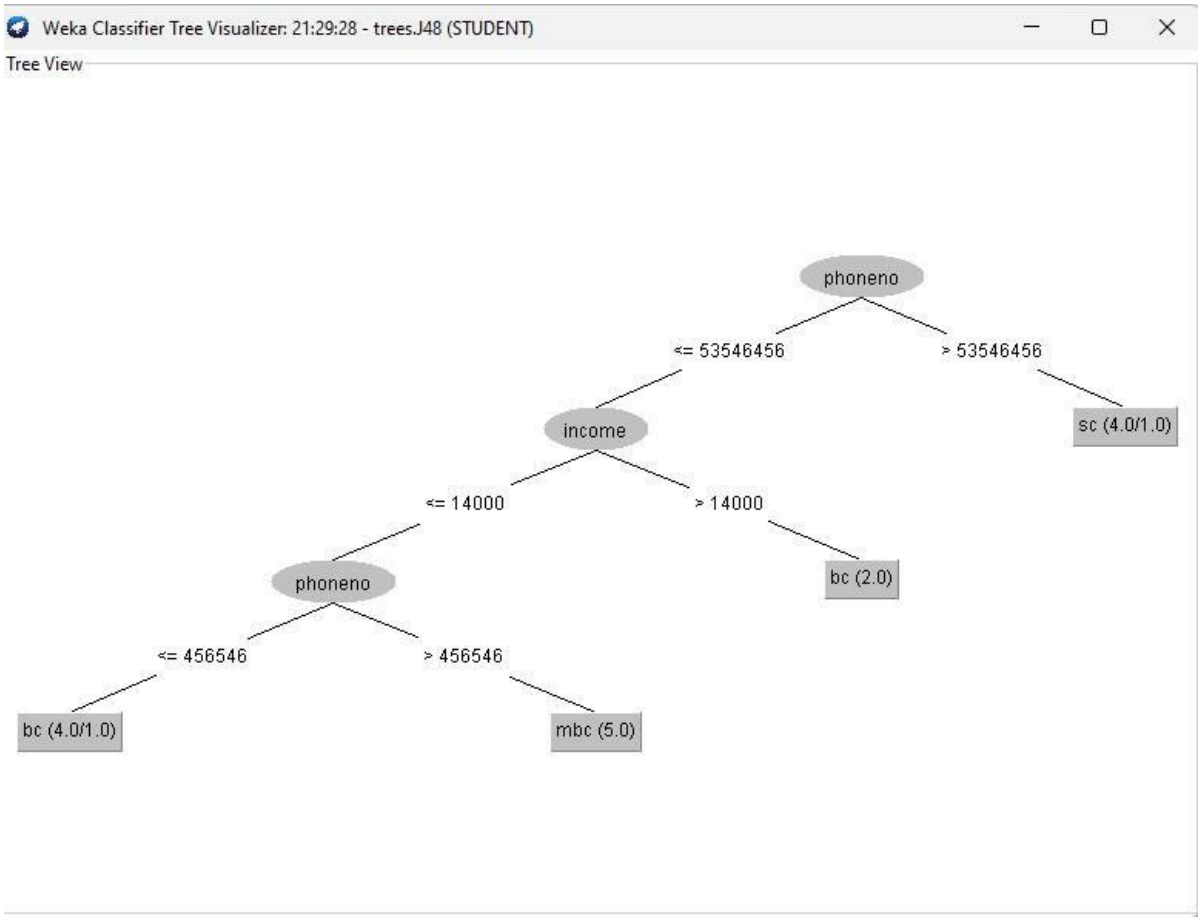
OUTPUT:

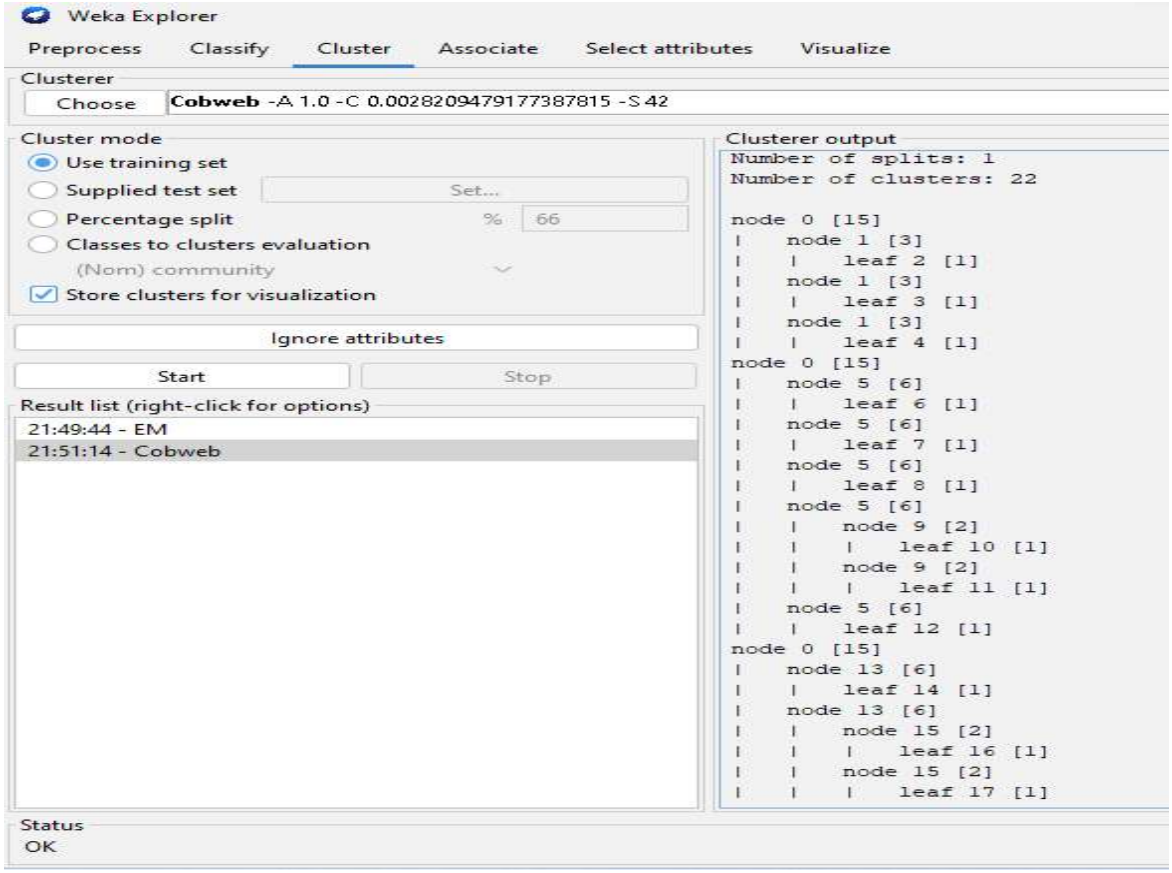


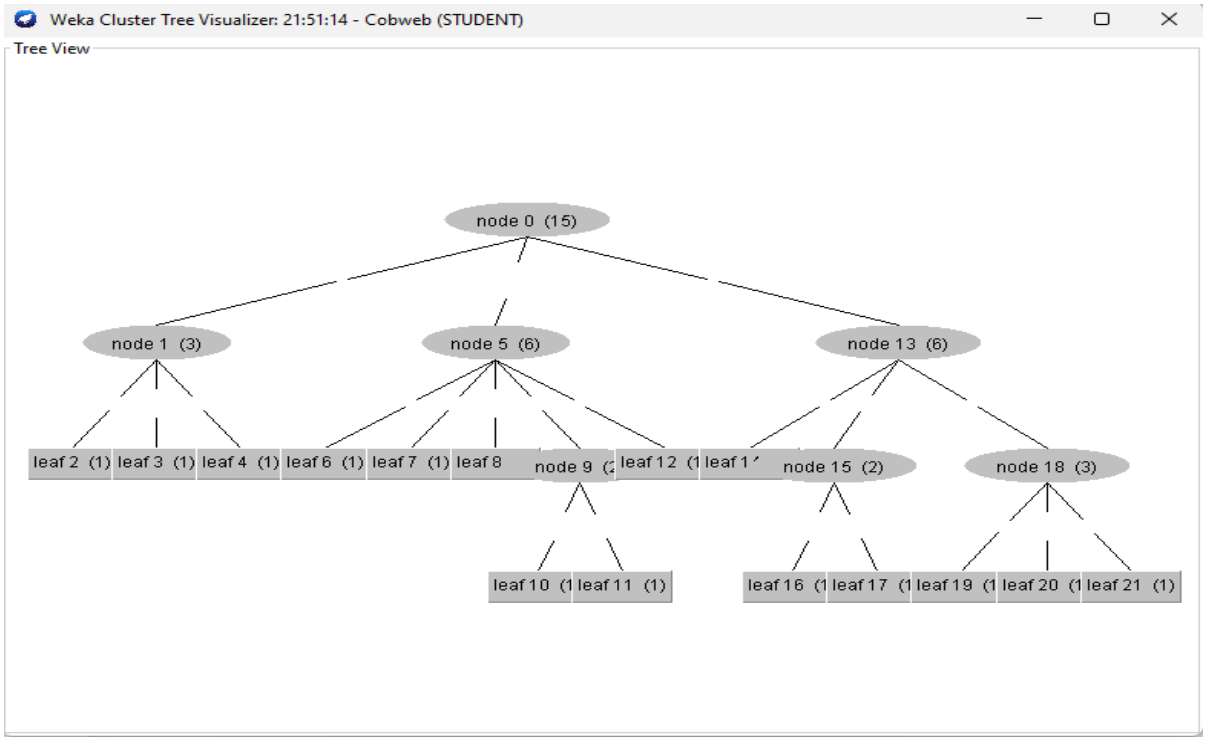


s.no	name	s_id	std	sec	marks	gender	phoneno	income	community
1	barath	101	2	A	100	male	456546	12000	bc
2	gayathri	102	2	A	50	female	456456	14000	bc
3	laya	103	2	A	45	female	123213	15000	bc
4	suganya	104	2	A	78	female	1321321	12000	mbc
5	harish	105	2	A	58	male	12312321	12000	mbc
6	minion	106	2	A	65	male	21665456	12000	mbc
7	prdhi	107	2	A	12	female	53546456	14000	mbc
8	ashvi	108	2	B	100	female	5241	15000	bc
9	bansa	109	2	B	65	male	54645645	15000	sc
10	barath	110	2	B	35	male	54654654	15000	sc
11	kiruthi	111	2	B	26	female	456456	14000	bc
12	rajesh	112	2	B	55	male	45645654	12000	mbc
13	elakkiya	113	2	B	55	female	54654654	14000	sc
14	palani	114	2	B	48	male	45645645	15000	bc
15	anu	115	2	B	52	female	45645	14000	mbc

Classification – decision tree (j48)







RESULT:

The classification and clustering algorithms using the given training data and test with the unknown sample by using “Weka tool” is successfully implemented.

EX.NO:2 APPLY WEKA TOOL FOR DATA VALIDATION

AIM

Work with Weka tool for data validation in the classification,clustering algorithms to the given training data and test with the unknown sample.

INTRODUCTION

Data validation processes check for the validity of the data. Using a set of rules, it checks whether the data is within the acceptable values defined for the field or not. The system ensures the inputs stick to the set rules, for instance, the type, uniqueness, format, or consistency of the data. The data validation checks are employed by declaring integrity rules through business rules. This enforcement of rules needs to be done at the start of the process to ensure the system does not accept any invalid data.

PROGRAM

```
@RELATION WEATHER

@ATTRIBUTE Day {D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,D12,D13,D14}
@ATTRIBUTE Outlook {sunny,overcast,rain}
@ATTRIBUTE Temp {hot,cool,mild}
@ATTRIBUTE Humidlty {high,normal}
@ATTRIBUTE Wind {weak,strong}
@ATTRIBUTE PlayTennis {no,yes}


@DATA
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
```

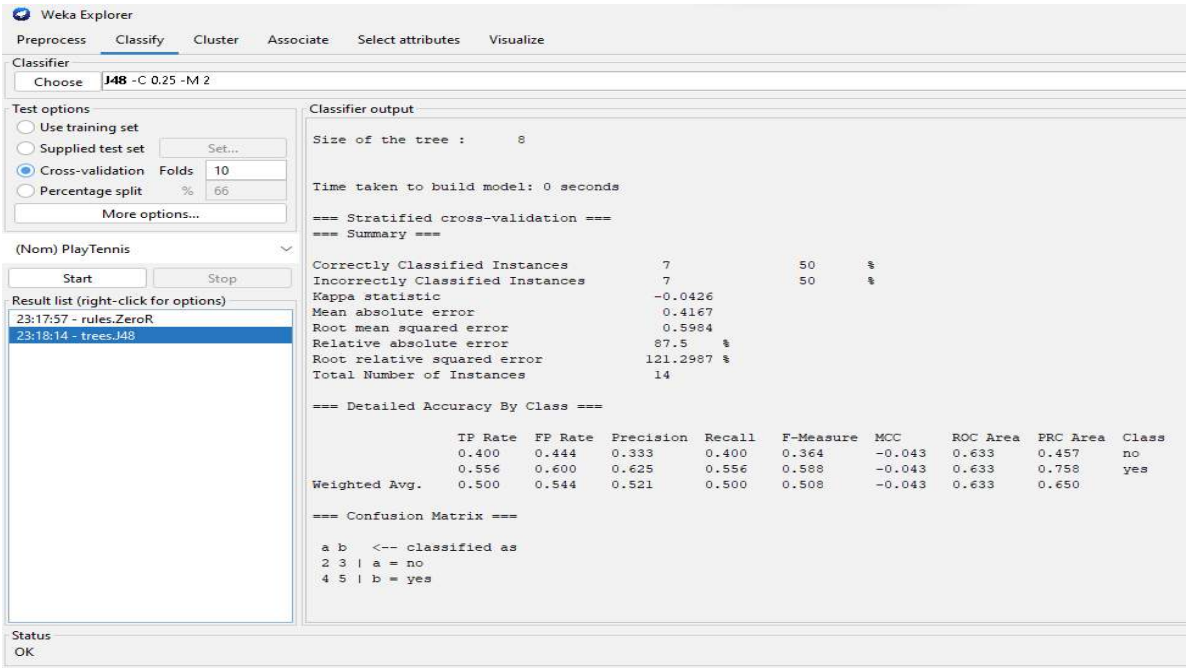
OUTPUT:

