# AGRISOCIETY NETWORK MANAGEMENT SYSTEM

# TEAM 30

Prepared By:

Bhatt Bhavik(18BIT010)

Aaryan Satpal(18BIT001)

Jaimin Rana(18BIT036)

Manan Bharwad(18BIT061)

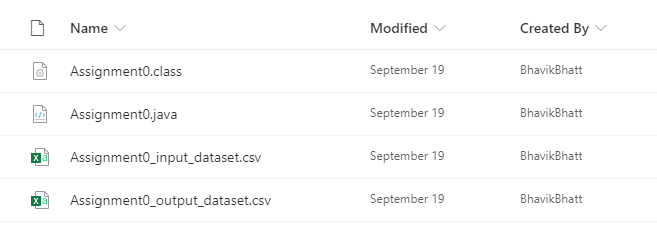
**INDEX**

|  |  |
| --- | --- |
| Topic | Page no. |
| 1. Experiment description | 3 |
| 2.Dataset query program | 8 |
| 3.Project Description | 11 |
| 4.DBMS vs File system | 13 |
| 5.Relational model | 17 |
| 6.Relational algebra | 31 |
| 7.ER Model | 32 |
| 8. Installation of MYSQL workbench | 34 |
| 9.Create table Syntax | 35 |
| 10.Filled tables | 42 |
| 11.SQL queries | 49 |
| 12.Normalization of tables | 56 |
| 13.Assignment topic (Stored procedures) | 68 |
| 14.Research publication | 72 |
| 15.Our contribution | 73 |

# EXPERIMENT DESCRIPTION

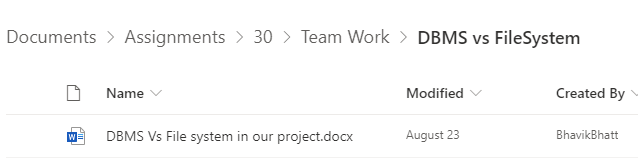
## Experiment 0

## (For assignment 0, dataset was taken in csv file and query was written in java programming and input dataset, output dataset and java code are uploaded here)



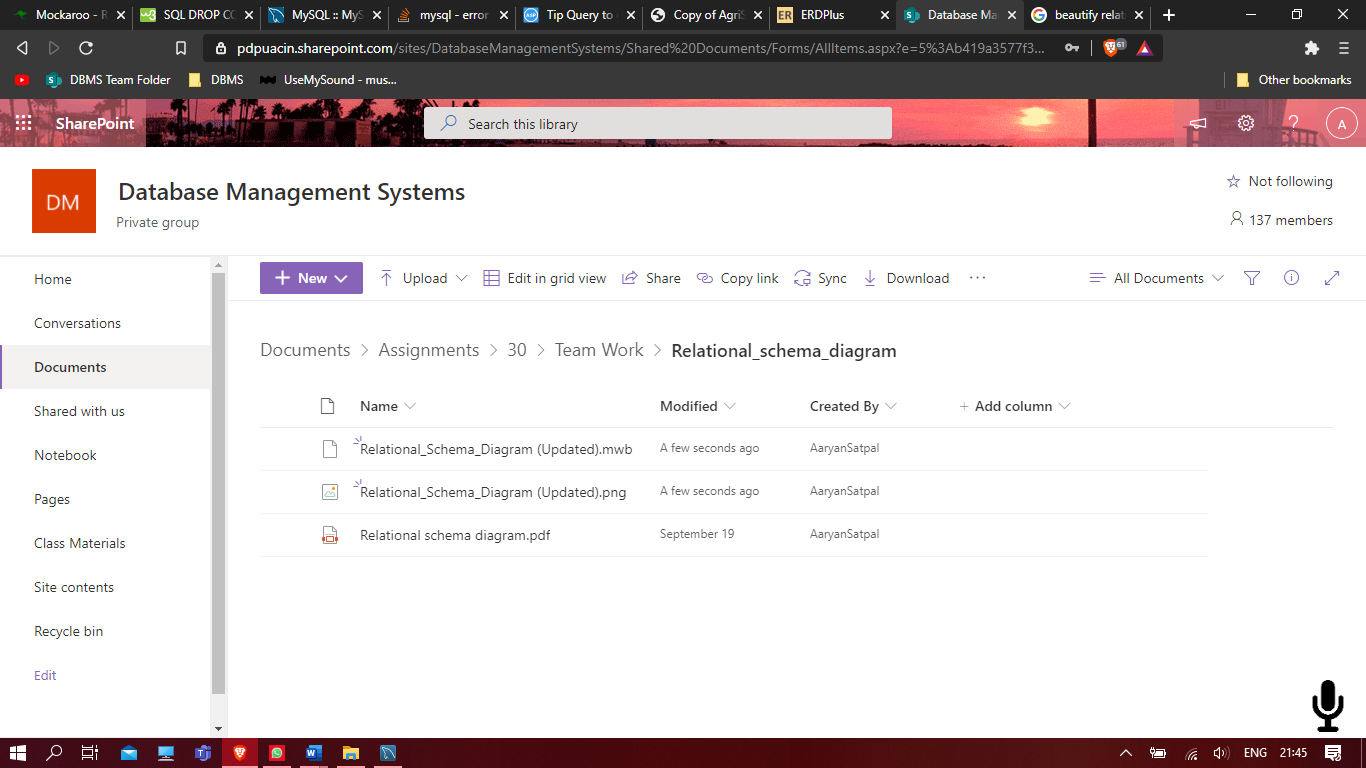
* **Experiment 1**

(DBMS vs File system)

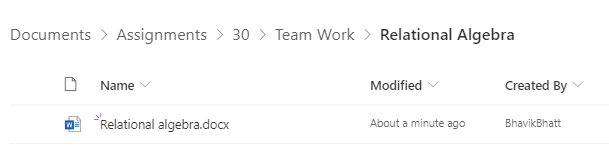


* **Experiment 2**

(Relational model)

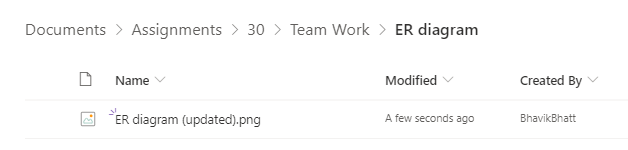


* **Experiment 3**

(Relational algebra) 

* **Experiment4**

(ERDiagram)

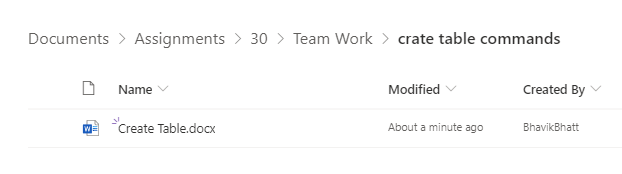


* **Experiment5**

(All the our team member installed MYSQL Workbench and one member installed SQL server management studio)

