**Big-Data Project**

**on**

**Census Data using Hadoop Eco-System Products**

**Prepared By:**

* **Annu Sharma**

**BigData Analysis on Census Data in Hadoop**

**Abstract**

**Big Data**

Big data is a term that describes the large volume of data – both structured and unstructured – that inundates a business on a day-to-day basis. But it’s not the amount of data that’s important. It’s what organizations do with the data that matters. Big data can be analysed for insights that lead to better decisions and strategic business moves.

**Hadoop**

Hadoop is an open-source software framework for storing data and running applications on clusters of commodity hardware. It provides massive storage for any kind of data, enormous processing power and the ability to handle virtually limitless concurrent tasks or jobs.

**Hadoop History**

As the World Wide Web grew in the late 1900s and early 2000s, search engines and indexes were created to help locate relevant information amid the text-based content. In the early years, search results were returned by humans. But as the web grew from dozens to millions of pages, automation was needed. Web crawlers were created, many as university-led research projects, and search engine start-ups took off (Yahoo, AltaVista, etc.).

One such project was an open-source web search engine called Nutch – the brainchild of Doug Cutting and Mike Cafarella. They wanted to return web search results faster by distributing data and calculations across different computers so multiple tasks could be accomplished simultaneously. During this time, another search engine project called Google was in progress. It was based on the same concept – storing and processing data in a distributed, automated way so that relevant web search results could be returned faster.

In 2006, Cutting joined Yahoo and took with him the Nutch project as well as ideas based on Google’s early work with automating distributed data storage and processing. The Nutch project was divided – the web crawler portion remained as Nutch and the distributed computing and processing portion became Hadoop (named after Cutting’s son’s toy elephant). In 2008, Yahoo released Hadoop as an open-source project. Today, Hadoop’s framework and ecosystem of technologies are managed and maintained by the non-profit Apache Software Foundation (ASF), a global community of software developers and contributors.

**Why is Hadoop important?**

* **Ability to store and process huge amounts of any kind of data, quickly.** With data volumes and varieties constantly increasing, especially from social media and the Internet of Things (IoT), that's a key consideration.
* **Computing power.** Hadoop's distributed computing model processes big data fast. The more computing nodes you use, the more processing power you have.
* **Fault tolerance.** Data and application processing are protected against hardware failure. If a node goes down, jobs are automatically redirected to other nodes to make sure the distributed computing does not fail. Multiple copies of all data are stored automatically.
* **Flexibility.** Unlike traditional relational databases, you don’t have to pre-process data before storing it. You can store as much data as you want and decide how to use it later. That includes unstructured data like text, images and videos.
* **Low cost.** The open-source framework is free and uses commodity hardware to store large quantities of data.
* **Scalability.** You can easily grow your system to handle more data simply by adding nodes. Little administration is required.

**Hadoop Glossary**

#### Currently, four core modules are included in the basic framework from the Apache Foundation:

**Hadoop Common** – the libraries and utilities used by other Hadoop modules.

**Hadoop Distributed File System (HDFS)**– the Java-based scalable system that stores data across multiple machines without prior organization.

**YARN**– (Yet Another Resource Negotiator) provides resource management for the processes running on Hadoop.

**MapReduce** – a parallel processing software framework. It is comprised of two steps. Map step is a master node that takes inputs and partitions them into smaller sub problems and then distributes them to worker nodes. After the map step has taken place, the master node takes the answers to all of the sub problems and combines them to produce output.

**Hadoop Ecosystem Products**

|  |  |
| --- | --- |
| **Ambari** | A web interface for managing, configuring and testing Hadoop services and components. |
| **Cassandra** | A distributed database system. |
| **Flume** | Software that collects, aggregates and moves large amounts of streaming data into HDFS. |
| **HBase** | A nonrelation, distributed database that runs on top of Hadoop. HBase tables can serve as input and output for MapReduce jobs. |
| **HCatalog** | A table and storage management layer that helps users share and access data. |
| **Hive** | A data warehousing and SQL-like query language that presents data in the form of tables. Hive programming is similar to database programming. |
| **Oozie** | A Hadoop job scheduler. |
| **Pig** | A platform for manipulating data stored in HDFS that includes a compiler for MapReduce programs and a high-level language called Pig Latin. It provides a way to perform data extractions, transformations and loading, and basic analysis without having to write MapReduce programs. |
| **Solr** | A scalable search tool that includes indexing, reliability, central configuration, failover and recovery. |
| **Spark** | An open-source cluster computing framework with in-memory analytics. |
| **Sqoop** | A connection and transfer mechanism that moves data between Hadoop and relational databases. |
| **Zookeeper** | An application that coordinates distributed processing. |