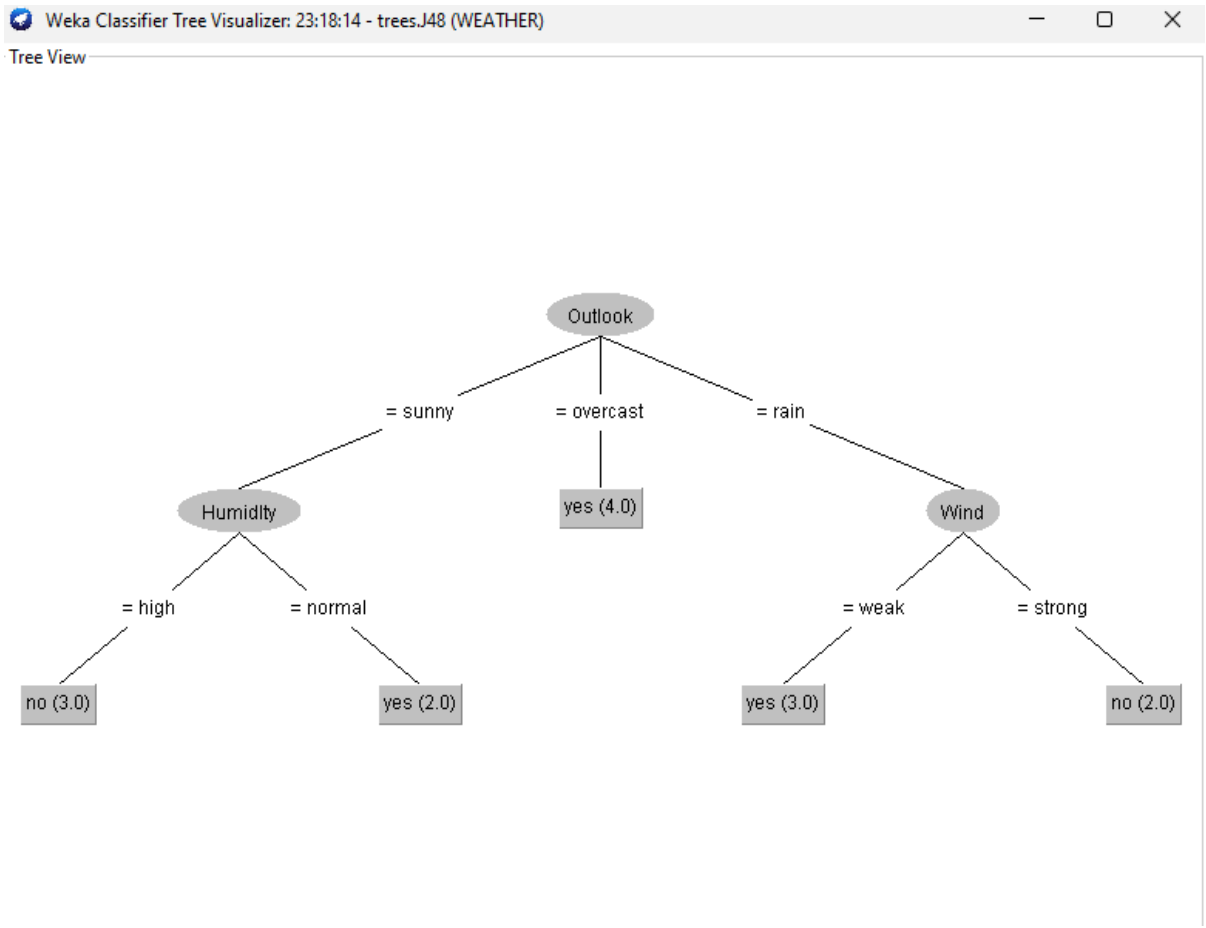
Day	Outlook	Temp	Humidlty	Wind	PlayTennis
D1	sunny	hot	high	weak	no
D2	sunny	hot	high	strong	no
D3	overcast	hot	high	weak	yes
D4	rain	mild	high	weak	yes
D5	rain	cool	normal	weak	yes
D6	rain	cool	normal	strong	no
D7	overcast	cool	normal	strong	yes
D8	sunny	mild	high	weak	no
D9	sunny	cool	normal	weak	yes
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

Classification

In machine learning and statistics, **classification** is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known.

Decision tree- id3 algorithm numerical algorithm





RESULT:

The classification algorithms using the given training data and test with the unknown sample by using “Weka tool” is successfully implemented.

EX.NO:3 PLAN THE ARCHITECTURE FOR REAL TIME APLLICATION

AIM:

To write and implement the architecture for real time application using iris dataset.

PROCEDURE:

This data sets consists of 3 different types of irises' (Setosa, Versicolour, and Virginica) petal and sepal length, stored in a 150x4 numpy.ndarray

The rows being the samples and the columns being: Sepal Length, Sepal Width, Petal Length and Petal Width.

The below plot uses the first two features. See here for more information on this dataset.

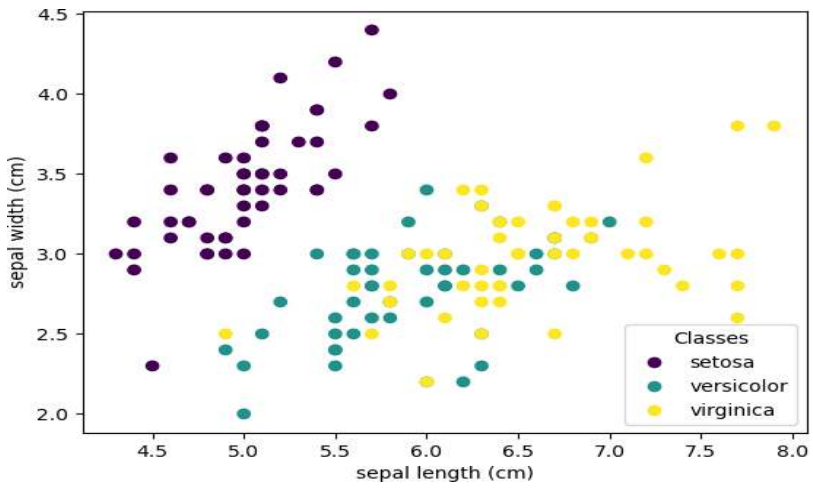
Loading the iris dataset

from sklearn import datasets

iris= datasets.load_iris()

Scatter Plot of the Iris dataset

```
import matplotlib.pyplot as plt_, ax = plt.subplots()
scatter = ax.scatter(iris.data[:, 0], iris.data[:, 1], c=iris.target)
ax.set(xlabel=iris.feature_names[0], ylabel=iris.feature_names[1])
_ = ax.legend(
    scatter.legend_elements()[0],     iris.target_names,     loc="lower
right", title="Classes")
```



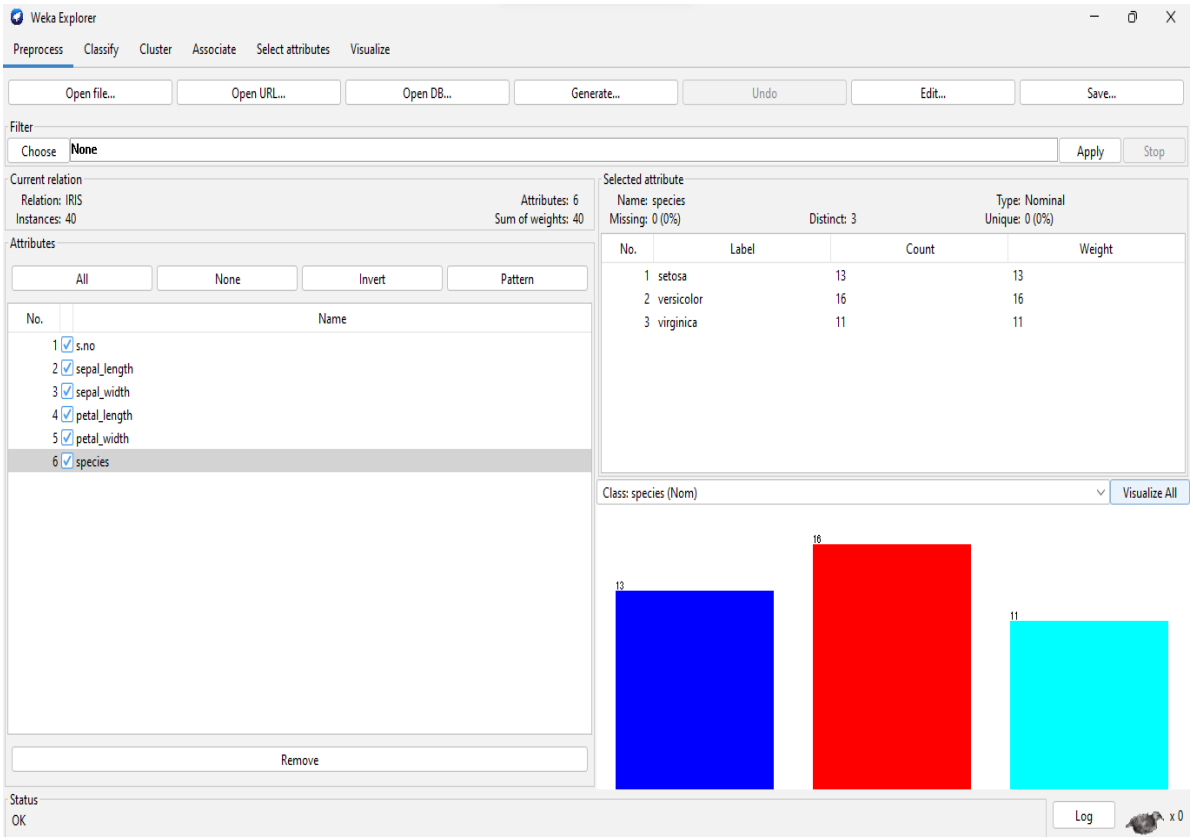
Each point in the scatter plot refers to one of the 150 iris flowers in the dataset, with the color indicating their respective type (Setosa, Versicolour, and Virginica). You can already see a pattern regarding the Setosa type, which is easily identifiable based on its short and wide sepal. Only considering these 2 dimensions, sepal width and length, there's still overlap between the Versicolor and Virginica types.

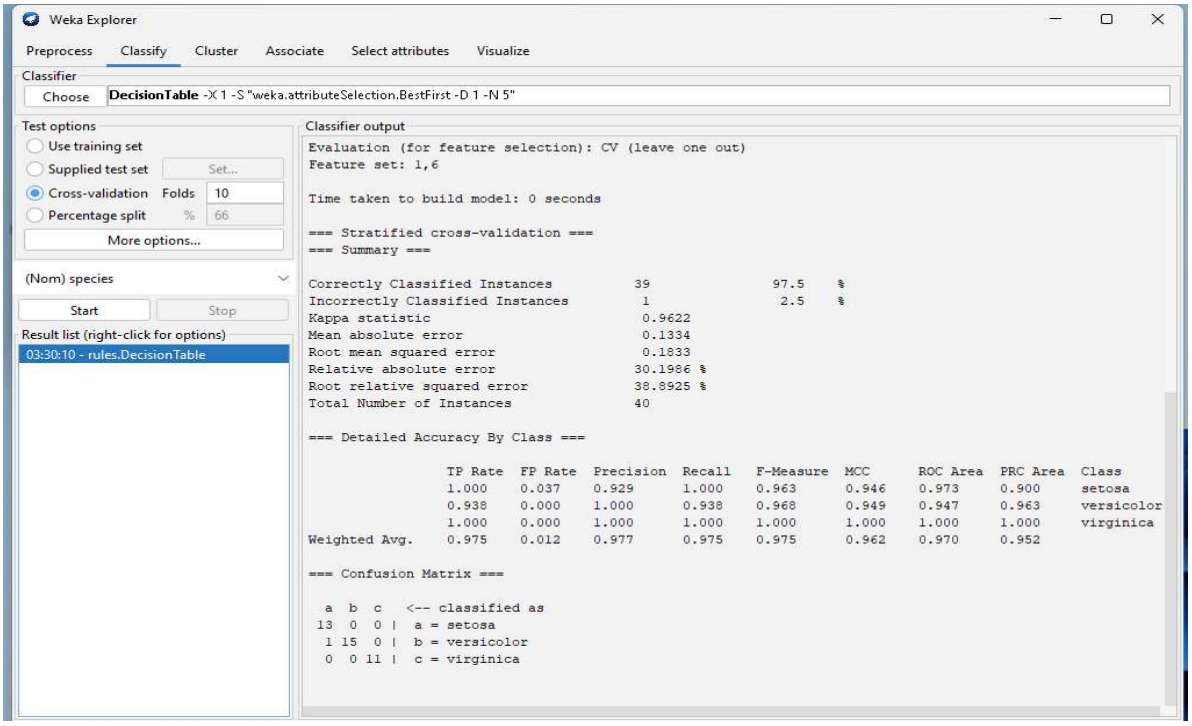
PROGRAM:

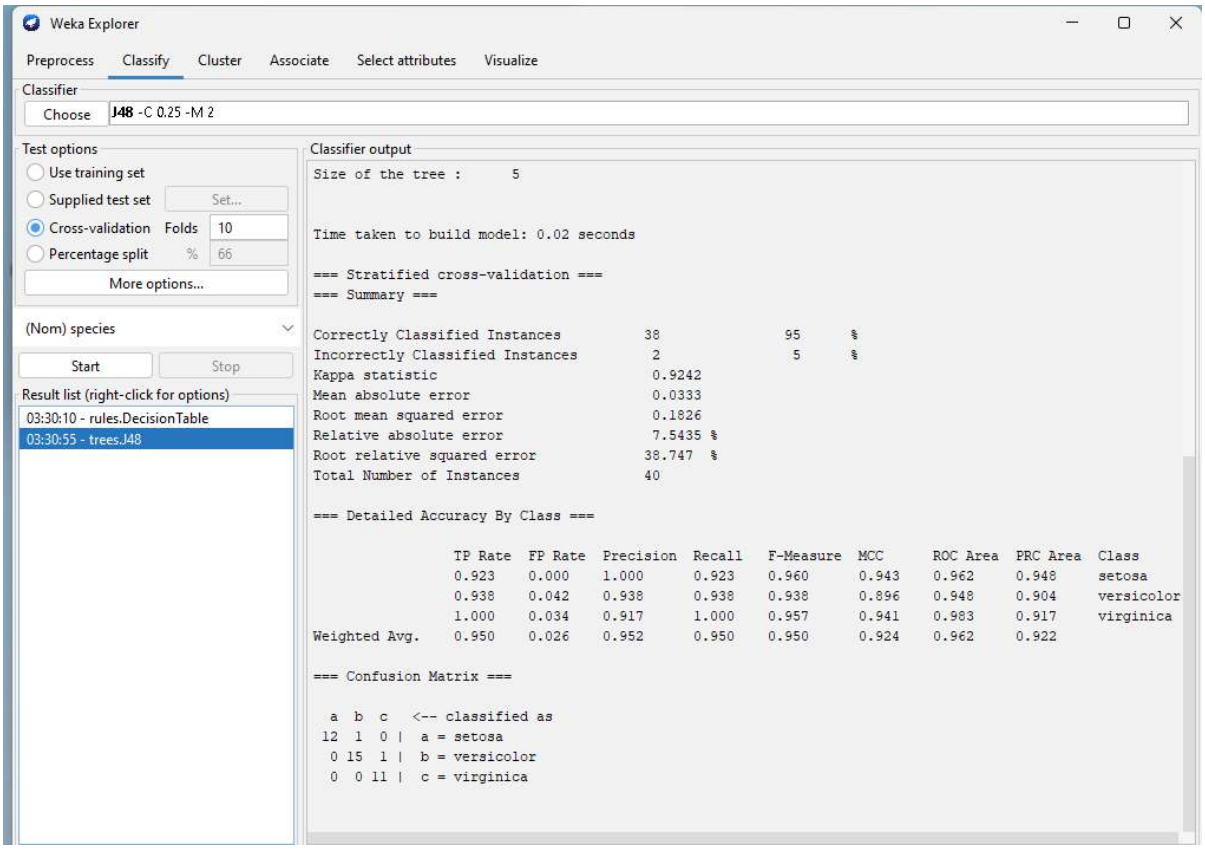
s.no	sepal_len	sepal_wi	petal_len	petal_wic	species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3	1.4	0.1	setosa
14	7	3.2	4.7	1.4	versicolor
15	6.4	3.2	4.5	1.5	versicolor
16	6.9	3.1	4.9	1.5	versicolor
17	5.5	2.3	4	1.3	versicolor
18	6.5	2.8	4.6	1.5	versicolor
19	5.7	2.8	4.5	1.3	versicolor
20	6.3	3.3	4.7	1.6	versicolor
21	4.9	2.4	3.3	1	versicolor
22	6.6	2.9	4.6	1.3	versicolor
23	5.2	2.7	3.9	1.4	versicolor

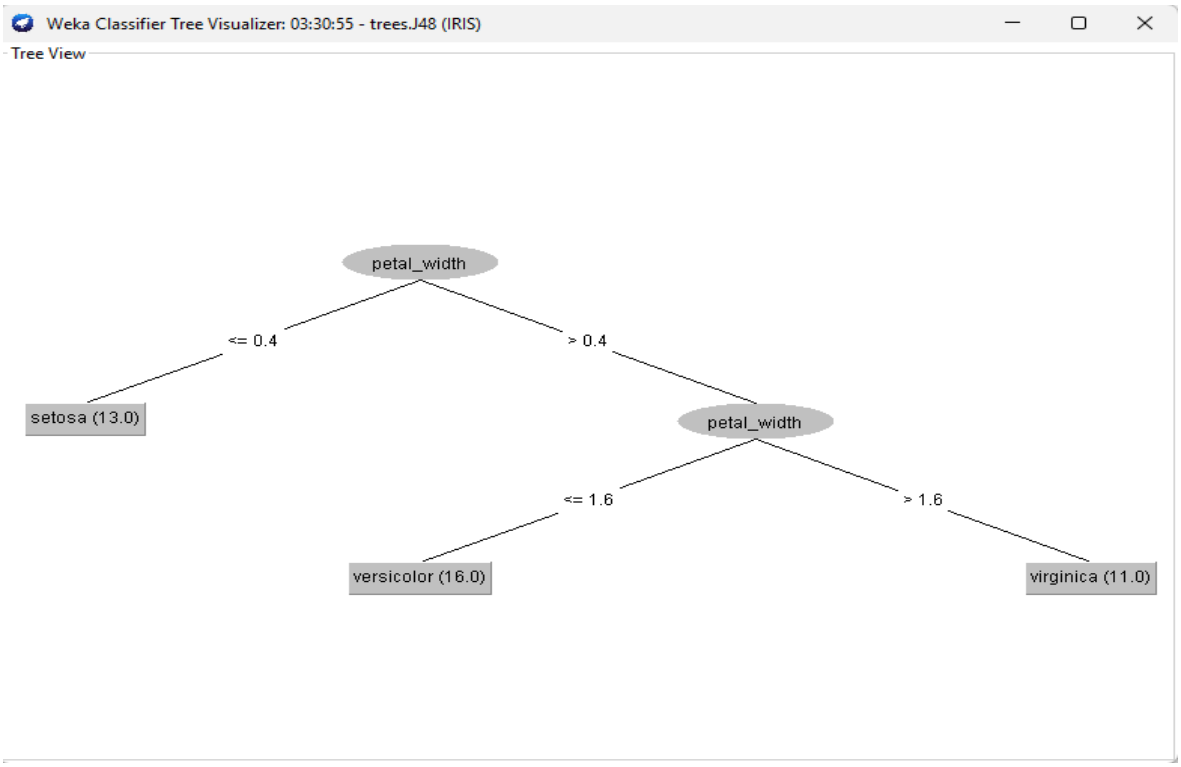
24	5	2	3.5	1	versicolor
25	5.9	3	4.2	1.5	versicolor
26	6	2.2	4	1	versicolor
27	6.3	3.3	6	2.5	virginica
28	5.8	5.8	2.7	1.9	virginica
29	7.1	3	5.9	2.1	virginica
30	6.3	2.9	5.6	1.8	virginica
31	6.5	3	5.8	2.2	virginica
32	7.6	3	6.6	2.1	virginica
33	4.9	2.5	4.5	1.7	virginica
34	7.3	2.9	6.3	1.8	virginica
35	6.7	2.5	5.8	1.8	virginica
36	7.2	3.6	6.1	2.5	virginica
37	6.5	3.2	5.1	2	virginica
38	6.2	2.9	4.3	1.3	versicolor
39	5.1	2.5	3	1.1	versicolor
40	5.7	2.8	4.1	1.3	versicolor

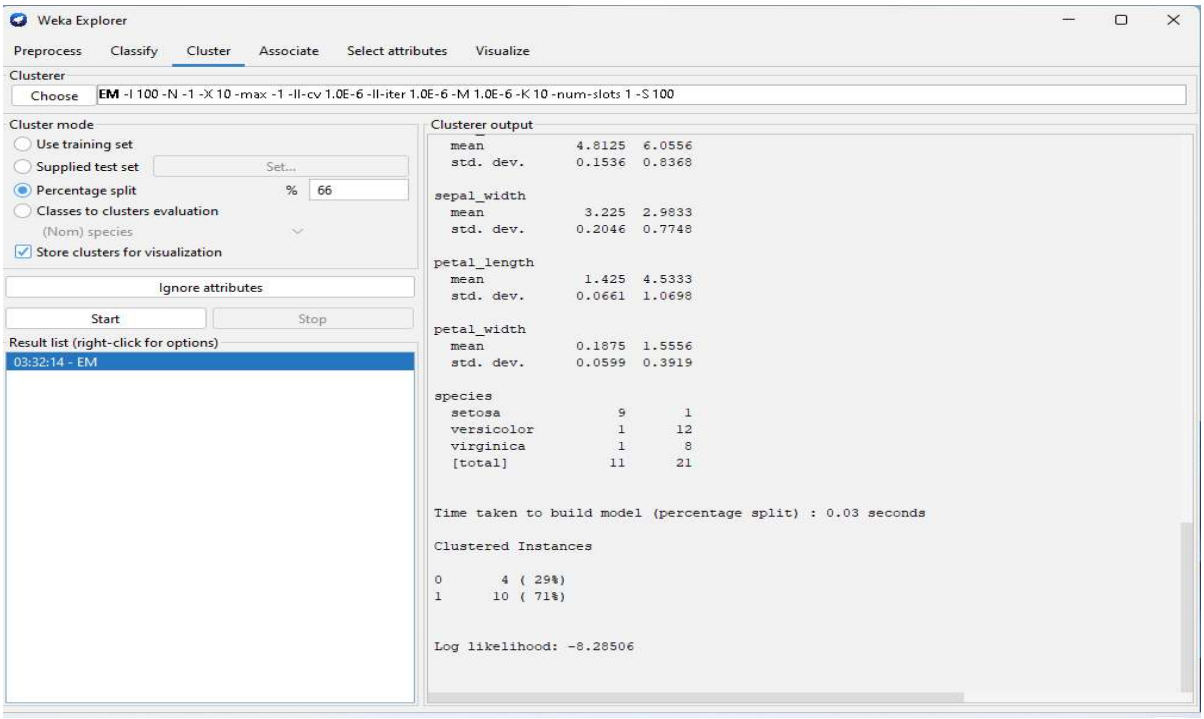
OUTPUT:











RESULT:

Thus the implementation of real time application using iris dataset.

EX.NO:4 WRITE THE QUERY FOR SCHEMA DEFINITION

AIM:

To write the query for schema definition for data warehouse

Data Warehouse Schema
In a data warehouse, a schema is used to define the way to organize the system with all the database entities (fact tables, dimension tables) and their logical association.

Here are the different types of Schemas in DW:

1. Star Schema
2. SnowFlake Schema
3. Galaxy Schema
4. Star Cluster Schema

#1) Star Schema
This is the simplest and most effective schema in a data warehouse. A fact table in the center surrounded by multiple dimension tables resembles a star in the Star Schema model.

The fact table maintains one-to-many relations with all the dimension tables. Every row in a fact table is associated with its dimension table rows with a foreign key reference.

Due to the above reason, navigation among the tables in this model is easy for querying aggregated data. An end-user can easily understand this structure. Hence all the Business Intelligence (BI) tools greatly support the Star schema model.

While designing star schemas the dimension tables are purposefully de-normalized. They are wide with many attributes to store the contextual data for better analysis and reporting.

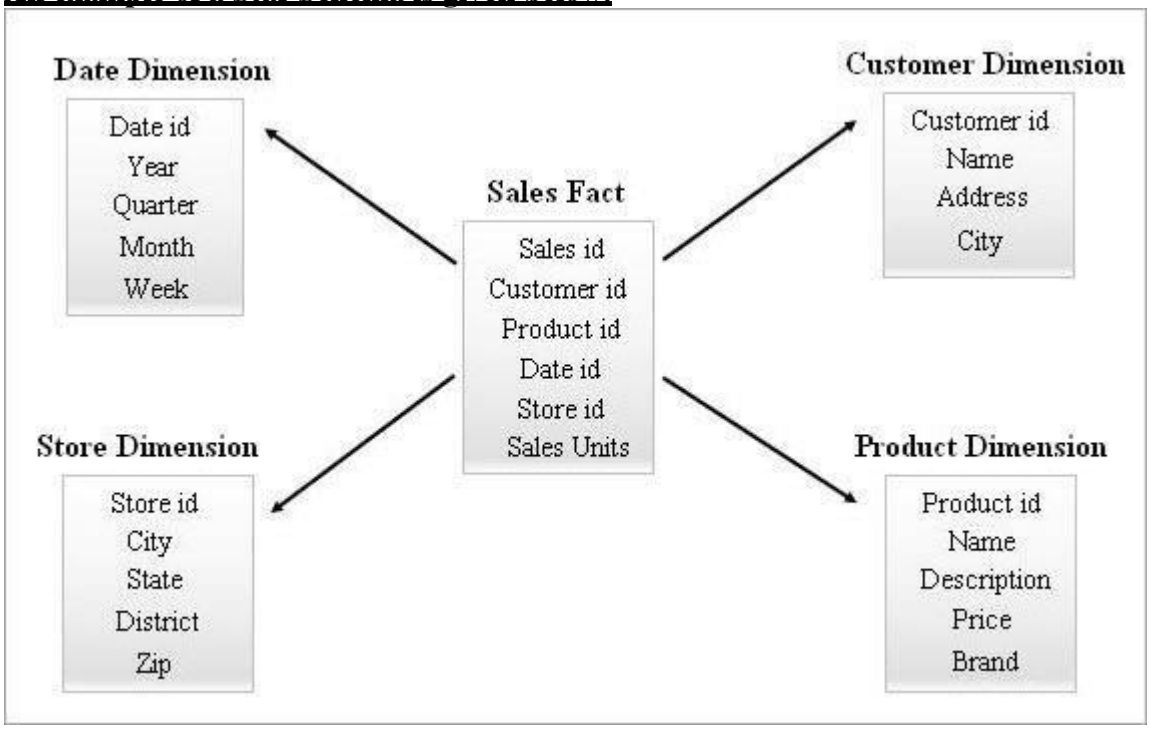
Benefits Of Star Schema

- Queries use very simple joins while retrieving the data and thereby query performance is increased.
- It is simple to retrieve data for reporting, at any point of time for any period.
-

Disadvantages Of Star Schema

- If there are many changes in the requirements, the existing star schema is not recommended to modify and reuse in the long run.
- Data redundancy is more as tables are not hierarchically divided.

An example of a Star Schema is given below.



Querying A Star Schema

An end-user can request a report using Business Intelligence tools. All such requests will be processed by creating a chain of "SELECT queries" internally. The performance of these queries will have an impact on the report execution time.

From the above Star schema example, if a business user wants to know how many Novels and DVDs have been sold in the state of Kerala in January in 2018, then you can apply the query as follows on Star schema tables:

```
SELECT   pdim.NameProduct_Name,
          Sum(sfact.sales_units) Quanity_Sold
FROM     Product pdim,
          Sales sfact,
          Store sdim,
          Dateddim
WHEREsfact.product_id = pdim.product_id
          ANDsfact.store_id = sdim.store_id
          ANDsfact.date_id = ddim.date_id
```

```
            ANDsdim.state = 'Kerala'
SELECT    pdim.NameProduct_Name,
            Sum(sfact.sales_units) Quanity_Sold
FROM      Product pdim,
            Sales sfact,
            Store sdim,
            Dateddim
WHEREsfact.product_id = pdim.product_id
            ANDsfact.store_id = sdim.store_id
            ANDsfact.date_id = ddim.date_id
            ANDsdim.state = 'Kerala'
            ANDddim.month = 1
            ANDddim.year    = 2018
            ANDpdim.Namein('Novels', 'DVDs')
GROUPBYpdim.Name
```

Results:

Product_Name	**Quantity_Sold**
Novels	12,702
DVDs	32,919

Hope you understood how easy it is to query a Star Schema.

#2) SnowFlake Schema

Star schema acts as an input to design a SnowFlake schema. Snow flaking is a process that completely normalizes all the dimension tables from a star schema.

The arrangement of a fact table in the center surrounded by multiple hierarchies of dimension tables looks like a SnowFlake in the SnowFlake schema model. Every fact table row is associated with its dimension table rows with a foreign key reference.