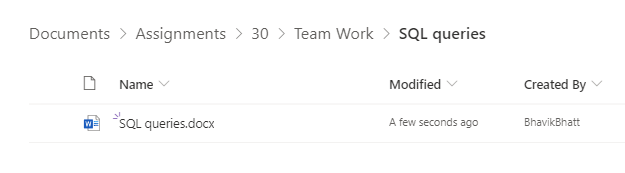
* **Experiment 6**

(Create table syntax)



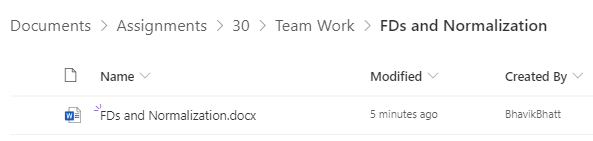
* **Experiment 7,8,9,10**

(SQL queries)



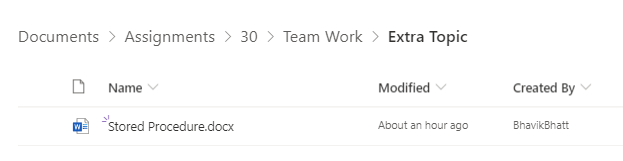
* **Experiment 11**

(Database normalization)



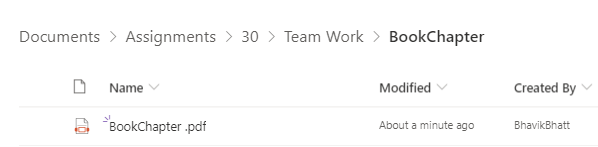
* **Experiment 12**

(Given topic Stored procedure)

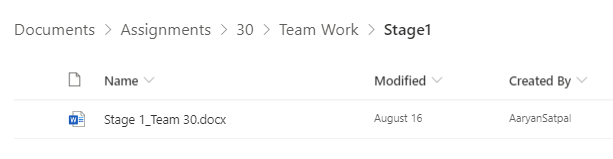


* **Experiment 15**

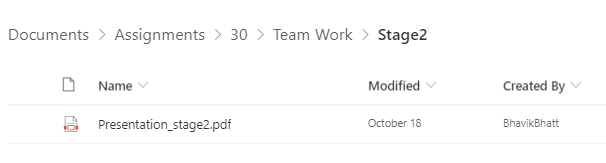
(Book Chapter)



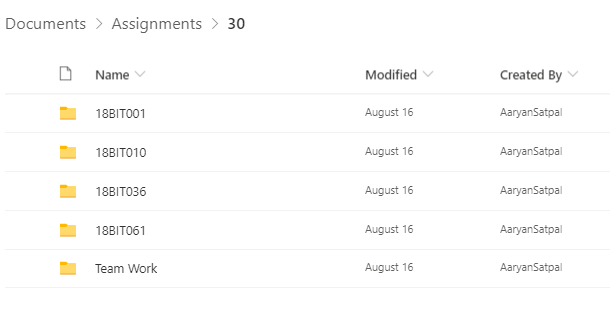
* **Stage1**

****

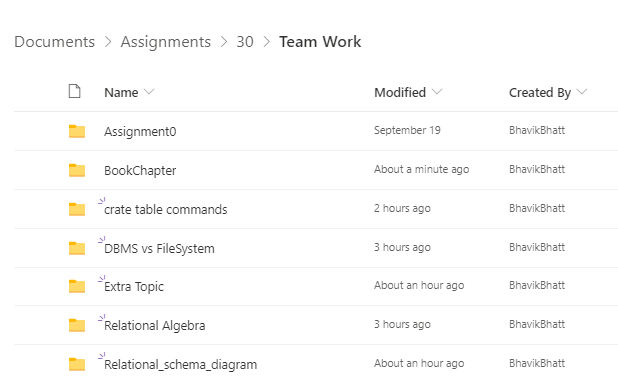
* **Stage2 (Presentation given in class)**

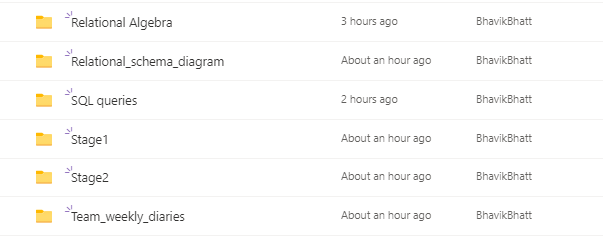
****

* **Personal and Team work folders**

****

* **Team work folder**

****

****

# Dataset query program

Java program for registered murder cases district wise from CSV file of overall registered crime cases' dataset and store that in CSV file..

Program:

/\* Query: Make a list of number of registered murder cases district wise from CSV file of overall registered crime cases' dataset and store that in CSV file. \*/

//package com.java.createcsvfile;

import java.io.FileWriter;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import java.util.\*;

public class Assignment0 {

public static FileWriter writer=null;

public static String location;

public static void generateCsvFile(String fileName)

{

/\*This method will create csv file where output will be stored \*/

location="Assignment0\_output\_dataset.csv";

try{

writer = new FileWriter(location);

}

catch(IOException e){

e.printStackTrace(); }

}

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String line="";

String splitby=",";

int c=0;

int i=0;

int l=0;

String [] district=new String[9040];

try{

generateCsvFile(location);

//query to find districts with less than 100 Murders in 2001

BufferedReader br=new BufferedReader(new FileReader("Assignment0\_input\_dataset.csv"));

/\* here Assignment0\_input\_dataset is the source dataset \*/

System.out.println("Stated districts below had less than 100 murder cases");

writer.append("District"+","+"State"+","+"Murder Cases");

while((line=br.readLine())!=null){

if(c==0){

c=1;

}

else{

String[] data=line.split(splitby);

int k= Integer.parseInt(data[3]);

if((k<100 && data[2].equals("2001"))){

System.out.println(data[1]+" "+ "of " + data[0]);

writer.append('\n');

writer.append(data[1]+","+data[0]+","+data[3]);

writer.append(",");

i++;

writer.flush();

// writer.close();

}

}

}

}

catch(IOException e){

e.printStackTrace();

}

/\*finally{

writer.flush();

writer.close();

} \*/

}

}

# PROJECT DESCRIPTION

In this project we are making an e-platform that will manage a network of people in agriculture business. We are going to connect them on the basis of their requirements In abstract this project will lead us to a platform that will provide more opportunities for agricultural growth by developing a proper business system and avoiding problems like price rise in a society. Need is to control price rises of basic needs of society and it is possible by proper management of base of a system (agribusiness). A major purpose of this project is to control hoarding of essential products by providing a better and reliable business platform with greater opportunities for small business holders.