**Acknowledgements**

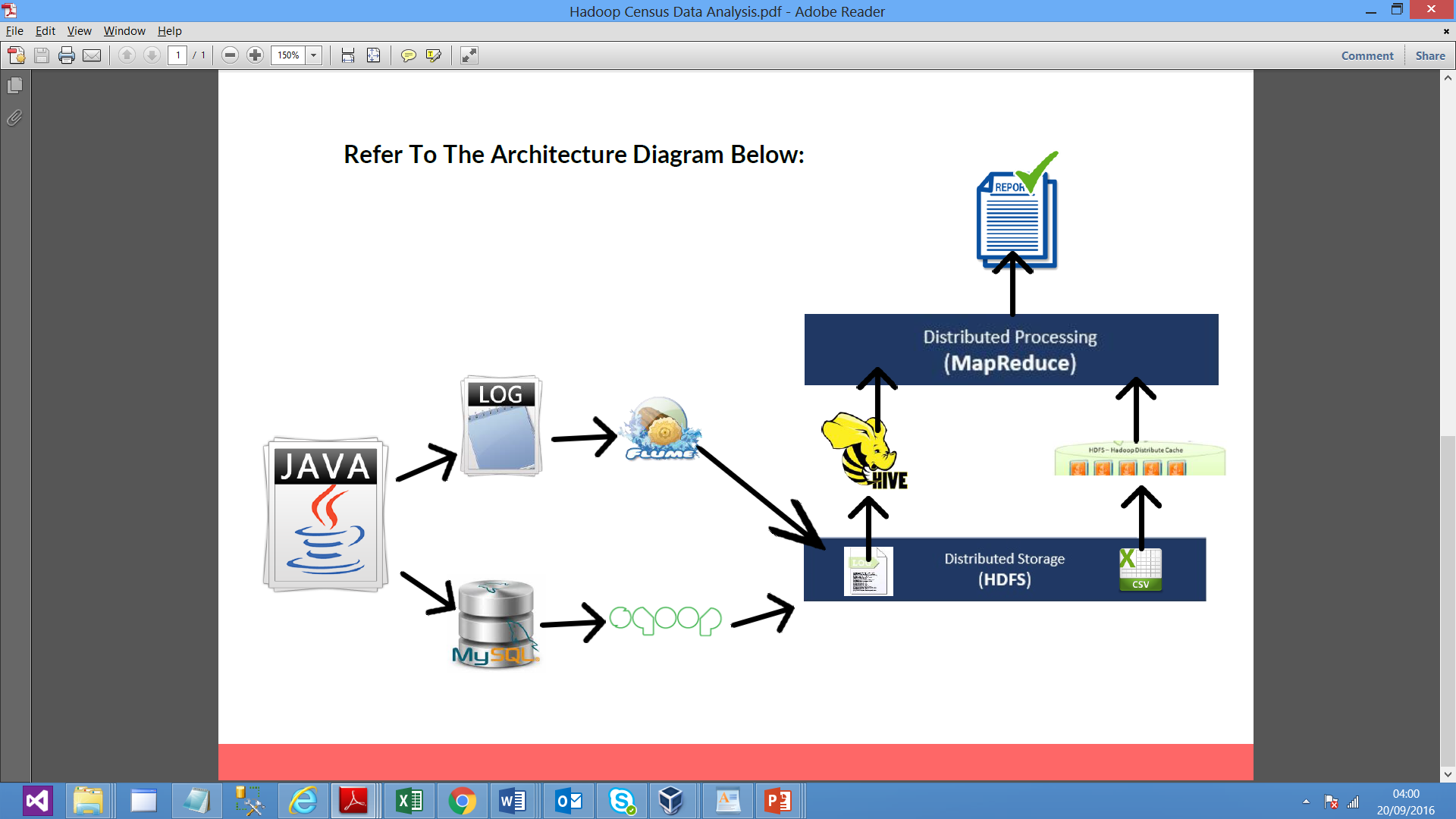
I would like to thank Mr. Abhishek and Mr. Amit for providing complete learning on Big-Data and Hadoop Eco-System products and his guidance towards completing the projects.