While designing SnowFlake schemas the dimension tables are purposefully normalized. Foreign keys will be added to each level of the dimension tables to link to its parent attribute. The complexity of the SnowFlake schema is directly proportional to the hierarchy levels of the dimension tables.

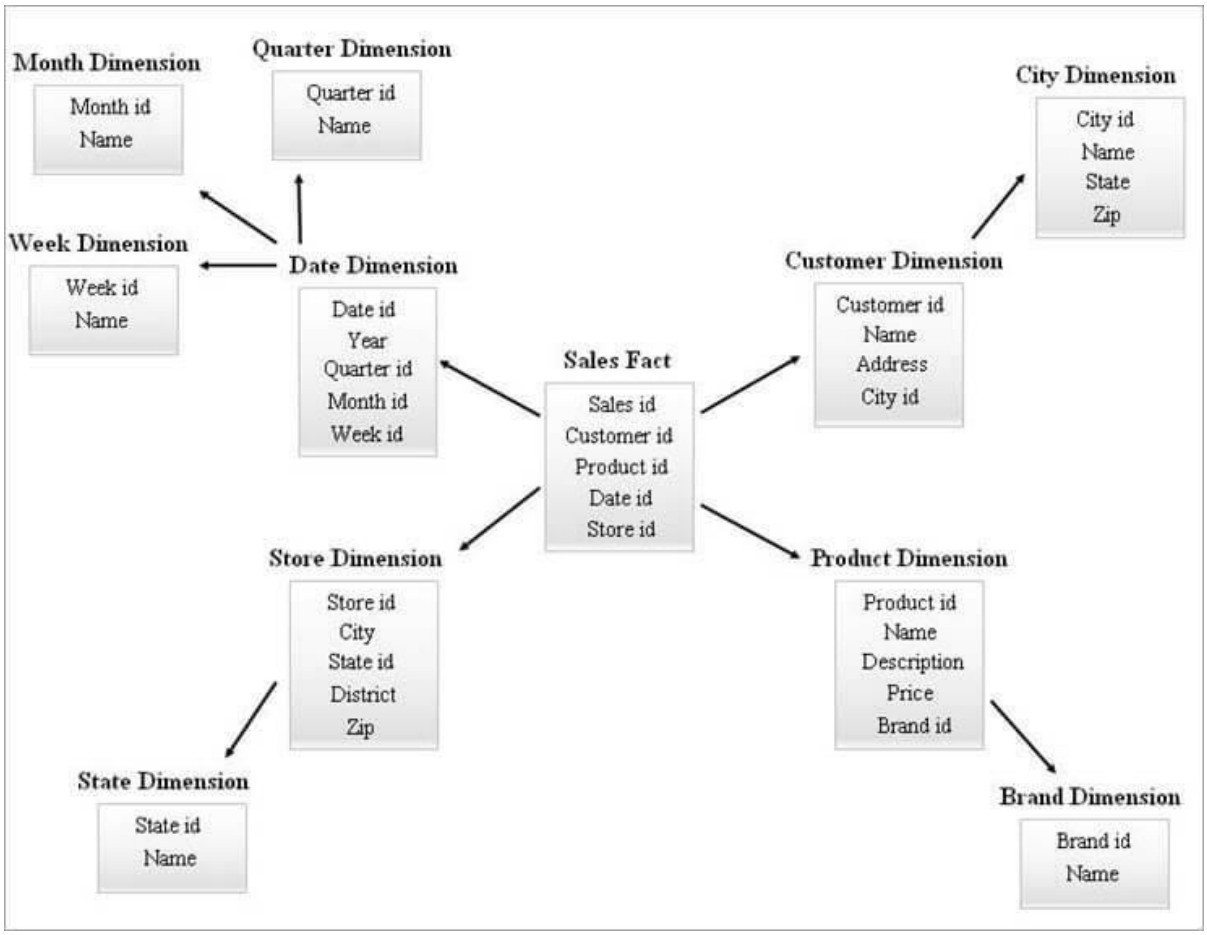
Benefits of SnowFlake Schema:

- Data redundancy is completely removed by creating new dimension tables.
- When compared with star schema, less storage space is used by the Snow Flaking dimension tables.
- It is easy to update (or) maintain the Snow Flaking tables.

Disadvantages of SnowFlake Schema:

- Due to normalized dimension tables, the ETL system has to load the number of tables.
- You may need complex joins to perform a query due to the number of tables added. Hence query performance will be degraded.

<u>An example of a SnowFlake Schema is given below.</u>



The Dimension Tables in the above SnowFlake Diagram are normalized as explained below:

- Date dimension is normalized into Quarterly, Monthly and Weekly tables by leaving foreign key ids in the Date table.
- The store dimension is normalized to comprise the table for State.
- The product dimension is normalized into Brand.
- In the Customer dimension, the attributes connected to the city are moved into the new City table by leaving a foreign key id in the Customer table.

In the same way, a single dimension can maintain multiple levels of hierarchy.

Different levels of hierarchies from the above diagram can be referred to as follows:

- Quarterly id, Monthly id, and Weekly ids are the new surrogate keys that are created for Date dimension hierarchies and those have been added as foreign keys in the Date dimension table.
- State id is the new surrogate key created for Store dimension hierarchy and it has been added as the foreign key in the Store dimension table.
- Brand id is the new surrogate key created for the Product dimension hierarchy and it has been added as the foreign key in the Product dimension table.
- City id is the new surrogate key created for Customer dimension hierarchy and it has been added as the foreign key in the Customer dimension table.

Querying A Snowflake Schema

We can generate the same kind of reports for end-users as that of star schema structures with SnowFlake schemas as well. But the queries are a bit complicated here.

From the above SnowFlake schema example, we are going to generate the same query that we have designed during the Star schema query example.

That is if a business user wants to know how many Novels and DVDs have been sold in the state of Kerala in January in 2018, you can apply the query as follows on SnowFlake schema tables.

```
SELECT   pdim.NameProduct_Name,

              Sum(sfact.sales_units) Quanity_Sold

FROM      Sales sfact

INNERJOINProduct pdim ONsfact.product_id = pdim.product_id

INNERJOINStore sdim ONsfact.store_id = sdim.store_id

INNERJOINState stdim ONsdim.state_id = stdim.state_id

INNERJOINDateddim ONsfact.date_id = ddim.date_id

INNERJOINMonthmdim ONddim.month_id = mdim.month_id
```

```
WHEREstdim.state = 'Kerala'
        ANDmdim.month  = 1
        ANDddim.year    = 2018
        ANDpdim.Namein('Novels', 'DVDs')
GROUPBYpdim.Name
```

Results:

Product_Name	**Quantity_Sold**
Novels	12,702
DVDs	32,919

Points To Remember While Querying Star (or) SnowFlake Schema Tables

Any query can be designed with the below structure:

SELECT Clause:

- The attributes specified in the select clause are shown in the query results.
- The Select statement also uses groups to find the aggregated values and hence we must use group by clause in the where condition.

FROM Clause:

- All the essential fact tables and dimension tables have to be chosen as per the context.

WHERE Clause:

- Appropriate dimension attributes are mentioned in the where clause by joining with the fact table attributes. Surrogate keys from the dimension tables are joined with the respective foreign keys from the fact tables to fix the range of data to be queried. Please refer to the above-written star schema query example to understand this. You can also filter data in the from clause itself if in case you are using inner/outer joins there, as written in the SnowFlake schema example.
- Dimension attributes are also mentioned as constraints on data in the where clause.
- By filtering the data with all the above steps, appropriate data is returned for the reports.

As per the business needs, you can add (or) remove the facts, dimensions, attributes, and constraints to a star schema (or) SnowFlake schema query by following the above structure. You can also add sub-queries (or) merge different query results to generate data for any complex reports.

RESULT:

Thus the different types of Data Warehouse Schemas, along with their benefits and disadvantages.

EX.NO: 5 DESIGN DATA WAREHOUSE FOR REAL TIME APPLICATION

AIM:

To write and implement the design of data warehouse for real time application

INTRODUCTION:

A data warehouse is a single data repository where a record from multiple data sources is integrated for online business analytical processing (OLAP). This implies a data warehouse needs to meet the requirements from all the business stages within the entire organization. Thus, data warehouse design is a hugely complex, lengthy, and hence error-prone process. Furthermore, business analytical functions change over time, which results in changes in the requirements for the systems. Therefore, data warehouse and OLAP systems are dynamic, and the design process is continuous.

Data warehouse design takes a method different from view materialization in the industries. It sees data warehouses as database systems with particular needs such as answering management related queries. The target of the design becomes how the record from multiple data sources should be extracted, transformed, and loaded (ETL) to be organized in a database as the data warehouse.

There are two approaches

1. "top-down" approach
2. "bottom-up" approach

Top-down Design Approach

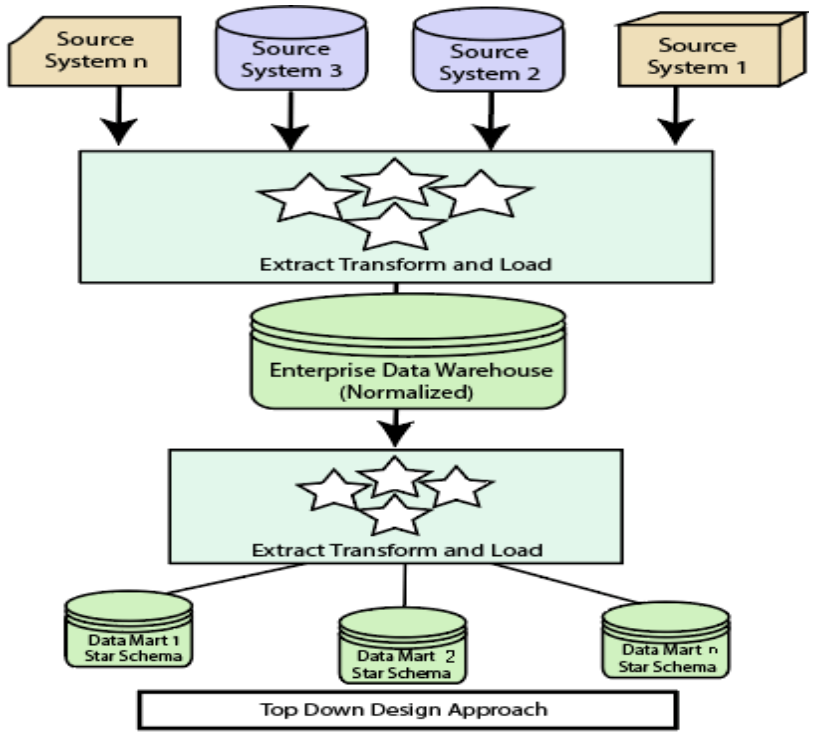
In the "Top-Down" design approach, a data warehouse is described as a subject-oriented, time-variant, non-volatile and integrated data repository for the entire enterprise data from different sources are validated, reformatted and saved in a normalized (up to 3NF) database as the data warehouse. The data warehouse stores "atomic" information, the data at the lowest level of granularity, from where dimensional data marts can be built by selecting the data required for specific business subjects or particular departments. An approach is a data-driven approach as the information is gathered and integrated first and then business requirements by subjects for building data marts are formulated. The advantage of this method is which it supports a single integrated data source. Thus data marts built from it will have consistency when they overlap.

Advantages of top-down design

- Data Marts are loaded from the data warehouses.
- Developing new data mart from the data warehouse is very easy.

Disadvantages of top-down design

- This technique is inflexible to changing departmental needs.
- The cost of implementing the project is high.



Top Down Design Approach

Bottom-Up Design Approach

In the "Bottom-Up" approach, a data warehouse is described as "a copy of transaction data specifical architecture for query and analysis," term the star schema. In this approach, a data mart is created first to necessary reporting and analytical capabilities for particular business processes (or subjects). Thus it is needed to be a business-driven approach in contrast to Inmon's data-driven approach.

Data marts include the lowest grain data and, if needed, aggregated data too. Instead of a normalized database for the data warehouse, a denormalized dimensional database is adapted to meet the data delivery requirements of data warehouses. Using this method, to use the set of data marts as the enterprise data warehouse, data marts should be built with conformed dimensions in mind, defining that ordinary objects are represented the same in different data marts. The conformed dimensions connected the data marts to form a data warehouse, which is generally called a virtual data warehouse.

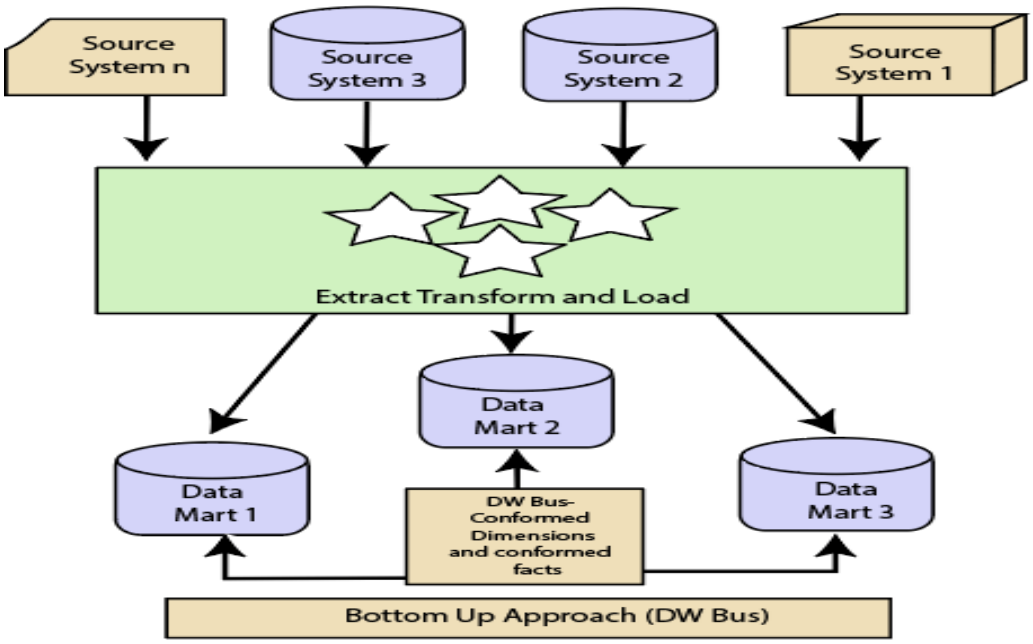
The advantage of the "bottom-up" design approach is that it has quick ROI, as developing a data mart, a data warehouse for a single subject, takes far less time and effort than developing an enterprise-wide data warehouse. Also, the risk of failure is even less. This method is inherently incremental. This method allows the project team to learn and grow.