* How we added IOT devices to our database?

🡪In our sensor based irrigation system we embedded moisture sensor, temperature sensor and methane sensor.

🡪The reading those will be shown by these sensors will be stored in table named Irrigation\_system.

🡪And the developers will take the data from tables like farm\_area from farm table, moisture, temperature and methane information from irrigation\_system table and growing crop info from farm table and the needed levels of element will be compared with the existing and the call like watering crop will be taken.

* Here is some information about out IOT devices:

|  |  |
| --- | --- |
| IOT device name | In which table it is? |
| Moisture sensor | irrigation\_system |
| Temperature sensor | irrigation\_system |
| Methane sensor | irrigation\_system |

|  |  |
| --- | --- |
| IOT device name | What it does? |
| Moisture sensor | Shows moisture level |
| Temperature sensor | Shows temperature |
| Methane sensor | Methane element level |

**Outcomes and potential impacts of this project**:

* As there will be all the data about past and present of production of crop and products, government can take further decision about what to import and what to export for a country
* People involved in agriculture, will be well aware of market situation and government’s new schemes
* Circulation of crop or agricultural products will be known to government and if in any circumstances government find lack of crop than according to circulation and production rate they can find if there is hoarding of things happening or not
* Agriculture business world will be reachable and narrow.
* Government and agribusiness holders will be able to interact with each other
* Price rises of basic needs will be controlled due to transparent platform
* There will be a transparent and efficient platform for agribusiness
* Due to gathering of many different professions on a single platform there will be more options for people to grow their business and to connect with people
* There will be less fraud problem because there will not be any unauthorized account and even if any fraud happen than due to deal record, cheater will be punished
* There will be an online platform after completion of this project that will suggest and offer solutions for almost all agricultural businesses.

**Benefits**:

* “Hoarding of basic products” will be prohibited
* Poor business holder will have more opportunities
* Same types of products can be made available at nearly same price
* Proper requirement based cycle will be generated among people
* Efficient guidance will be available using features of platform
* Efficient use of resources
* An efficient network can be made
* Control on price rises of daily essential products

# DBMS vs Filesystem

* **Data redundancy**

In our project we are having some information related to one user, and to represent that if we use file system we might not have facility to store them properly and data may be stored redundantly.

Examples:

🡪Our platform is based on requirement of people related to agriculture and we are making record to show user’s requirement on platform and certain requirements may not change with time and if we use file system there than there will be several DR issues compare to DBMS.

🡪In our project there is a department of transportation and while showing live location of transportation vehicle there might be possibility that vehicle may not be driven to other place due to some of reason and in that case live location will not be change for specific time and it may cause DR problem. And so we can’t use file system here.

🡪In our project we are storing info of farmer’s previous season’s crop and there are some types of farming which can be happened all the seasons and in that case previous season’s crop value will be repeated and that may generate DR.

🡪On our platform we are storing information of some of users whose past data we are storing but that might be possibility that it will not change with time and will be repeated. For example, we farmer may not have leased land for many years and so that field will have same value for long time and it will create DR.

* **Data inconsistency**

Examples:

🡪If registered organization or shop has been sold by registered owner than all the related information of shops and owner must be changed and if we store such information in file system than there will be difficulties and we may need to make another files to represent new data but while in DBMS data edition will be easier and it will be easy to handle data inconsistency.

🡪If any user leaves platform than we have to remove all the data related to that user from platform and if that data is stored in different files, it may create problem to remove that data because it is scattered but if we handle that data using DBMS than adding or removing data will be easier and DI will be handled.

🡪We are registering address and contact number of a user and using them at some specific places but if user changes them than we must have to change them from everywhere they exist else there will be data inconsistency and to handle this DBMS is easier to use compare to file system.

🡪On our platform we are registering workers of small transportation company and if owner sells his one of vehicle or worker stops working than information related to them must be changed else there will be data inconsistency and it is possible to handle easily in DBMS.

* **Atomicity:**

Examples:

🡪In our application if transportation provider has taken an order so after order is taken than availability status of vehicle should be changed.

🡪Fertilizer merchant will show available stock of fertilizer but if he gets an order and if order is accepted than at that point of time stock updates should be changed.

🡪On our application, crop buyer will add his required crops for buying but until he updates that list and buys crop of interest than certain changes in required crop record should be done.

🡪Once deal between two parties has been done than deal details should be added in deal record.

* **Difficulty in accessing data:**

Examples:

🡪On our application there are some sensitive information related to user’s business and it must be accessed to user only and to do so we must set some criteria by adding some constraints and if we follow file system than we have to do different coding for different kind of data access and hence DBMS is useful in such condition.

🡪As our project is related to agricultural business so if government want to analyse specific data related to agribusiness than they have to access or retrieve some data and it is easier to do with DBMS than file system.

🡪If any farmer wants to search about in what price crop buyers want to buy crop than he will have organised results for his search if database for that subject is created using DBMS as data access will be easier.

🡪If users on application want to know live location of transportation provider than that real time data can be accessed easily than File system.

* **Security issue**

🡪We are having deal record between two parties and if any cheating occurs than it can be inspected from record but because of real time addition in record it is quite difficult to do with file system than DBMS.

🡪On our platform we are asking information from user and in that there are some points to be shown on platform and some of are not to be shown on platform due to security purpose and to do so file system can’t provide access or constraints and it is easy to do so with DBMS.

🡪We are not allowing any random person to register on platform without specific information and it is possible by adding some conditions or qualification criteria to register and it will be very complicated with file system as there we will need to write several codes but it can be done using DBMS.

🡪As all the data of users and their actions is on server, if data gets deleted somehow than it can be restored using back up facility but in case of file system it is difficult.

* **Integrity issues**

Examples:

🡪On platform, we are making record for deal happened between two parties and there are no restriction on number of deals to do in a day for a user and to handle such a condition file system is not reliable and DBMS can provide such feature.

🡪In our database, there are some parameters like contact number, time and date, and rating of service which should contain specific type and range and to do so DBMS is feasible solution.

🡪On our virtual platform we are seeking information like Adhaar card number from user and Adhaar card number is of 12 digits, if user does any mistake while entering it system must stop it and it can be done in DBMS by adding constraints.

🡪For farmer on our platform, we are asking him about he has leased land or not and answer should be in ‘Y’ or ‘N’ and similar example for transportation driver about his availability at current time and he must add this detail with ‘Y’ and ‘N’ and not in another form so this kind of situations are easier to handle in DBMS than file system.

* **Data abstraction**

Examples:

🡪To register on our platform, we are asking information form farmer about contact number, address, current crop in his farm but apart from that for analysis purpose and to predict some schemes for them we are asking some information about Loan and area of leased land and so we must hide such information and we should show his abstract information form on platform and which is possible by creating back end system by DBMS.