I want to also thank Mrs. Bharati B, Mrs. Deepti and Mr. Narender Singh for arranging this type high-end technological training and providing all support to make it successful.

**Project Outline**

|  |  |
| --- | --- |
| **Title** | **Big-Data Analysis on Census Data using Hadoop Eco-System**. |
| **Inputs** | Census Data |
| **Data Elements** | Age, Education, Marital Status, Gender, TaxFiler Status, Income, Parents, Country of Birth, Citizenship, WeeksWorked |
| **Analysis Relevance** | Education, Social, Finance and Planning |
| **Purpose** | To provide Analysed Results to Government to help it plan policies to improve social and economic life of citizens of the country. |
| **Methodology** | Agile |

**Architectural Diagram:-**

****

**Project Implementation**

**Assumptions :**

1. Hadoop Cluster is Running
2. Ecosystem Products (Pig, Hive, Sqoop) are installed
3. Census data is available on HDFS in JSON Format

**Prerequisites for All Jobs:**

The Census data is in JSON format and hence needs to be converted in csv format in hadoop filesystem.

**Steps for Conversion**

**Step 1**: Create Table in hive to read entire record as one json string

**Step 2**: Create Table to convert and store json string into different fields which will create a file on hadoop filesystem in csv format

***Note*** *: Look up / Supporting Tables to be created based on Analysis Jobs*

**Job 1 : Educational Analysis**

**Task 1 : Calculate :**

1. **Education Qualification Count : Sub-grouped by Gender**
2. **Education Qualification Count based on Employment**
3. **Calculate Sex Ratio (Male : Female)**

* ***No Supporting Table Required***

**Implementation Steps**

**1.a Calculate : Education Qualification Count : Sub-grouped by Gender**

**Technology Used : Pig**

**Pig Script :**

a = load '/user/hive/warehouse/mydb.db/census/000000\_0' using PigStorage(',') as (age:chararray,education:chararray,ms:chararray,gender:chararray);

b = foreach a generate education,gender;

c = group b by (education,gender);

d = foreach c generate group.education,group.gender,COUNT(b.gender);

dump d;

**Sample Output :**

**( Children, Male,228)**

**( Children, Female,224)**

**( 9th grade, Male,27)**

**( 9th grade, Female,29)**

**( 10th grade, Male,38)**

**1.b Calculate : Education Qualification Count based on Employment**

**Technology Used : Pig**

**Pig Script :**

a = load '/user/hive/warehouse/mydb.db/census/000000\_0' using PigStorage(',') as (age:chararray,education:chararray,ms:chararray,gender:chararray,taxpayer:chararray,income:double,parents:chararray,cob:chararray,citizen:chararray,weekworked:int);

b = foreach a generate education,weekworked;

c = filter b by weekworked>0;

d = group c by education;

e = foreach d generate group,COUNT(c.weekworked);

dump e;

**Solution:**

**( 9th grade,25)**

**( 10th grade,37)**

**( 11th grade,42)**

**( 5th or 6th grade,12)**

**1.c Calculate Sex Ratio (Male : Female)**

**Technology Used : Pig**

**Pig Script :**

a = load '/user/hive/warehouse/mydb.db/census/' using PigStorage(',') as (age:int,education:chararray,ms:chararray,gender:chararray);

b = foreach a generate gender;

c = group b by gender;

d = foreach c generate group,COUNT(b.gender);

dump d;