Advantages of bottom-up design

- Documents can be generated quickly.
- The data warehouse can be extended to accommodate new business units.
- It is just developing new data marts and then integrating with other data marts.

Disadvantages of bottom-up design

the locations of the data warehouse and the data marts are reversed in the bottom-up approach design.



Bottom Up Design Approach

PROGRAM:

@RELATION IRIS

@ATTRIBUTE Species_No NUMBER

@ATTRIBUTE Petal_width NUMBER

@ATTRIBUTE Petal_length NUMBER

@ATTRIBUTE Sepal_width NUMBER

@ATTRIBUTE Sepal_length NUMBER

@ATTRIBUTE Species_name { Setosa, Versicolor, Verginica}

@DATA

1,0.2,1.4,3.5,5.1, Setosa

1,0.2,1.4,3,4.9, Setosa

1,0.2,1.3,3.2,4.7, Setosa

1,0.2,1.5,3.1,4.6, Setosa

1,0.2,1.4,3.6,5, Setosa

1,0.4,1.7,3.9,5.4, Setosa

1,0.3,1.4,3.4,4.6, Setosa

1,0.2,1.5,3.4,5, Setosa

1,0.2,1.4,2.9,4.4, Setosa

1,0.1,1.5,3.1,4.9, Setosa

1,0.2,1.5,3.7,5.4, Setosa

1,0.2,1.6,3.4,4.8, Setosa

1,0.1,1.4,3,4.8, Setosa

1,0.1,1.1,3,4.3, Setosa

1,0.2,1.2,4,5.8, Setosa

1,0.4,1.5,4.4,5.7, Setosa

1,0.4,1.3,3.9,5.4, Setosa

1,0.3,1.4,3.5,5.1, Setosa

1,0.3,1.7,3.8,5.7, Setosa

1,0.3,1.5,3.8,5.1, Setosa

1,0.2,1.7,3.4,5.4, Setosa

1,0.4,1.5,3.7,5.1, Setosa

1,0.2,1,3.6,4.6, Setosa

1,0.5,1.7,3.3,5.1, Setosa

1,0.2,1.9,3.4,4.8, Setosa

1,0.2,1.6,3,5, Setosa

1,0.4,1.6,3.4,5, Setosa

1,0.2,1.5,3.5,5.2, Setosa

1,0.2,1.4,3.4,5.2, Setosa
1,0.2,1.6,3.2,4.7, Setosa
1,0.2,1.6,3.1,4.8, Setosa
1,0.4,1.5,3.4,5.4, Setosa
1,0.1,1.5,4.1,5.2, Setosa
1,0.2,1.4,4.2,5.5, Setosa
1,0.2,1.5,3.1,4.9, Setosa
1,0.2,1.2,3.2,5, Setosa
1,0.2,1.3,3.5,5.5, Setosa
1,0.1,1.4,3.6,4.9, Setosa
1,0.2,1.3,3,4.4, Setosa
1,0.2,1.5,3.4,5.1, Setosa
1,0.3,1.3,3.5,5, Setosa
1,0.3,1.3,2.3,4.5, Setosa
1,0.2,1.3,3.2,4.4, Setosa
1,0.6,1.6,3.5,5, Setosa
1,0.4,1.9,3.8,5.1, Setosa
1,0.3,1.4,3,4.8, Setosa
1,0.2,1.6,3.8,5.1, Setosa
1,0.2,1.4,3.2,4.6, Setosa
1,0.2,1.5,3.7,5.3, Setosa
1,0.2,1.4,3.3,5, Setosa
2,1.4,4.7,3.2,7, Versicolor
2,1.5,4.5,3.2,6.4, Versicolor
2,1.5,4.9,3.1,6.9, Versicolor
2,1.3,4,2.3,5.5, Versicolor
2,1.5,4.6,2.8,6.5, Versicolor
2,1.3,4.5,2.8,5.7, Versicolor
2,1.6,4.7,3.3,6.3, Versicolor
2,1,3.3,2.4,4.9, Versicolor
2,1.3,4.6,2.9,6.6, Versicolor
2,1.4,3.9,2.7,5.2, Versicolor
2,1,3.5,2,5, Versicolor
2,1.5,4.2,3,5.9, Versicolor

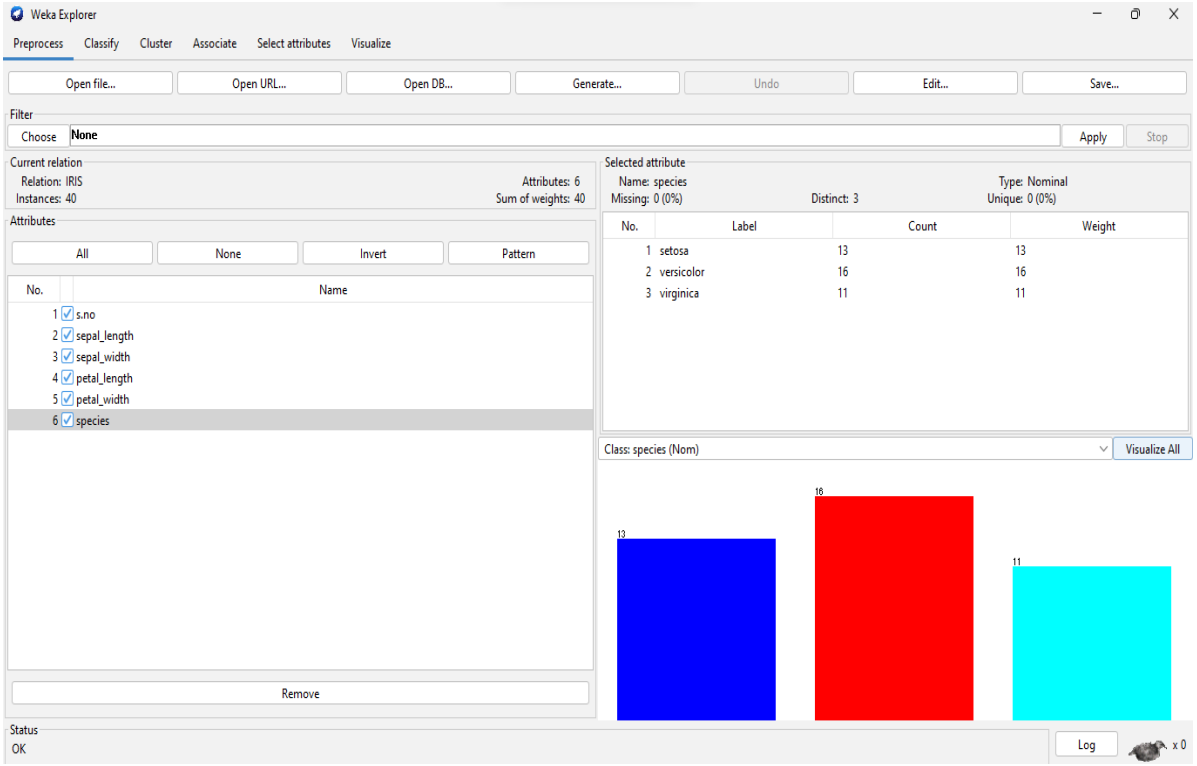
2,1,4,2.2,6, Versicolor
2,1.4,4.7,2.9,6.1, Versicolor
2,1.3,3.6,2.9,5.6, Versicolor
2,1.4,4.4,3.1,6.7, Versicolor
2,1.5,4.5,3,5.6, Versicolor
2,1,4.1,2.7,5.8, Versicolor
2,1.5,4.5,2.2,6.2, Versicolor
2,1.1,3.9,2.5,5.6, Versicolor
2,1.8,4.8,3.2,5.9, Versicolor
2,1.3,4,2.8,6.1, Versicolor
2,1.5,4.9,2.5,6.3, Versicolor
2,1.2,4.7,2.8,6.1, Versicolor
2,1.3,4.3,2.9,6.4, Versicolor
2,1.4,4.4,3,6.6, Versicolor
2,1.4,4.8,2.8,6.8, Versicolor
2,1.7,5,3,6.7, Versicolor
2,1.5,4.5,2.9,6, Versicolor
2,1,3.5,2.6,5.7, Versicolor
2,1.1,3.8,2.4,5.5, Versicolor
2,1,3.7,2.4,5.5, Versicolor
2,1.2,3.9,2.7,5.8, Versicolor
2,1.6,5.1,2.7,6, Versicolor
2,1.5,4.5,3,5.4, Versicolor
2,1.6,4.5,3.4,6, Versicolor
2,1.5,4.7,3.1,6.7, Versicolor
2,1.3,4.4,2.3,6.3, Versicolor
2,1.3,4.1,3,5.6, Versicolor
2,1.3,4,2.5,5.5, Versicolor
2,1.2,4.4,2.6,5.5, Versicolor
2,1.4,4.6,3,6.1, Versicolor
2,1.2,4,2.6,5.8, Versicolor
2,1,3.3,2.3,5, Versicolor
2,1.3,4.2,2.7,5.6, Versicolor
2,1.2,4.2,3,5.7, Versicolor

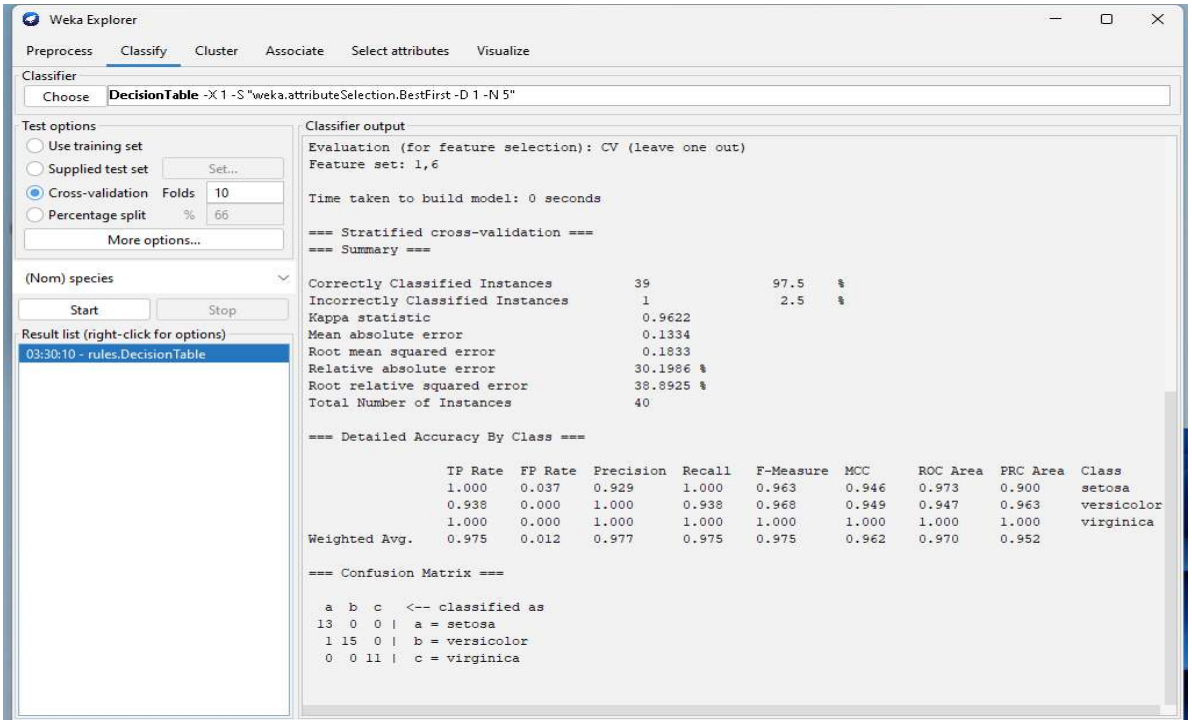
2,1.3,4.2,2.9,5.7, Versicolor
2,1.3,4.3,2.9,6.2, Versicolor
2,1.1,3,2.5,5.1, Versicolor
2,1.3,4.1,2.8,5.7, Versicolor
3,2.5,6,3.3,6.3, Verginica
3,1.9,5.1,2.7,5.8, Verginica
3,2.1,5.9,3,7.1, Verginica
3,1.8,5.6,2.9,6.3, Verginica
3,2.2,5.8,3,6.5, Verginica
3,2.1,6.6,3,7.6, Verginica
3,1.7,4.5,2.5,4.9, Verginica
3,1.8,6.3,2.9,7.3, Verginica
3,1.8,5.8,2.5,6.7, Verginica
3,2.5,6.1,3.6,7.2, Verginica
3,2,5.1,3.2,6.5, Verginica
3,1.9,5.3,2.7,6.4, Verginica
3,2.1,5.5,3,6.8, Verginica
3,2,5,2.5,5.7, Verginica
3,2.4,5.1,2.8,5.8, Verginica
3,2.3,5.3,3.2,6.4, Verginica
3,1.8,5.5,3,6.5, Verginica
3,2.2,6.7,3.8,7.7, Verginica
3,2.3,6.9,2.6,7.7, Verginica
3,1.5,5,2.2,6, Verginica
3,2.3,5.7,3.2,6.9, Verginica
3,2,4.9,2.8,5.6, Verginica
3,2,6.7,2.8,7.7, Verginica
3,1.8,4.9,2.7,6.3, Verginica
3,2.1,5.7,3.3,6.7, Verginica
3,1.8,6,3.2,7.2, Verginica
3,1.8,4.8,2.8,6.2, Verginica
3,1.8,4.9,3,6.1, Verginica
3,2.1,5.6,2.8,6.4, Verginica
3,1.6,5.8,3,7.2, Verginica