🡪We are asking yearly income of users to analyse average income of users and to compare current income with past’s income and this should not be visible to other users apart from details of visible business information of users and even in this case data abstraction helps to hide some information using DBMS.

🡪There will be details needed to check proof of approval from government for “soil analysis lab” and “fertilizer factories” and such information should not be visible to others apart from organization’s visible business information. And this is the concept of data abstraction.

🡪Transportation provider will have to give some information like RC number of vehicles, vehicle number, vehicle types but in that data set RC number should not be shown on platform but to show other information data abstraction will be used by DBMS.

* **Concurrent access to data:**

# on our virtual system, there are some processes which will be done parallel and if we use file system to do so then concurrent access will not be good option and so we will use DBMS for them and examples are given below.

🡪If farmer and crop buyer does a deal and they have to add their deal details in farmer-crop buyer business record but there is possibility that more than one pair will do deal and update record at the same time.

🡪Similar example is of fertilizer dealer and farmer that even they will have business deal records and multiple deals between different pairs may occur and they simultaneously add their deal entry.

🡪 Transportation provider will take orders for transportation and accordingly availability of transportation provider should be changed and there will be possibility that at same time he might get more than 1 order so data must be handled in such case.

🡪Fertilizer merchant will get order for fertilizer and if he gets more than 1 order at a time than there should be some criteria whom to deliver first and availability stock of fertilizer must be changed accordingly. So in such case of simultaneous access, DBMS is feasible solution.

* **Isolation of data:**

Examples:

🡪There might be possibility that if application adds new kind of user type with allocated features for them then there is possibility that new features are added by new logical language and as platform is about connecting different kind of users than platform must show compatibility with new technology and so application’s database should be made using DBMS else there will be data isolation.

🡪As our project is about application for agribusiness, so there are high chances that with time passing we need to add some features and if we use file system to create database and if in past we used specific language to add features and now we want to add another feature using another language than there might be possibility that we can’t connect certain data of past in file system with new language and it can create data isolation problem.

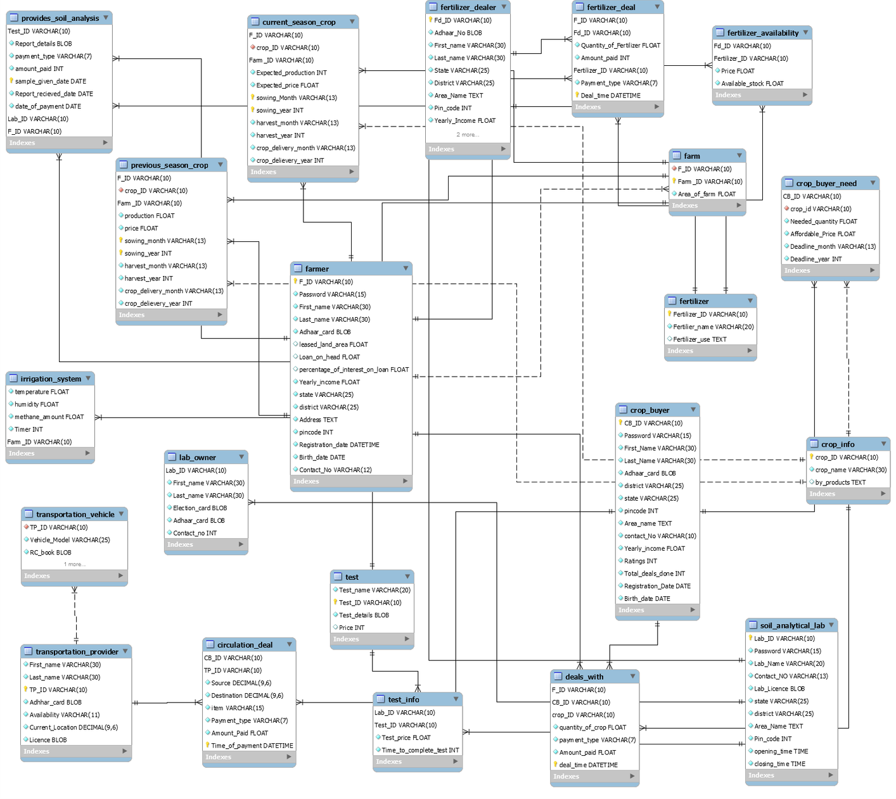
🡪If we add new features for farmers by seeking some information of them by using new language than the language used for existing data than there should be compatibility between new and old data and as DBMS is compatible with different languages it can be done and data isolation will be avoided.

🡪In our project, if we want to add voice feature for transportation customer for giving information about vehicle location and to do so we have to make additional data set required for voice feature by connecting location data base that might be added using different language than language is used for voice feature but still we have to have compatibility and it is possible using DBMS only and there will not be data isolation.