**Solution:**

**( Male,939)**

**( Female,1061)**

**Job 2 : Financial Analysis**

**Task 1 : Calculate :**

1. **Tax based Income Generated**
2. Total Income Generated , Gender wise Total Income Generated
3. Total Tax Payers
4. Total Tax to be collected
5. **Per Capita Income Analysis**
6. Per Capita Income
7. Age Group wise Per Capita Income
8. Gender wise Per Capita Income

**Supporting Tables Required :**

1. **Tax\_Mapping Table:**

**Table Definition :**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Column Type** | **Remarks** |
| Tid | int | Primary Key |
| minincome | int | Stores min salary of a salary slab |
| maxincome | int | Stores max salary of a salary slab |
| taxper | double | Tax (%) for a salary slab |

**Supporting Data File Required :**

1. **agegroup :** Data file in csv format on hadoop filesystem

**Task 1.a : Calculate :Tax based Income Generated**

1. Total Income Generated , Gender wise Total Income Generated
2. Total Tax Payers
3. Total Tax to be collected

**Technology Used : MapReduce Job using Java**

**Implementation Highlights :**

1. **MapSide Join performance**
   1. Supporting Table – Tax\_Mapping Data is fetched in setup() of Mapper
   2. Data of Tax\_Mapping stored in Collection in sorted order
   3. Census data read line by line in map() method wherein income value is compared with Tax\_Mapping records and taxpercentage is retrieved.
   4. Income, Taxpercentage, Gender and TaxFiler Status sent as Values to Reducer
   5. Reducer calculates the total tax as required and generates the output

**Task 1.b : Perform Per Capita Income Analysis**

1. **Per Capita Income**

**Technology Used : Pig**

**Pig Script :**

a = load '/user/hive/warehouse/mydb.db/census/000000\_0' using PigStorage(',') as (age:int, education:chararray,ms:chararray,gender:chararray,taxfiler:chararray,income:double,parents:chararray,cob:chararray,citizenship:chararray,weeks:int);

b = foreach a generate income;

c = group b all ; -- // describe c

d = foreach c generate SUM(b.income) /COUNT(b.income) as pci ;

dump d;

**Solution:**

**(1728.2616350000017)**

1. **Age Group wise Per Capita Income**

**Technology Used : Pig**

**Pig Script :**

a = load '/user/cloudera/agegroup.dat' using PigStorage('\t') as (age:int,cat:chararray);

b = load '/user/hive/warehouse/mydb.db/census/000000\_0' using PigStorage(',') as (age:int, education:chararray,ms:chararray,gender:chararray,taxfiler:chararray,income:double,parents:chararray,cob:chararray,citizenship:chararray,weeks:int);

c = foreach b generate income,age;

d = foreach a generate cat,age;

e = join c by age,d by age ;

f = foreach e generate cat,income;

g = group f by cat;

h = foreach g generate group,SUM(f.income)/COUNT(f.income) as pci;

dump h;

**Sample Output**

**(adult,1828.8573029045647)**

**(elderly,1822.1314705882355)**

**(infants,1632.2484130982368)**

**(Teenager,1758.8825362318848)**

**(middle-aged,1671.9536820083677)**

**(senior citizen,1662.542173913042)**

1. **Gender wise Per Capita Income**

**Technology Used : Pig**

**Pig Script :**

a = load '/user/hive/warehouse/mydb.db/census/000000\_0' using PigStorage(',') as (age:int, education:chararray,ms:chararray,gender:chararray,taxfiler:chararray,income:double,parents:chararray,cob:chararray,citizenship:chararray,weeks:int);

b = foreach a generate income,gender;

c = group b by gender ; -- // describe c

d = foreach c generate group, SUM(b.income) /COUNT(b.income) as pci ;

dump d;

**Sample Output**

**( Male,1792.9552289669857)**

**( Female,1671.0068897266733)**

**Job 3 : Social Analysis**

**Task 1 : Calculate :**

1. **Pension Amount to be added after x years**
2. **No. of Orphans for each category based on Parents Present**
3. **No. of Employable Female Citizens who are Widows or Divorced**