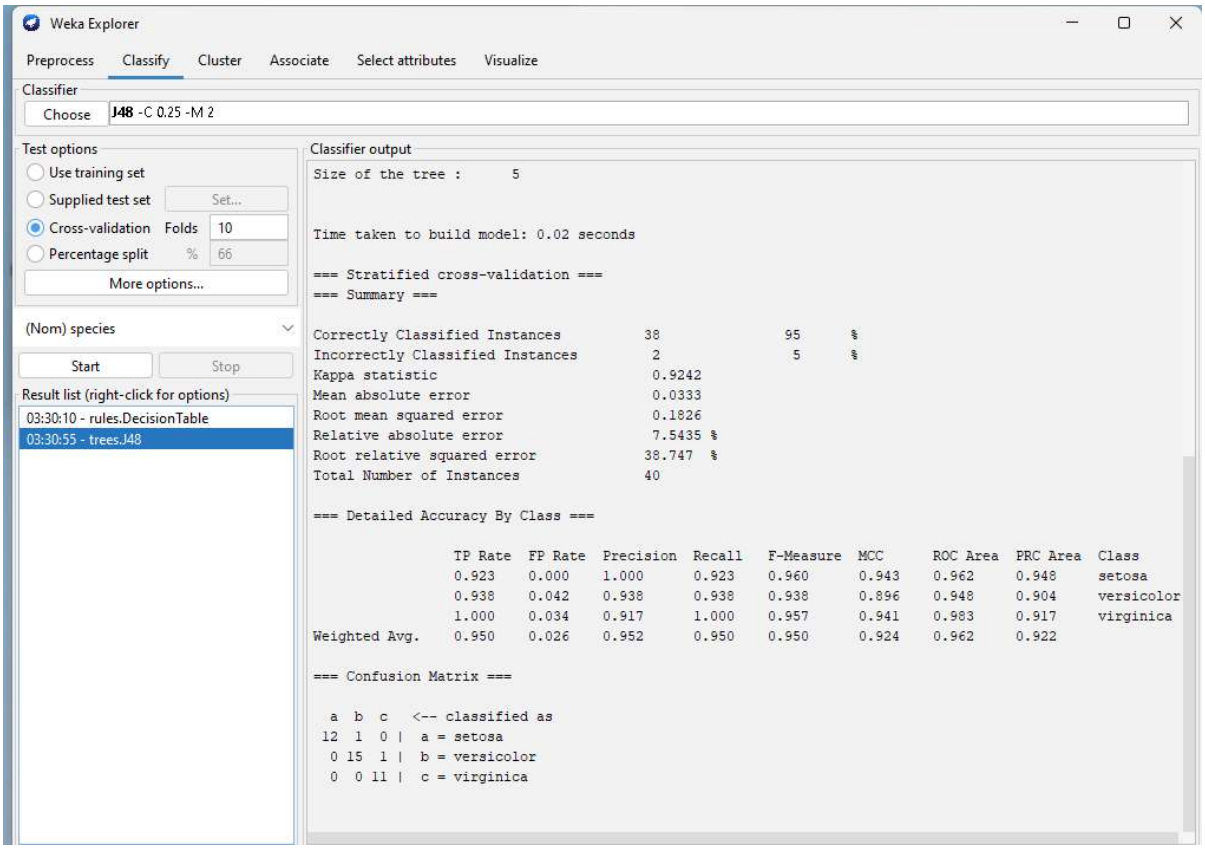
3,1.9,6.1,2.8,7.4, Verginica
3,2,6.4,3.8,7.9, Verginica
3,2.2,5.6,2.8,6.4, Verginica
3,1.5,5.1,2.8,6.3, Verginica
3,1.4,5.6,2.6,6.1, Verginica
3,2.3,6.1,3,7.7, Verginica
3,2.4,5.6,3.4,6.3, Verginica
3,1.8,5.5,3.1,6.4, Verginica
3,1.8,4.8,3,6, Verginica
3,2.1,5.4,3.1,6.9, Verginica
3,2.4,5.6,3.1,6.7, Verginica
3,2.3,5.1,3.1,6.9, Verginica
3,1.9,5.1,2.7,5.8, Verginica
3,2.3,5.9,3.2,6.8, Verginica
3,2.5,5.7,3.3,6.7, Verginica
3,2.3,5.2,3,6.7, Verginica
3,1.9,5,2.5,6.3, Verginica
3,2,5.2,3,6.5, Verginica
3,2.3,5.4,3.4,6.2, Verginica
3,1.8,5.1,3,5.9, Verginica

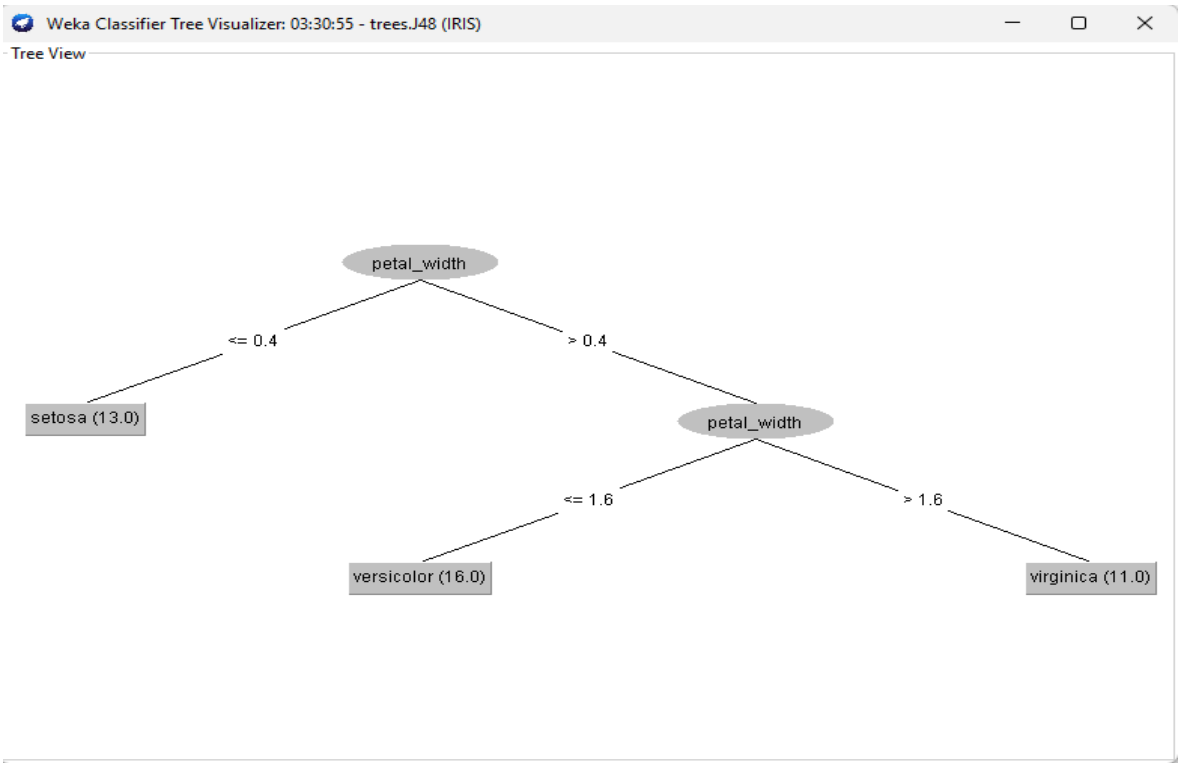
s.no	sepal_len	sepal_wi	petal_len	petal_wic	species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3	1.4	0.1	setosa
14	7	3.2	4.7	1.4	versicolor
15	6.4	3.2	4.5	1.5	versicolor
16	6.9	3.1	4.9	1.5	versicolor
17	5.5	2.3	4	1.3	versicolor
18	6.5	2.8	4.6	1.5	versicolor
19	5.7	2.8	4.5	1.3	versicolor
20	6.3	3.3	4.7	1.6	versicolor
21	4.9	2.4	3.3	1	versicolor
22	6.6	2.9	4.6	1.3	versicolor
23	5.2	2.7	3.9	1.4	versicolor
24	5	2	3.5	1	versicolor
25	5.9	3	4.2	1.5	versicolor
26	6	2.2	4	1	versicolor
27	6.3	3.3	6	2.5	virginica
28	5.8	5.8	2.7	1.9	virginica
29	7.1	3	5.9	2.1	virginica
30	6.3	2.9	5.6	1.8	virginica
31	6.5	3	5.8	2.2	virginica
32	7.6	3	6.6	2.1	virginica
33	4.9	2.5	4.5	1.7	virginica
34	7.3	2.9	6.3	1.8	virginica
35	6.7	2.5	5.8	1.8	virginica
36	7.2	3.6	6.1	2.5	virginica
37	6.5	3.2	5.1	2	virginica
38	6.2	2.9	4.3	1.3	versicolor
39	5.1	2.5	3	1.1	versicolor
40	5.7	2.8	4.1	1.3	versicolor

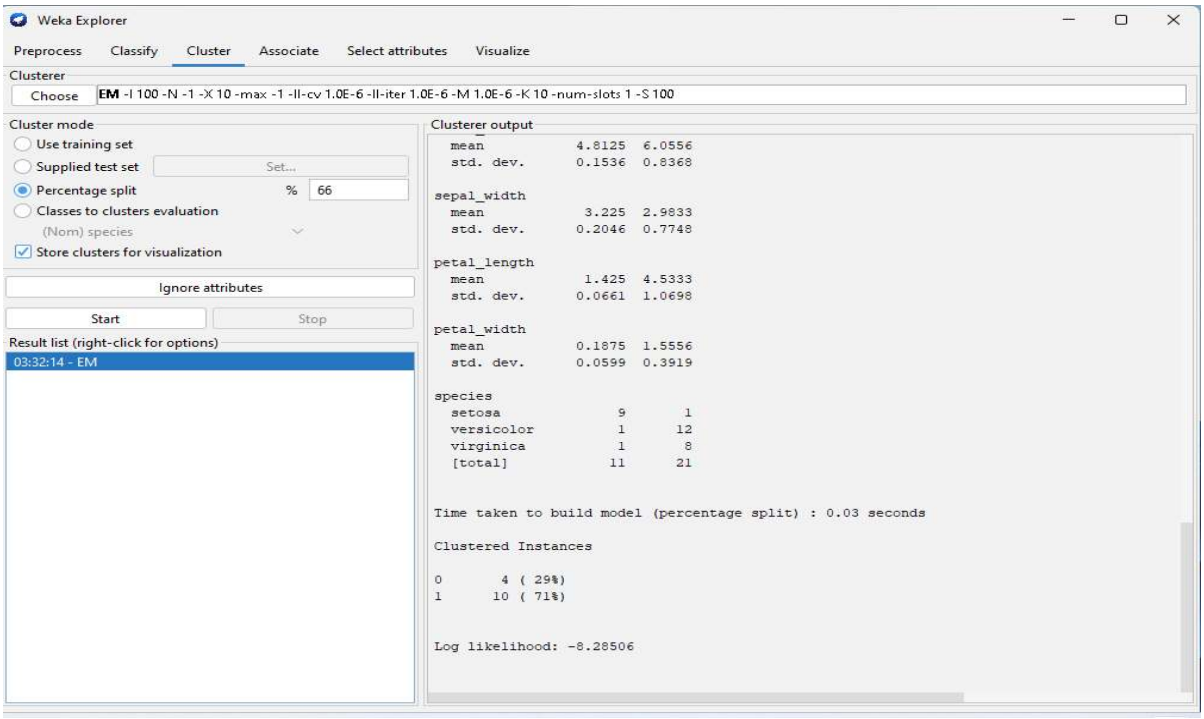
OUTPUT:











RESULT:

Thus the design of data warehouse real time application using datasets.

EX.NO: 6 CASE STUDY:ANALYSE THE DIMENSINAL MODELING

AIM:

To write and analyse the dimensional modelling for data warehouse.

PROCEDURE:

Dimensional modeling provides set of methods and concepts that are used in DW design. According to DW consultant, Ralph Kimball, dimensional modeling is a design technique for databases intended to support end-user queries in a data warehouse. It is oriented around understandability and performance. According to him, although transaction-oriented ER is very useful for the transaction capture, it should be avoided for end-user delivery.

Dimensional modeling always uses facts and dimension tables. Facts are numerical values which can be aggregated and analyzed on the fact values. Dimensions define hierarchies and description on fact values.

Dimension Table

Dimension table stores the attributes that describe objects in a Fact table. A Dimension table has a primary key that uniquely identifies each dimension row. This key is used to associate the Dimension table to a Fact table.

Dimension tables are normally de-normalized as they are not created to execute transactions and only used to analyze data in detail.

Example

In the following dimension table, the customer dimension normally includes the name of customers, address, customer id, gender, income group, education levels, etc.

Customer ID	Name	Gender	Income	Education	Religion
1	Brian Edge	M	2	3	4
2	Fred Smith	M	3	5	1
3	Sally Jones	F	1	7	3

Fact Tables

Fact table contains numeric values that are known as measurements. A Fact table has two types of columns − facts and foreign key to dimension tables.

Measures in Fact table are of three types −

- **Additive** − Measures that can be added across any dimension.

- **Non-Additive** − Measures that cannot be added across any dimension.
- **Semi-Additive** − Measures that can be added across some dimensions.

Example

Time ID	Product ID	Customer ID	Unit Sold
4	17	2	1
8	21	3	2
8	4	1	1

This fact tables contains foreign keys for time dimension, product dimension, customer dimension and measurement value unit sold.

Suppose a company sells products to customers. Every sale is a fact that happens within the company, and the fact table is used to record these facts.

Common facts are − number of unit sold, margin, sales revenue, etc. The dimension table list factors like customer, time, product, etc. by which we want to analyze the data.

Now if we consider the above Fact table and Customer dimension then there will also be a Product and time dimension. Given this fact table and these three dimension tables, we can ask questions like: How many watches were sold to male customers in 2010?

Difference between Dimension and Fact Table

The functional difference between dimension tables and fact tables is that fact tables hold the data we want to analyze and dimension tables hold the information required to allow us to query it.

Aggregate Table

Aggregate table contains aggregated data which can be calculated by using different aggregate functions.

An **aggregate function** is a function where the values of multiple rows are grouped together as input on certain criteria to form a single value of more significant meaning or measurement.

Common aggregate functions include −

- Average()
- Count()
- Maximum()
- Median()
- Minimum()
- Mode()

- Sum()

These aggregate tables are used for performance optimization to run complex queries in a data warehouse.

Example

You save tables with aggregated data like yearly (1 row), quarterly (4 rows), monthly (12 rows) and now you have to do comparison of data, like Yearly only 1 row will be processed. However in an un-aggregated table, all the rows will be processed.

MIN	Returns the smallest value in a given column
MAX	Returns the largest value in a given column
SUM	Returns the sum of the numeric values in a given column
AVG	Returns the average value of a given column
COUNT	Returns the total number of values in a given column
COUNT (*)	Returns the number of rows in a table

Select Avg (salary) from employee where title = ‘developer’. This statement will return the average salary for all employees whose title is equal to 'Developer'.

Aggregations can be applied at database level. You can create aggregates and save them in aggregate tables in the database or you can apply aggregate on the fly at the report level.

RESULT:

Thus the dimensional modelling for data warehouse analyse successfully.

EX.NO: 7 CASE STUDY USING: OLAP

AIM:

To write and analyse the OLAP operation.

INTODUCTION:

Online Analytical Processing Server (OLAP) is based on the multidimensional data model. It allows managers, and analysts to get an insight of the information through fast, consistent, and interactive access to information. This chapter cover the types of OLAP, operations on OLAP, difference between OLAP, and statistical databases and OLTP.