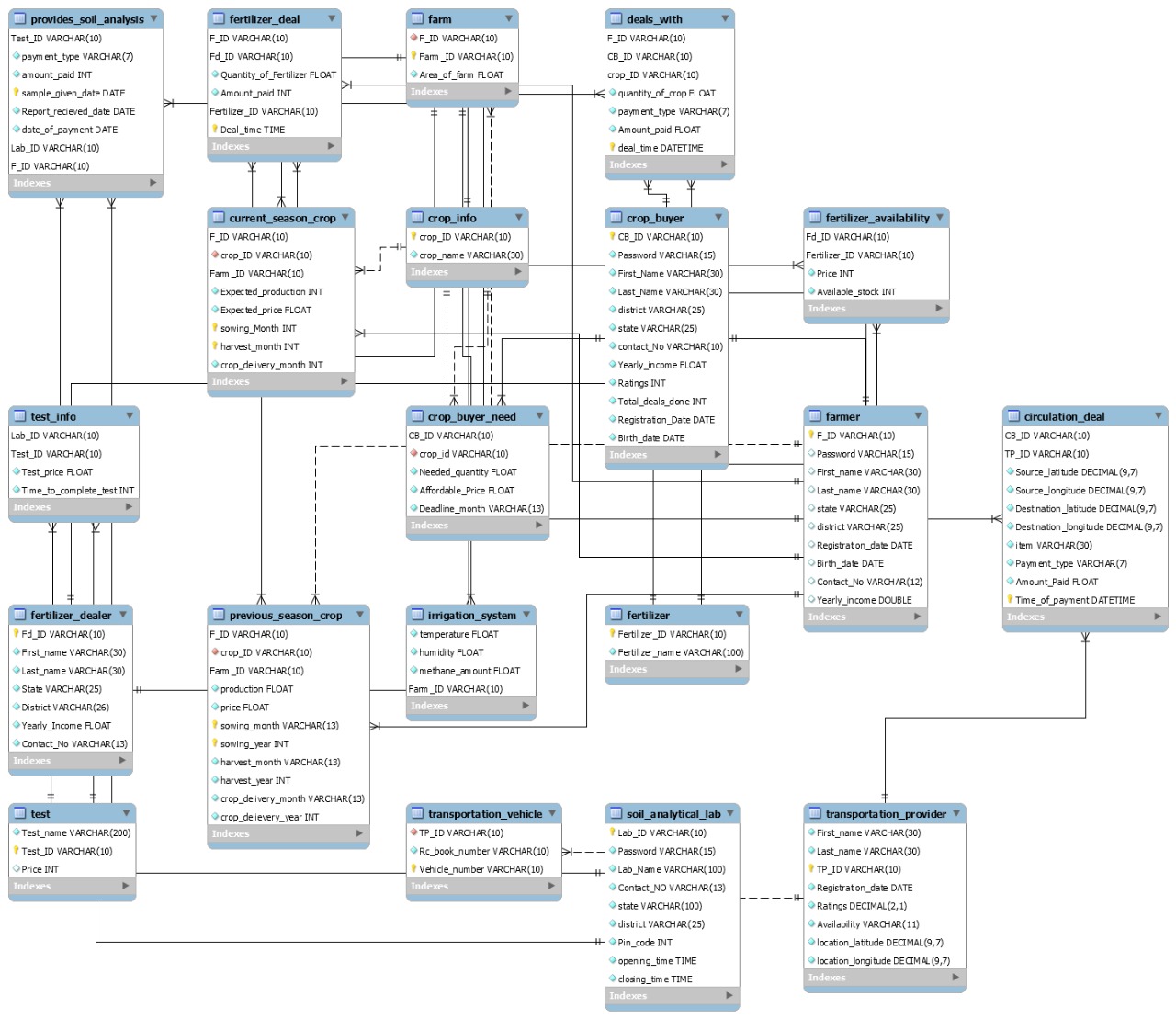
# RELATIONAL MODEL (DIAGRAM)

A basic relational model is made to get a look of how our system will look. Nothing is perfect and we will keep on improving our system always seeking the perfect.

(If the image is not clear we have uploaded it in a folder of relational model)