**Supporting Tables Required :**

1. **Pension\_Mapping Table:**

**Table Definition :**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Column Type** | **Remarks** |
| Pid | int | Primary Key |
| min\_income | int | Stores min salary of a salary slab |
| max\_income | int | Stores max salary of a salary slab |
| pension | int | Pension amount paid in a salary slab |

1. **Orphan\_Mapping Table:**

**Table Definition :**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Column Type** | **Remarks** |
| Oid | int | Primary Key |
| parent\_present | varchar | Status of Parents present |
| subsidy | int | Subsidy amount paid to orphan |

**Task 1.a : Calculate : Pension Amount to be added after x years**

**Technology Used : MapReduce Job using Java**

**Implementation Highlights :**

1. MapSide Join performed
   1. Supporting Table – Pension\_Mapping Data is fetched in setup() of Mapper from Distributed Cache
   2. Data of Pension\_Mapping stored in Collection in sorted order
   3. Census data read line by line in map() method wherein income value is compared with Pension\_Mapping records and pension amount is retrieved.
   4. Income and Pension Amount sent as Values to Reducer
   5. Reducer calculates the total pension amount to be added after the no. of years as inputted by user and generates the output
2. No. of Years to be taken as User Input in Driver and is read by Mapper in map() method.

**Task 1.b : Calculate : No. of Orphans for each category based on Parents Present**

**Technology Used : Pig**

**Pig Script :**

a = load '/user/hive/warehouse/mydb2.db/census/' using PigStorage(',') as (age:int,education:chararray,ms:chararray,gender:chararray,taxfiler:chararray,income:double,parents:chararray,cob:chararray,citizenship:chararray,week:int);

b = foreach a generate parents;

c = filter b by TRIM(parents)!='Both parents present' ;

d = group c by parents;

e = foreach d generate group, COUNT(c.parents) as total\_count;

dump e;

**Sample Output**

**( Not in universe,1472)**

**( Father only present,17)**

**( Mother only present,133)**

**( Neither parent present,18)**

**Task 1.c : Calculate : No. of Employable Female Citizens who are Widows or Divorced**

**Technology Used : Pig**

**Pig Script :**

a = load '/user/hive/warehouse/mydb2.db/census/' using PigStorage(',') as (age:int,education:chararray,ms:chararray,gender:chararray,taxfiler:chararray,income:double,parents:chararray,cob:chararray,citizenship:chararray,week:int);

b = filter a by TRIM(ms)=='Divorced' or TRIM(ms)=='Widowed';

c = filter b by TRIM(gender)=='Female';

d = group c by ms;

e = foreach d generate group,COUNT(c.ms) ;

dump e;

**Sample Output**

**( Widowed,75)**

**( Divorced,70)**

**Job 4 : Planning Analysis**

**Task 1 : Calculate :**

1. **No. of Voters to get added in next X years**
2. **No. of Senior Citizen to get added in next X years**
3. **Sex Ratio**
4. **Citizen vs. Immigrants Ratio for all Employed**

**Task 1.a : Calculate No. of Voters to get added in next X years**

**Technology Used : MapReduce Job using Java**

**Implementation Highlights :**

1. No. of Years to be taken as User Input in Driver and is read by Mapper in map() method
   1. Census data read line by line in map() method
   2. Year value inputted is passed as key and age are outputted from mapper to reducer
   3. Reducer sets the current age value based on year value inputted and matches with the age in every record wherein its assumed that 18years is the minimum age for a citizen to be a voter.
   4. No. of users whose age is x years less than 18 are counted and written to output by reducer

**Task 1.b : Calculate No. of Senior Citizen to get added in next X years**

**Technology Used : MapReduce Job using Java**

**Implementation Highlights :**

1. No. of Years to be taken as User Input in Driver and is read by Mapper in map() method
   1. Census data read line by line in map() method
   2. Year value inputted is passed as key and age are outputted from mapper to reducer
   3. Reducer sets the current age value based on year value inputted and matches with the age in every record
   4. No. of users whose age is x years less than 65 are counted and written to output by reducer