Types of OLAP Servers

We have four types of OLAP servers −

- Relational OLAP (ROLAP)
- Multidimensional OLAP (MOLAP)
- Hybrid OLAP (HOLAP)
- Specialized SQL Servers

Relational OLAP

ROLAP servers are placed between relational back-end server and client front-end tools. To store and manage warehouse data, ROLAP uses relational or extended-relational DBMS.

ROLAP includes the following −

- Implementation of aggregation navigation logic.
- Optimization for each DBMS back end.
- Additional tools and services.

Multidimensional OLAP

MOLAP uses array-based multidimensional storage engines for multidimensional views of data. With multidimensional data stores, the storage utilization may be low if the data set is sparse. Therefore, many MOLAP server use two levels of data storage representation to handle dense and sparse data sets.

Hybrid OLAP

Hybrid OLAP is a combination of both ROLAP and MOLAP. It offers higher scalability of ROLAP and faster computation of MOLAP. HOLAP servers allows to store the large data volumes of detailed information. The aggregations are stored separately in MOLAP store.

INNERJOINMonthmdim ONddim.month_id = mdim.month_id

WHEREstdim.state = 'Kerala'

ANDmdim.month = 1

ANDddim.year = 2018

ANDpdim.Namein(‘Novels’, ‘DVDs’)

GROUPBYpdim.Name

Specialized SQL servers provide advanced query language and query processing support for SQL queries over star and snowflake schemas in a read-only environment.

OLAP Operations

Since OLAP servers are based on multidimensional view of data, we will discuss OLAP operations in multidimensional data.

Here is the list of OLAP operations −

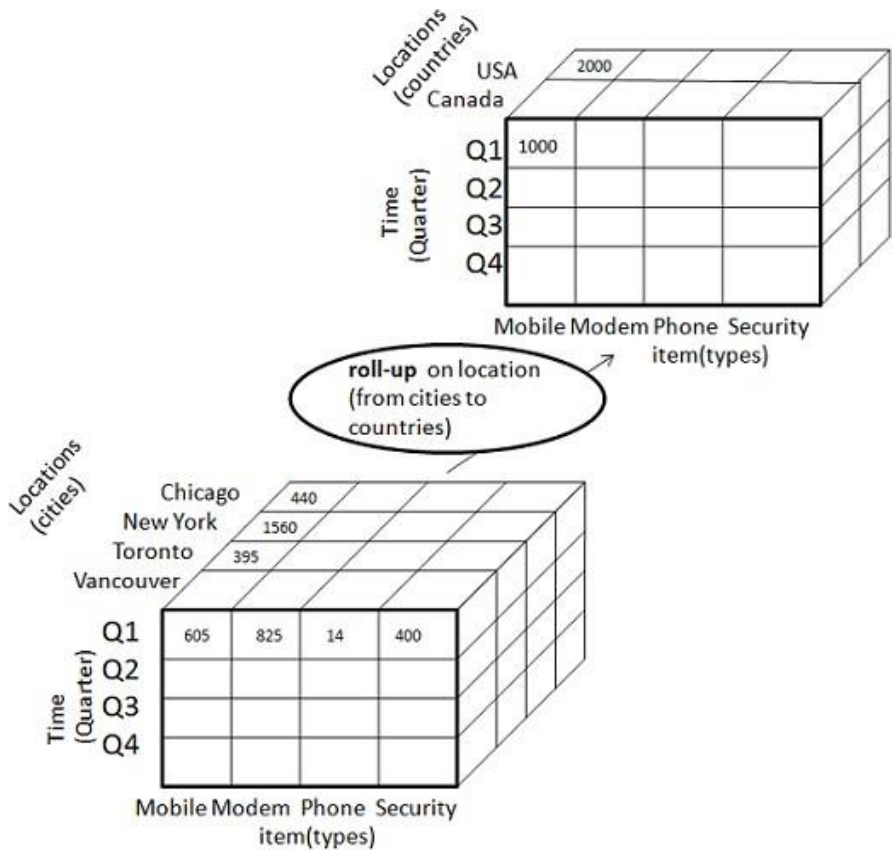
- Roll-up
- Drill-down
- Slice and dice
- Pivot (rotate)

Roll-up

Roll-up performs aggregation on a data cube in any of the following ways −

- By climbing up a concept hierarchy for a dimension
- By dimension reduction

The following diagram illustrates how roll-up works.



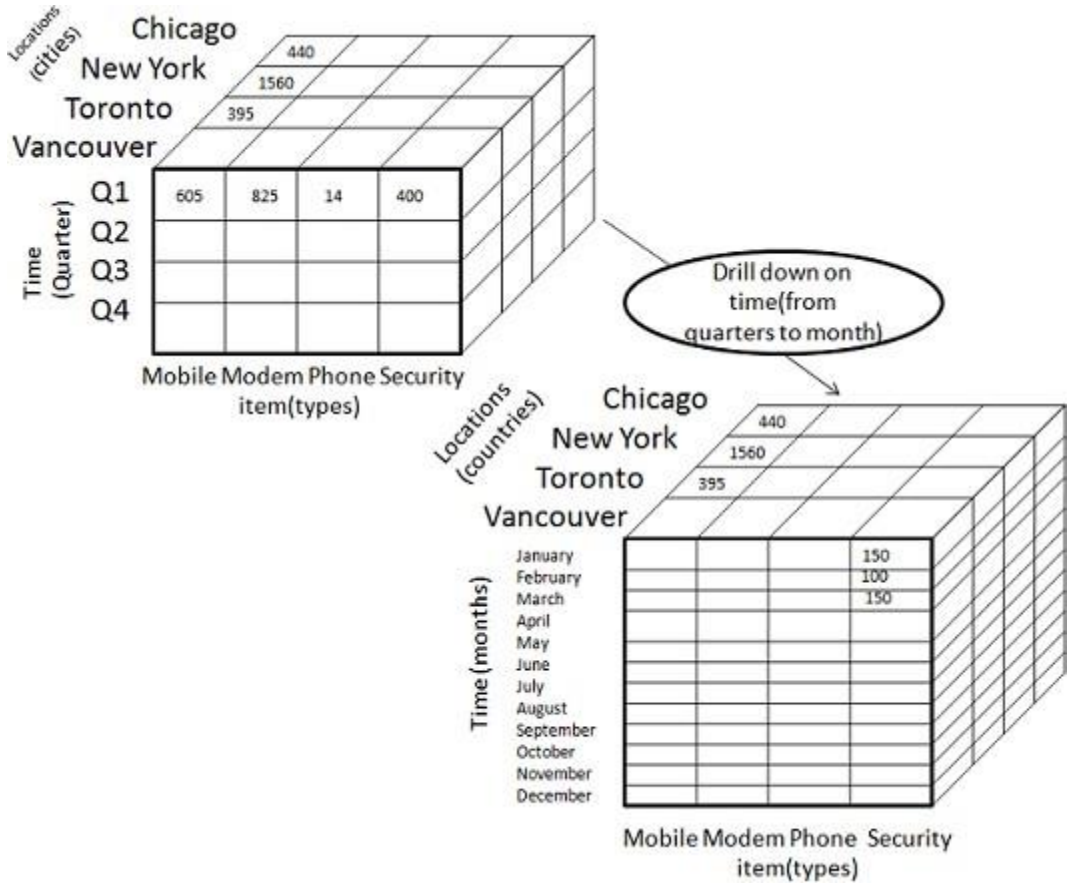
- Roll-up is performed by climbing up a concept hierarchy for the dimension location.
- Initially the concept hierarchy was "street < city < province < country".
- On rolling up, the data is aggregated by ascending the location hierarchy from the level of city to the level of country.
- The data is grouped into cities rather than countries.
- When roll-up is performed, one or more dimensions from the data cube are removed.

Drill-down

Drill-down is the reverse operation of roll-up. It is performed by either of the following ways −

- By stepping down a concept hierarchy for a dimension
- By introducing a new dimension.

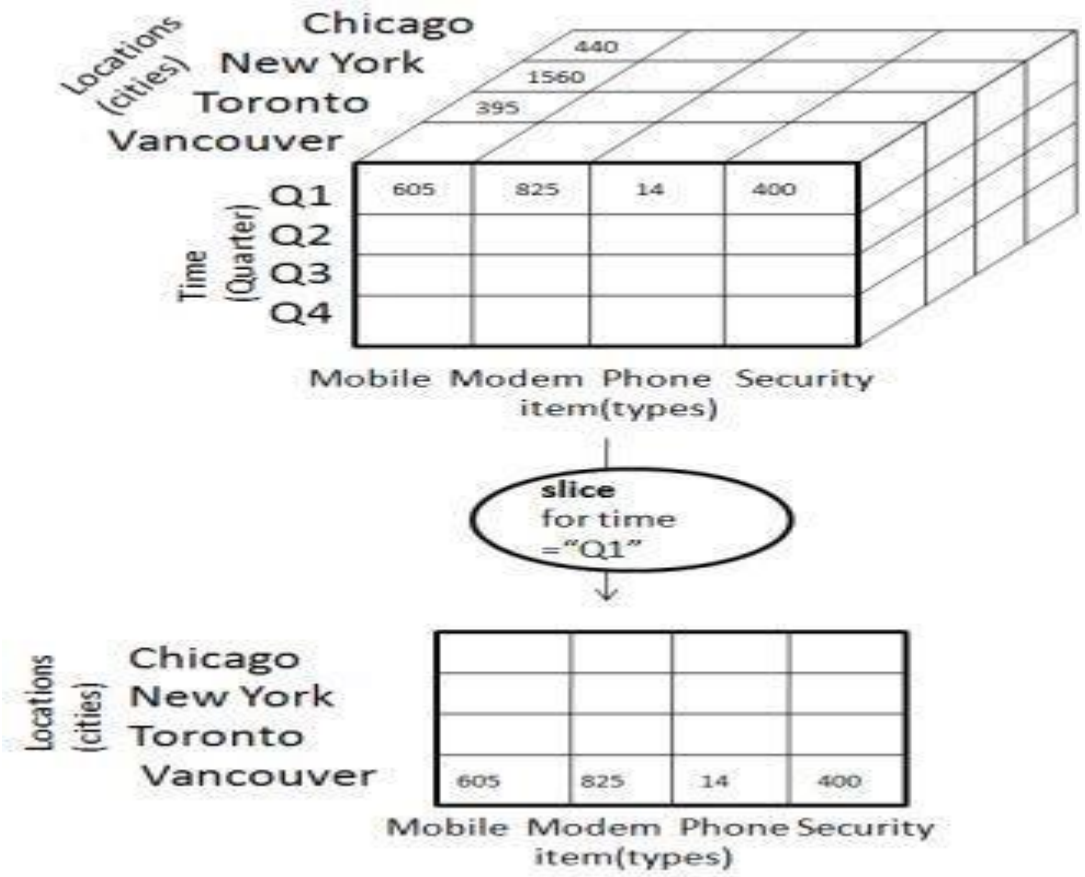
The following diagram illustrates how drill-down works −



- Drill-down is performed by stepping down a concept hierarchy for the dimension time.
- Initially the concept hierarchy was "day < month < quarter < year."
- On drilling down, the time dimension is descended from the level of quarter to the level of month.
- When drill-down is performed, one or more dimensions from the data cube are added.
- It navigates the data from less detailed data to highly detailed data.

Slice

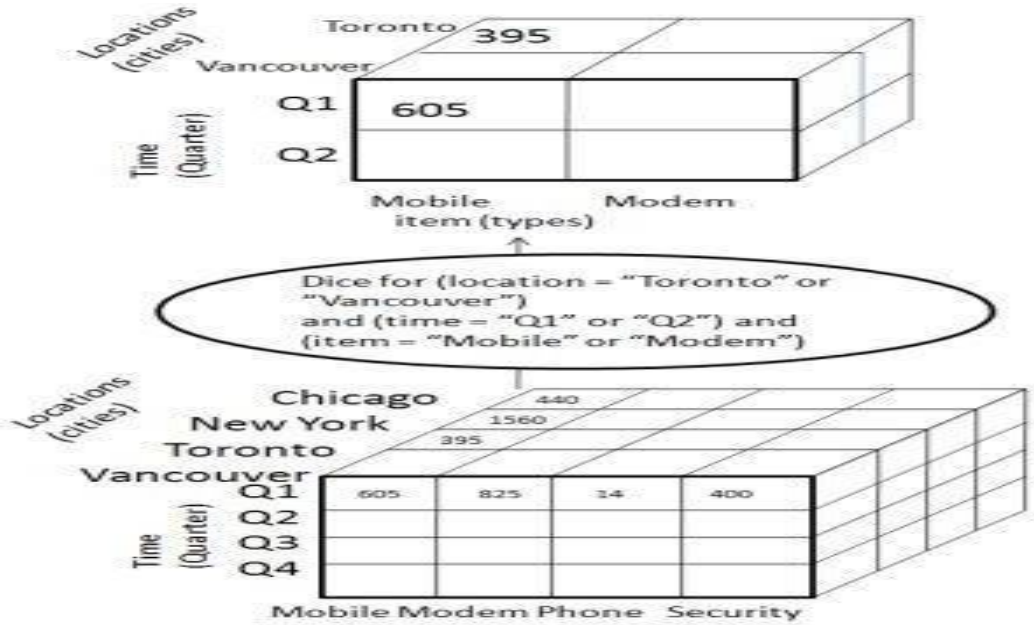
The slice operation selects one particular dimension from a given cube and provides a new sub-cube. Consider the following diagram that shows how slice works.



- Here Slice is performed for the dimension "time" using the criterion time = "Q1".
- It will form a new sub-cube by selecting one or more dimensions.

Dice

Dice selects two or more dimensions from a given cube and provides a new sub-cube. Consider the following diagram that shows the dice operation.