**Relational model (Updated)**



**Relational Model (final Updated)**

|  |  |
| --- | --- |
| Sr. No. | Tables |
| 1. | Table name: crop\_buyer  Primary key: CB\_ID  Candidate key: CB\_ID  Foreign key: No key  Super key:   1. CB\_ID , Password 2. CB\_ID , First\_Name 3. CB\_ID , Last\_Name 4. CB\_ID , district 5. CB\_ID , state 6. CB\_ID , contact\_No 7. CB\_ID , Yearly\_income 8. CB\_ID , Ratings 9. CB\_ID , Total\_deals\_done 10. CB\_ID , Registration\_Date 11. CB\_ID , Birth\_date 12. CB\_ID , Password , First\_Name 13. CB\_ID , Password , Last\_Name 14. CB\_ID , Password , district 15. CB\_ID , Password , state 16. CB\_ID , Password , contact\_No 17. CB\_ID , Password , Yearly\_income 18. CB\_ID , Password , Ratings 19. CB\_ID , Password , Total\_deals\_done 20. CB\_ID , Password , Registration\_Date 21. CB\_ID , Password , Birth\_date 22. CB\_ID , Password , First\_Name , Last\_Name 23. CB\_ID , Password , First\_Name , district 24. CB\_ID , Password , First\_Name , state 25. CB\_ID , Password , First\_Name , contact\_No 26. CB\_ID , Password , First\_Name , Yearly\_income 27. CB\_ID , Password , First\_Name , Ratings 28. CB\_ID , Password , First\_Name , Total\_deals\_done 29. CB\_ID , Password , First\_Name , Registration\_Date 30. CB\_ID , Password , First\_Name , Birth\_date 31. CB\_ID , Password , First\_Name , Last\_Name , district 32. CB\_ID , Password , First\_Name , Last\_Name , state 33. CB\_ID , Password , First\_Name , Last\_Name , contact\_No 34. CB\_ID , Password , First\_Name , Last\_Name , Yearly\_income 35. CB\_ID , Password , First\_Name , Last\_Name , Ratings 36. CB\_ID , Password , First\_Name , Last\_Name , Total\_deals\_done 37. CB\_ID , Password , First\_Name , Last\_Name , Registration\_Date 38. CB\_ID , Password , First\_Name , Last\_Name , Birth\_date 39. CB\_ID , Password , First\_Name , Last\_Name , district , state 40. CB\_ID , Password , First\_Name , Last\_Name , district , contact\_No 41. CB\_ID , Password , First\_Name , Last\_Name , district , Yearly\_income 42. CB\_ID , Password , First\_Name , Last\_Name , district , Ratings 43. CB\_ID , Password , First\_Name , Last\_Name , district , Total\_deals\_done 44. CB\_ID , Password , First\_Name , Last\_Name , district , Registration\_Date 45. CB\_ID , Password , First\_Name , Last\_Name , district , Birth\_date 46. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No 47. CB\_ID , Password , First\_Name , Last\_Name , district , state , Yearly\_income 48. CB\_ID , Password , First\_Name , Last\_Name , district , state , Ratings 49. CB\_ID , Password , First\_Name , Last\_Name , district , state , Total\_deals\_done 50. CB\_ID , Password , First\_Name , Last\_Name , district , state , Registration\_Date 51. CB\_ID , Password , First\_Name , Last\_Name , district , state , Birth\_date 52. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income 53. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Ratings 54. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Total\_deals\_done 55. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Registration\_Date 56. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Birth\_date 57. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Ratings 58. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Total\_deals\_done 59. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Registration\_Date 60. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Birth\_date 61. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Ratings , Total\_deals\_done 62. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Ratings , Registration\_Date 63. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Ratings , Birth\_date 64. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Ratings , Total\_deals\_done , Registration\_Date 65. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Ratings , Total\_deals\_done , Birth\_date 66. CB\_ID , Password , First\_Name , Last\_Name , district , state , contact\_No , Yearly\_income , Ratings , Total\_deals\_done , Registration\_Date , Birth\_date |
| 2. | Table name: crop\_buyer\_need  Primary key: CB\_ID  Candidate key: CB\_ID  Foreign key:   1. CB\_ID 2. crop\_id   Super key:   1. CB\_ID , crop\_id 2. CB\_ID , Needed\_quantity 3. CB\_ID , Affordable\_Price 4. CB\_ID , Deadline\_month 5. CB\_ID , crop\_id , Needed\_quantity 6. CB\_ID , crop\_id , Affordable\_Price 7. CB\_ID , crop\_id , Deadline\_month 8. CB\_ID , crop\_id , Needed\_quantity , Affordable\_Price 9. CB\_ID , crop\_id , Needed\_quantity , Deadline\_month 10. CB\_ID , crop\_id , Needed\_quantity , Affordable\_Price , Deadline\_month |
| 3. | Table name: crop\_info  Primary key: crop\_ID  Candidate key: crop\_ID  Foreign key: No key  Super key:   1. crop\_ID , crop\_name |
| 4. | Table name: current\_season\_crop  Primary key: F\_ID , Farm\_ID , sowing\_Month , harvest\_month  Candidate key: F\_ID , Farm\_ID , sowing\_Month , harvest\_month  Foreign key:   1. F\_ID 2. crop\_ID 3. Farm\_ID   Super key:   1. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , crop\_ID 2. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , Expected\_production 3. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , Expected\_price 4. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , crop\_delivery\_month 5. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , crop\_ID , Expected\_production 6. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , crop\_ID , Expected\_price 7. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , crop\_ID , crop\_delivery\_month 8. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , crop\_ID , Expected\_production , Expected\_price 9. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , crop\_ID , Expected\_production , crop\_delivery\_month 10. F\_ID , Farm\_ID , sowing\_Month , harvest\_month , crop\_ID , Expected\_production , Expected\_price , crop\_delivery\_month |
| 5. | Table name: farm  Primary key: Farm\_ID  Candidate key: Farm\_ID  Foreign key:   1. F\_ID   Super key:   1. Farm\_ID , F\_ID 2. Farm\_ID , Area\_of\_farm 3. Farm\_ID , F\_ID , Area\_of\_farm |
| 6. | Table name: provides\_soil\_analysis  Primary key: Fertilizer\_ID  Candidate key: Fertilizer\_ID  Foreign key: No key  Super key:   1. Fertilizer\_ID , Fertilizer\_name |
| 7. | Table name: provides\_soil\_analysis  Primary key: Lab\_ID , F\_ID , Test\_ID , sample\_given\_date  Candidate key: Lab\_ID , F\_ID , Test\_ID , sample\_given\_date  Foreign key:   1. Lab\_ID 2. Test\_ID 3. F\_ID   Super key:   1. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date, payment\_type 2. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date , amount\_paid 3. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date , Report\_recieved\_date 4. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date , date\_of\_payment 5. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date, payment\_type , amount\_paid 6. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date, payment\_type , Report\_recieved\_date 7. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date, payment\_type , date\_of\_paymen 8. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date, payment\_type , amount\_paid , Report\_recieved\_date 9. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date, payment\_type , amount\_paid , date\_of\_payment 10. Lab\_ID , F\_ID , Test\_ID , sample\_given\_date, payment\_type , amount\_paid , Report\_recieved\_date , date\_of\_payment |
| 8. | Table name: soil\_analytical\_lab  Primary key: Lab\_ID  Candidate key: Lab\_ID  Foreign key: No key  Super key:   1. Lab\_ID , Password 2. Lab\_ID , Lab\_Name 3. Lab\_ID , Contact\_NO 4. Lab\_ID , state 5. Lab\_ID , district 6. Lab\_ID , Pin\_code 7. Lab\_ID , opening\_time 8. Lab\_ID , closing\_time 9. Lab\_ID , Password , Lab\_Name 10. Lab\_ID , Password , Contact\_NO 11. Lab\_ID , Password , state 12. Lab\_ID , Password , district 13. Lab\_ID , Password , Pin\_code 14. Lab\_ID , Password , opening\_time 15. Lab\_ID , Password , closing\_time 16. Lab\_ID , Password , Lab\_Name , Contact\_NO 17. Lab\_ID , Password , Lab\_Name , state 18. Lab\_ID , Password , Lab\_Name , district 19. Lab\_ID , Password , Lab\_Name , Pin\_code 20. Lab\_ID , Password , Lab\_Name , opening\_time 21. Lab\_ID , Password , Lab\_Name , closing\_time 22. Lab\_ID , Password , Lab\_Name , Contact\_NO , state 23. Lab\_ID , Password , Lab\_Name , Contact\_NO , district 24. Lab\_ID , Password , Lab\_Name , Contact\_NO , Pin\_code 25. Lab\_ID , Password , Lab\_Name , Contact\_NO , opening\_time 26. Lab\_ID , Password , Lab\_Name , Contact\_NO , closing\_time 27. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , district 28. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , Pin\_code 29. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , opening\_time 30. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , closing\_time 31. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , district , Pin\_code 32. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , district , opening\_time 33. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , district , closing\_time 34. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , district , Pin\_code , opening\_time 35. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , district , Pin\_code , closing\_time 36. Lab\_ID , Password , Lab\_Name , Contact\_NO , state , district , Pin\_code , opening\_time , closing\_time |
| 9. | Table name: test  Primary key: Test\_ID  Candidate key: Test\_ID  Foreign key: No key  Super key:   1. Test\_ID , Test\_name 2. Test\_ID , Price 3. Test\_ID , Test\_name , Price |
| 10. | Table name: test\_info  Primary key: Lab\_ID , Test\_ID  Candidate key: Lab\_ID , Test\_ID  Foreign key:  1. Lab\_ID  2. Test\_ID  Super key:  1. Lab\_ID , Test\_ID , Test\_price  2. Lab\_ID , Test\_ID , Time\_to\_complete\_test  3. Lab\_ID , Test\_ID , Test\_price , Time\_to\_complete\_test |
| 11. | Table name: transportation\_vehicle  Primary key: Vehicle\_number  Candidate key: Vehicle\_number  Foreign key: TP\_ID  Super key:   1. Vehicle\_number , TP\_ID 2. Vehicle\_number , Rc\_book\_number 3. Vehicle\_number , TP\_ID , Rc\_book\_number |
| 12 | Table name: fertilizer  Primary key: Fertilizer\_ID  Candidate key: Fertilizer\_ID  Foreign key: no key  Super key: Fertilizer\_ID, fertilizer\_name |
| 13 | Table name: fertilizer\_deal  Primary key: F\_ID ,Fd\_ID Fertilizer\_ID, Deal\_tim  Candidate key: F\_ID ,Fd\_ID Fertilizer\_ID, Deal\_tim  Foreign key:   1. Fd\_ID 2. Fertilizer\_ID 3. F\_ID   Super key: F\_ID ,Fd\_ID Fertilizer\_ID, Deal\_time, Price |
| 14 | Table name: fertilizer\_availability  Primary key: Fd\_ID, Fertilizer\_ID  Candidate key: Fd\_ID, Fertilizer\_ID  Foreign key: Fd\_ID,  Fertilizer\_ID  Super key; Fd\_ID, Fertilizer\_ID, price |
| 15 | Table name: fertilizer\_dealer  Primary key: Fd\_ID  Candidate key: Fd\_ID  Foreign key: no key  Super key:   1. Fd\_ID, First\_name 2. Fd\_ID, First\_name, Last\_name 3. Fd\_ID, First\_name, Last\_name 4. Fd\_ID, First\_name, Last\_name, State 5. Fd\_ID, First\_name, Last\_name, State, District 6. Fd\_ID, First\_name, Last\_name, State, District, Yearly\_Income 7. Fd\_ID, First\_name, Last\_name, State, District, Yearly\_Income, Contact\_No |
| 16 | Table name: irrigation\_system  Primary key: Farm\_ID  Candidate key: Farm\_ID  Foreign key: no key  Super key:   1. Farm\_ID, temperature 2. Farm\_ID, temperature, humidity 3. Farm\_ID, temperature, humidity, methane\_amount 4. Farm\_ID, temperature, humidity, methane\_amount, Timer |
| 17 | Table name: deals\_with  Primary key: F\_ID, CB\_ID , Crop\_ID, deal\_time  Candidate key: F\_ID, CB\_ID , Crop\_ID, deal\_time  Foreign key:   1. F\_ID 2. CB\_ID 3. Crop\_ID     Super key:   1. F\_ID, CB\_ID , Crop\_ID, deal\_time, quantity\_of\_crop 2. F\_ID, CB\_ID , Crop\_ID, deal\_time, quantity\_of\_crop, payment\_type 3. F\_ID, CB\_ID , Crop\_ID, deal\_time, quantity\_of\_crop, payment\_type, Amount\_paid |
| 18 | Table name: farmer  Primary key: F\_ID  Candidate key: F\_ID  Foreign key: no key  Super key:   1. F\_ID, First\_name 2. F\_ID, First\_name, Last\_name 3. F\_ID, First\_name, Last\_name, Adhaar\_card 4. F\_ID, First\_name, Last\_name, Adhaar\_card, leased\_land\_area 5. F\_ID, First\_name, Last\_name, Adhaar\_card, leased\_land\_area, Loan\_on\_head 6. F\_ID, First\_name, Last\_name, Adhaar\_card, leased\_land\_area, Loan\_on\_head , Yearly\_income 7. F\_ID, First\_name, Last\_name, Adhaar\_card, State leased\_land\_area, Loan\_on\_head , Yearly\_income, District 8. F\_ID, First\_name, Last\_name, Adhaar\_card, leased\_land\_area, Loan\_on\_head , Yearly\_income, State, District, Address 9. F\_ID, First\_name, Last\_name, Adhaar\_card, leased\_land\_area, Loan\_on\_head , Yearly\_income, State, District, Address, Registration\_date 10. F\_ID, First\_name, Last\_name, Adhaar\_card, leased\_land\_area, Loan\_on\_head , Yearly\_income, State, District, Address, Registration\_date, Birth\_date 11. F\_ID, First\_name, Last\_name, Adhaar\_card, leased\_land\_area, Loan\_on\_head , Yearly\_income, State, District, Address, Registration\_date, Birth\_date , Contact\_No |
| 19 | Table name: current\_season\_crop  Primary key: F\_ID,Farm\_ID,sowing\_Month ,harvest\_month  Candidate key: F\_ID,Farm\_ID,sowing\_Month ,harvest\_month  Foreign key:   1. F\_ID 2. `Farm\_ID 3. crop\_ID   Super key:   1. F\_ID,Farm\_ID,sowing\_Month ,harvest\_month, Expected\_production 2. F\_ID,Farm\_ID,sowing\_Month ,harvest\_month, Expected\_production , Expected\_price 3. F\_ID,Farm\_ID,sowing\_Month ,harvest\_month, Expected\_production, Expected\_price , sowing\_year 4. F\_ID,Farm\_ID,sowing\_Month ,harvest\_month, Expected\_production , Expected\_price , sowing\_year, harvest\_year 5. F\_ID,Farm\_ID,sowing\_Month ,harvest\_month, Expected\_production, Expected\_price, sowing\_year, harvest\_year, crop\_delivery\_month |
| 20 | Table name: circulation\_deal  Primary key: CB\_ID, TP\_ID, Time\_of\_payment  Candidate key: CB\_ID, TP\_ID, Time\_of\_payment  Foreign key:  Super key:   1. CB\_ID, TP\_ID, Time\_of\_payment , Source 2. CB\_ID, TP\_ID, Time\_of\_payment , Source, Destination 3. CB\_ID, TP\_ID, Time\_of\_payment, Source, Destination, item 4. CB\_ID, TP\_ID, Time\_of\_payment, Source, Destination, item, Payment\_type 5. CB\_ID, TP\_ID, Time\_of\_payment, Source, Destination, item, Payment\_type, Amount\_Paid |
| 21 | Table name: transportation\_provider  Primary key: TP\_ID  Candidate key: TP\_ID  Foreign key:  Super key:   1. TP\_ID, First\_name 2. TP\_ID, First\_name. Last\_name 3. TP\_ID, First\_name. Last\_name, Adhhar\_card 4. TP\_ID, First\_name. Last\_name, Adhhar\_card, Availability 5. TP\_ID, First\_name. Last\_name, Adhhar\_card, Availability, Current\_Location 6. TP\_ID, First\_name. Last\_name, Adhhar\_card , Availability, Current\_Location, Licence |