**Task 1.c : Sex Ratio**

**Technology Used : Pig**

**Pig Script :**

a = load '/user/hive/warehouse/mydb.db/census/' using PigStorage(',') as (age:int,education:chararray,ms:chararray,gender:chararray);

b = foreach a generate gender;

c = group b by gender;

d = foreach c generate group,COUNT(b.gender);

dump d;

**Sample Output**

**( Male,939)**

**( Female,1061)**

**Task 1.d : Citizen vs. Immigrants Ratio for all Employed**

**Technology Used : Pig**

**Pig Script :**

**Sample Output**

a = load '/user/hive/warehouse/mydb2.db/census/' using PigStorage(',') as (age:int,education:chararray,ms:chararray,gender:chararray,taxfiler:chararray,income:double,parents:chararray,cob:chararray,citizenship:chararray);

b = foreach a generate citizenship;

c = group b all;

d = foreach c generate 1 as flag,COUNT(b.citizenship) as total\_count;

e = filter b by (citizenship matches '.\*Not a citizen.\*');

f = group e all;

g = foreach f generate 1 as flag,COUNT(e.citizenship) as fore\_count;

h = join d by flag,g by flag;

i = foreach h generate fore\_count,total\_count-fore\_count;

dump i;

**Sample Output**

**(135,1865)**

**Hardware Software Configuration:-**

* Hardware :-
  + 8 GB Ram
  + Quad Core Processor
  + 100 GB HD
* Software :-
  + Virtual Box
  + Ubuntu & Cloudera virtual .ova file
  + Hadoop , HDFS, PIG, Hive, Scoop, Flume, MySQL, Java, TomCat

**Database:-**

1. MySQL
2. NoSQL

**Technological Uses inside the project:-**

1. Hive is used to create a Table after taking the data from JSON format.
2. MySQL is used to create support table
3. Sqoop is used to import the data from MySql
4. Pig is used to run the script in place of Map-Reduce for achieving the objective.
5. Mapreduce is used here to implement the logic to achieve any objective.

**Conclusion**

Following is the conclusion that we can draw based on the tasks performed by us :

1. Sqoop is useful when we have data in SQL Tables that need to be imported in hadoop filesystem
2. Hive helps in cleaning up of data. JSON Data can be easily converted to csv format using Hive
3. For normal group by , join and filter based data retrieval Pig is very efficient.
4. MapReduce code written in Java makes the complex analysis quite easy. Codes required to be written to collect user inputs and performing complex join operations are handled efficiently using this approach

**Webo-Graphy:-**

1. https://www.youtube.com/watch?v=Pq3OyQO-l3E&list=PLpc4L8tPSURCdIXH5FspLDUesmTGRQ39I&index=1
2. https://www.youtube.com/watch?v=DLutRT6K2rM&index=2&list=PLpc4L8tPSURCdIXH5FspLDUesmTGRQ39I
3. https://www.youtube.com/watch?v=6OemZEJdMp8&index=3&list=PLpc4L8tPSURCdIXH5FspLDUesmTGRQ39I
4. https://www.youtube.com/watch?v=rF5Urjosi9g&index=4&list=PLpc4L8tPSURCdIXH5FspLDUesmTGRQ39I
5. https://www.youtube.com/watch?v=9bYgOeD5j9E&index=5&list=PLpc4L8tPSURCdIXH5FspLDUesmTGRQ39I
6. https://www.youtube.com/watch?v=WIMlCvAVKps&list=PLpc4L8tPSURCdIXH5FspLDUesmTGRQ39I&index=6
7. https://www.youtube.com/watch?v=BcqiFsWZD1M&index=7&list=PLpc4L8tPSURCdIXH5FspLDUesmTGRQ39I
8. https://www.youtube.com/watch?v=JQdhWTp\_m0U&index=8&list=PLpc4L8tPSURCdIXH5FspLDUesmTGRQ39I
9. <https://www.youtube.com/watch?v=u8HarSx7glE&list=PLpc4L8tPSURCdIXH5FspLDUesmTGRQ39I&index=9>
10. Word Count Example :- https://www.youtube.com/watch?v=Df2Odze87dE