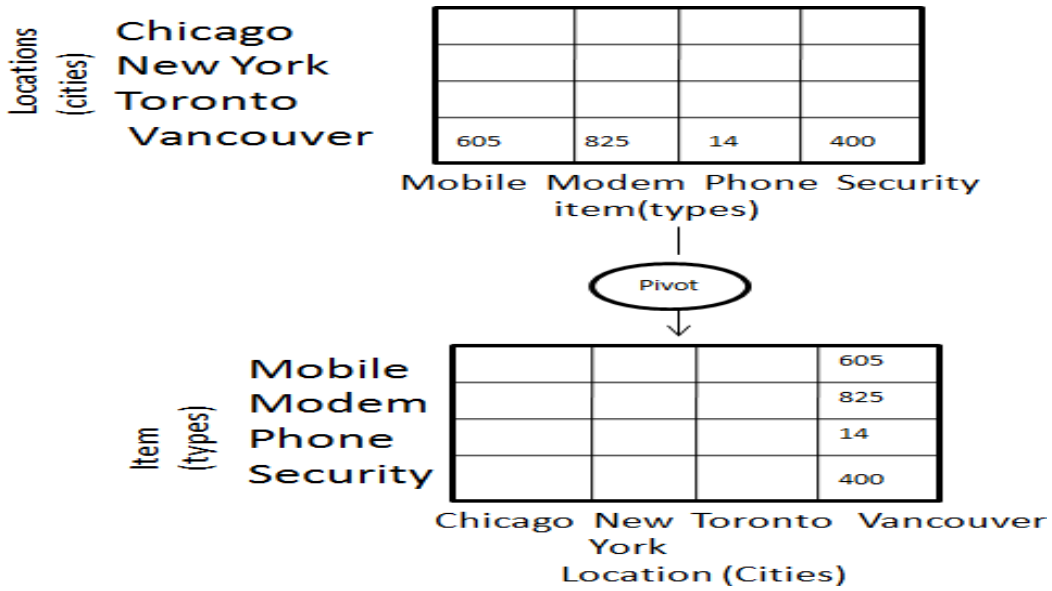


The dice operation on the cube based on the following selection criteria involves three dimensions.

- (location = "Toronto" or "Vancouver")
- (time = "Q1" or "Q2")
- (item =" Mobile" or "Modem")

Pivot

The pivot operation is also known as rotation. It rotates the data axes in view in order to provide an alternative presentation of data. Consider the following diagram that shows the pivot operation.



RESULT:

Thus the OLAP operation implement and analyse the successfully.

EX.NO: 8 CASE STUDY USING OTLP

AIM:

To write and analysis the On-Line Transaction Processing.

On-Line Transaction Processing (OLTP) System

It refers to the system that manage transaction oriented applications. These systems are designed to support on-line transaction and process query quickly on the Internet.

For example: POS (point of sale) system of any supermarket is a OLTP System.

Every industry in today's world use OLTP system to record their transactional data. The main concern of OLTP systems is to enter, store and retrieve the data. They covers all day to day operations such as purchasing, manufacturing, payroll, accounting, etc.of an organization. Such systems have large numbers of user which conduct short transaction. It supports simple database query so the response time of any user action is very fast.

The data acquired through an OLTP system is stored in commercial RDBMS, which can be used by an OLAP System for data analytics and other business intelligence operations.

Some other examples of OLTP systems include order entry, retail sales, and financial transaction systems

Advantages of an OLTP System:

- OLTP Systems are user friendly and can be used by anyone having basic understanding
- It allows its user to perform operations like read, write and delete data quickly.
- It responds to its user actions immediately as it can process query very quickly.
- This systems are original source of the data.
- It helps to administrate and run fundamental business tasks
- It helps in widening customer base of an organization by simplifying individual processes

Challenges of an OLTP system:

- It allows multiple users to access and change the same data at the same time. So it requires concurrency control and recovery mechanism to avoid any unprecedented situations

- The data acquired through OLTP systems are not suitable for decision making. OLAP systems are used for the decision making or "what if" analysis.

Type of queries that an OLTP system can Process:
An OLTP system is an online database modifying system. So it supports database query like INSERT, UPDATE and DELETE information from the database. Consider a POS system of a supermarket, Below are the sample queries that it can process –

- Retrieve the complete description of a particular product
- Filter all products related to any particular supplier
- Search for the record of any particular customer.
- List all products having price less than Rs 1000.

Type of queries that an OLTP system can not Process:
An OLTP system supports simple database query like INSERT, UPDATE and DELETE only. It does not support complex query. Reconsider the POS system of the supermarket, Below are the sample queries that it can not process –

- How much discount should they offer on a particular product?
- Which product should be introduced to its customer ?

The key <u>differences between OLTP and OLAP</u> databases:

OLTP	**OLAP**
OLTP is characterized by a large number of short on-line transactions (INSERT, UPDATE, DELETE).	OLAP is characterized by relatively low volume of transactions.
OLTP queries are simple and easy to understand.	OLAP Queries are often very complex and involve aggregations.
OLTP is widely used for small transaction.	OLAP applications are widely used by Data Mining techniques.
OLTP is highly normalized.	OLAP is typically de-normalized.
OLTP is used for Backup religiously.	OLAP is used for regular backup.

OLTP	OLAP
OLTP usually uses schema used to store transnational databases is the entity model (usually 3NF).	OLAP uses star model to store the data.
Performance of OLTP is comparably fast as compared to OLAP.	Performance of OLAP is comparably low as compared to OLTP.

RESULT:

Thus the OLT Processing implement and analysis successfully.

EX.NO:9 IMPLEMENTATION OF WAREHOUSE TESTING

AIM:

The Goals of Data Warehouse Testing, ETL Testing Responsibilities, Errors in DW and ETL Deployment in detail in this tutorial.

Data Warehouse (ETL) Testing

What is the significance of testing Data Warehouse and Business Intelligence systems?

Testing plays a critical role in the success of any of the above two systems, by ensuring the correctness of data that builds the faith of end-users.

In general, a defect found at the later stages of the software development life cycle costs more to fix that defect. This situation in the DW can be worsened because the wrong data found at the later stages might have been used in important business decisions by that time.

Thus, the fix in the DW is more expensive in terms of process, people and technology changes. You can begin the DW testing right from the requirements gathering phase.

A requirement traceability matrix is prepared & reviewed, and this mainly maps the DW features with their respective business requirements. The traceability matrix acts as an input to the DW test plan that is prepared by the testers. The test plan describes the tests to be performed to validate the DW system.

It also describes the types of tests that will be performed on the system. After the test plan is ready all the detailed test cases will be prepared for various DW scenarios. Then all the test cases will be executed and defects will be logged.

There is a standard in the operational world that maintains different environments for development, testing, and production. In the DW world, both the developers and testers will make sure that the development and test environments are available with the replica of production data before starting their work.

This is copied for a list of tables with limited or full data depending on the project needs, as the production data is really large. The developers develop their code in the developer's environment and deliver it to the testers.

The testers will test the code delivered in the testing environments to ensure if all the systems are working. Then the code will go live in the production environments. The DW code is also maintained in different versions based on the defects fixed in each release. Maintaining multiple environments and code versions helps to build a good quality system.

Goals Of Data Warehouse (ETL) Testing

Let's take a look at the Goals Of Data Warehouse Testing.

#1) Data Completeness: Ensure that all data from various sources is loaded into a Data Warehouse. The testing team validates if all the DW records are loaded, against the source database and flat files by following the below sample strategies.

- The total number of records count uploaded from the source system should match the total number of records loaded into DW. If there is a difference then you can think about the rejected records.
- Compare the data loaded into each field of DW with the source system data fields. This will bring out the data errors if any.

#2) Data Transformation: While uploading the source data to the Data warehouse, few fields can be directly loaded with the source data but few fields will be loaded with the data that is transformed as per the business logic. This is the complex portion of testing DW (ETL).

Below are the sample strategies to test this:

- You can test by creating and comparing data in spreadsheets. Load the source transformed data and DW data into spreadsheets and do a comparison. There should not be any mismatch.
- Testers should write the queries as per the transformation logic to compare the DW data with the source data. Query execution will guarantee that the data validation for any of the fields is not missing.
-

#3) Data Quality: Data warehouse (ETL) system must ensure the quality of the data loaded into it by rejecting (or) correcting the data.

DW may **reject** a few of the source system data based on the business requirements logic. **For Example,** reject a record if a certain field has non-numeric data. All the rejected records are loaded into the reject table for reference.

The rejected data is reported to the clients because there is no chance of getting to know about this missed data, as it will not be loaded into the DW system. DW may **correct** the data by loading zero in the place of null values etc.

#4) Scalability and Performance: Data warehouse must ensure the scalability of the system with increasing loads. With this, there should not be any degradation in the performance while executing the queries, with anticipated results in specific time frames. Thus performance testing uncovers any issues and fixes it before the production.

Below are sample strategies for Performance and Scalability Testing:

- Do the performance testing by loading production volumes of data and ensure that the time frames are not missed.
- Validate the performance of each query with bulk data. Test the performance by using simple joins and multiple joins.
- Load double (or) triple to the volumes of data expected to calculate the capacity of the system approximately.
- Test by running jobs for all the listed reports at the same time.

#5) Integration Testing: Data warehouse should perform Integration Testing with other upstream and downstream applications. If possible, it is better to copy the production data into the test environment for Integration Testing.

All system teams should be involved in this phase to bridge the gaps while understanding and testing all the systems together.

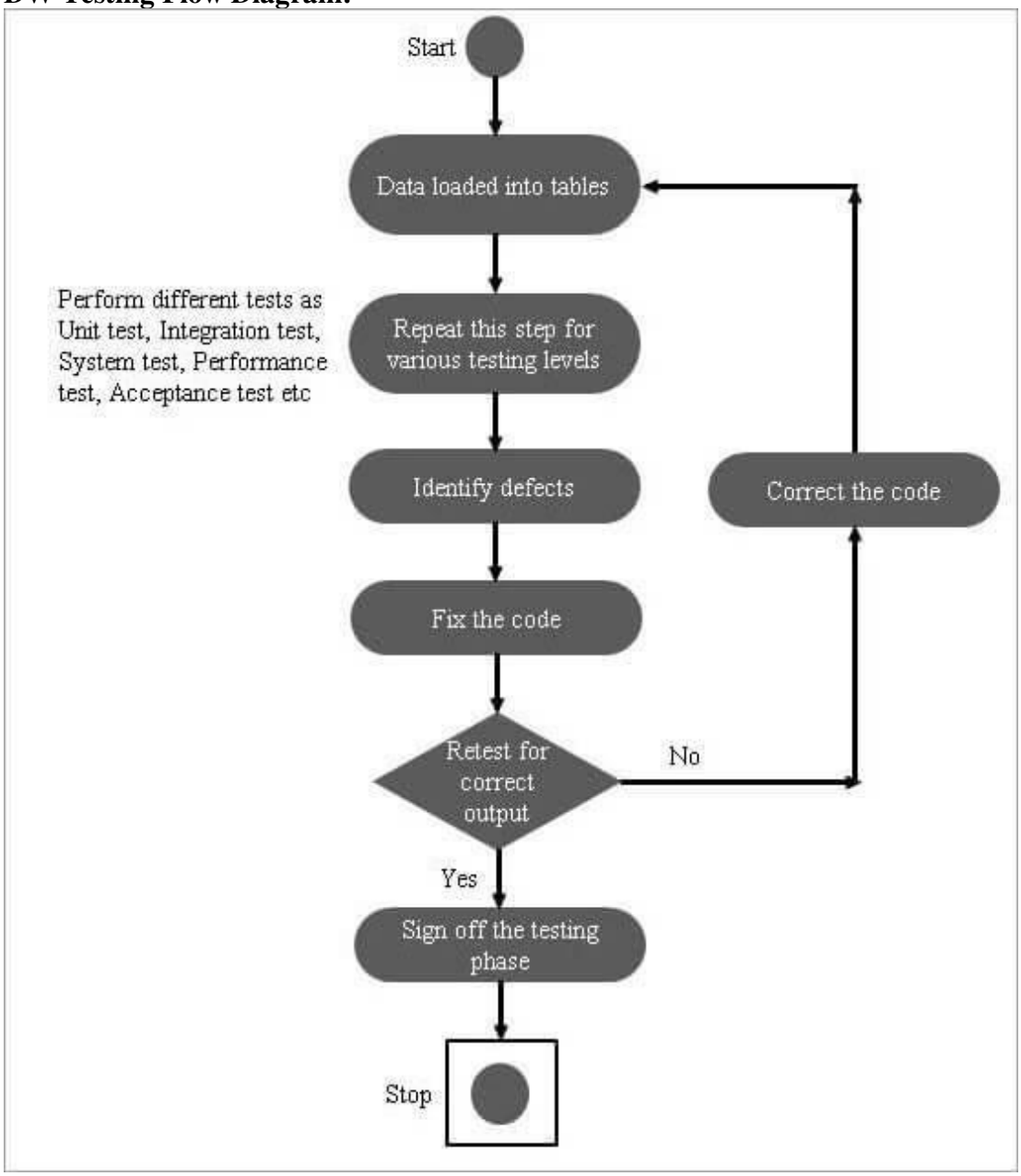
#6) Unit Testing: This is performed by the individual developers on their deliverables. Developers will prepare unit test scenarios based on their understanding of the requirements, run the unit tests and document the results. This helps the developers to fix any bugs if found, before delivering the code to the testing team.

#7) Regression Testing: Validates that the DW system is not malfunctioning after fixing any defects. This is performed many times with every new code change.

#8) User Acceptance Testing: This testing is performed by business users to validate system functionality. UAT environment is different from the QA environment. The sign off from UAT implies that we are ready to move the code to production.

From the Data Warehouse and Business Intelligence system perspective, business users can validate various reports through a User Interface (UI). They can validate the report specifications against the requirements, can validate the correctness of data in the reports, can validate how quickly the system is returning the results, etc.

DW Testing Flow Diagram:



Data Warehouse Testing Responsibilities

Enlisted below are the various teams involved in delivering a successful DW system:

- **Business Analysts:** Gather all the business requirements for the system and document those for everyone's preference.
- **Infrastructure Team:** Set up various environments as required for both developers and testers.
- **Developers:** Develop ETL code as per the requirements and perform unit tests.
- **QA (Quality Assurance)/Testers:** Develop test plan, test cases, etc. Identifies defects in the system by executing the test cases. Perform various levels of testing.
- **DBAs:** DBAs take charge of converting logical ETL database scenarios into physical ETL database scenarios and also involve in performance testing.
- **Business Users:** Involve in User Acceptance Testing, run queries and reports on DW tables.

Errors In Data Warehouse

When you are Extracting, Transforming and Loading (ETL) data from multiple sources there are chances that you will get bad data that may abort the long-running jobs.

Following are the key causes of failure in the DW system:

#1) Business Rule Violations (Logical Errors): Logically wrong data violates the business rules. Such data can be handled mostly during transformation or loading phases.

#2) Data Rule Violations (Data Errors): Data errors occur inside the DW database system like data type mismatches, data constraint failures, etc.

ETL Deployment

This is the phase where all your efforts go live. All the production support documents should be prepared.

The documentation will tell others about the sequence of jobs to run, failure recovery scenarios, training materials to the DW support teams to monitor the system after deployment and to the administrative support team to execute the reports.

RESULT:

Thus the Goals of Data Warehouse Testing, ETL Testing Responsibilities, Errors in DW and ETL Deployment in detail in this tutorial.