# Relational Algebra

1. This query returns Crop buyers(CB\_ID) having maximum ratings and expecting the crop that farmer sow in current season.

* ΠCB\_ID[ΠCB\_ID, Ratings(crop\_buyer) ⨯ ΠCB\_ID, Ratings(crop\_buyer)

– σRatings<Ratings2(ΠCB\_ID, Ratings(crop\_buyer)

⨯ ρRatings2/Ratings(ΠCB\_ID, Ratings(crop\_buyer)))]

∩ ΠCB\_ID[ΠCB\_ID,crop\_ID(crop\_buyer\_need)

⨝crop\_buyer\_need.crop\_ID = current\_season\_crop.crop\_ID

Πcrop\_ID(current\_season\_crop)]

1. This query returns Labs(Lab\_ID, Contact\_NO), When a farmer searches labs residing in the same city by Test\_name, Test\_price.

* ΠLab\_ID,Contact\_NO [ ΠLab\_ID,Contact\_NO,Pin\_code (soil\_analytical\_lab)

⨝soil\_analytical\_lab.Pin\_code = farmer.pincode ΠF\_ID,pincode (farmer) ]

⨝ΠLab\_ID [ ΠLab\_ID,Test\_ID (test\_info)

⨝ Π ( ΠTest\_ID (σTest\_name=”name”(test))

∩ ΠTest\_ID ( σTest\_price<=”price” (test\_info)) )]

1. This query returns Farmers’ details (F\_ID, First\_name, Contact\_No, district, state) whose current season’s crop production is higher than previous season’s crop production for the same crop.

* σstate=”Gujarat” [ ΠF\_ID,First\_name,Contact\_No,district,state (farmer)

⨝ σcurrent\_season\_crop.Expected\_production>>previous\_season\_crop.production

( ΠF\_ID,crop\_ID,Expected\_production (current\_season\_crop)

⨝ ΠF\_ID,crop\_ID,production (previous\_season\_crop) )

⨝ Πcrop\_ID,crop\_name (crop\_info) ]

1. This query returns Fertilizer dealers’ details (Fd\_ID, First\_name, Contact\_No) whose shop’s pincode is 382007 and who has Urea fertilizer with available stock greater than or equal to 5.

* ΠFd\_ID,First\_name,Contact\_No [

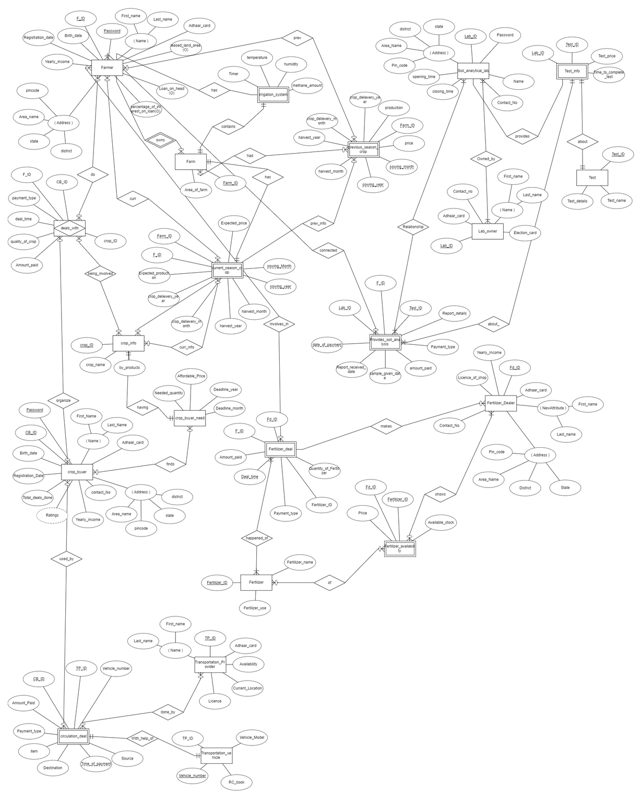
σPin\_code=382007 ( ΠFd\_ID,First\_name,Pin\_code,Contact\_No (fertilizer\_dealer)

⨝ σAvailable\_stock>=5( ΠFertilizer\_ID (σFertilizer\_name=”Urea” (fertilizer))

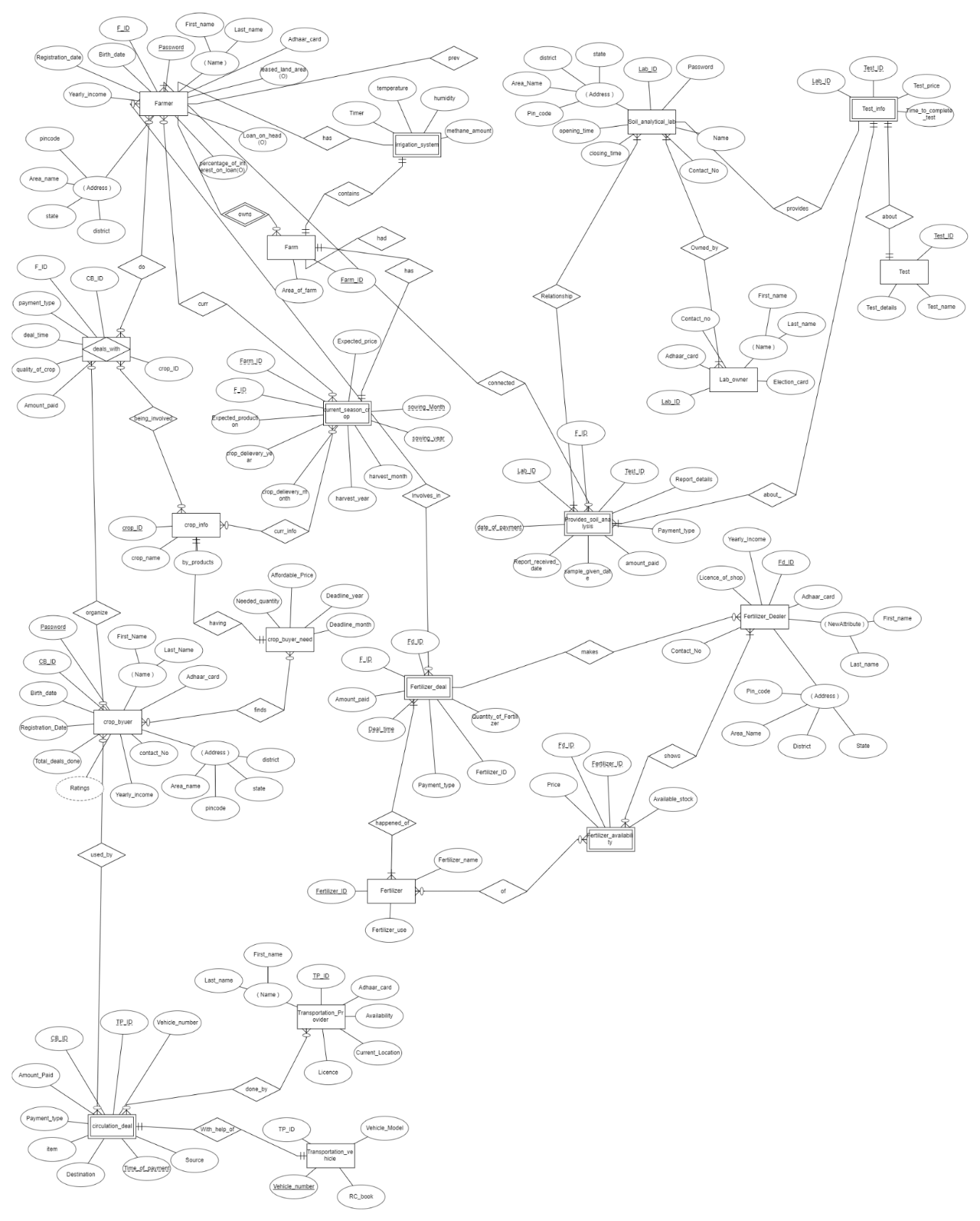
⨝ (fertilizer\_availability) ))

]

# ER Model



**Er diagram (first created diagram)**

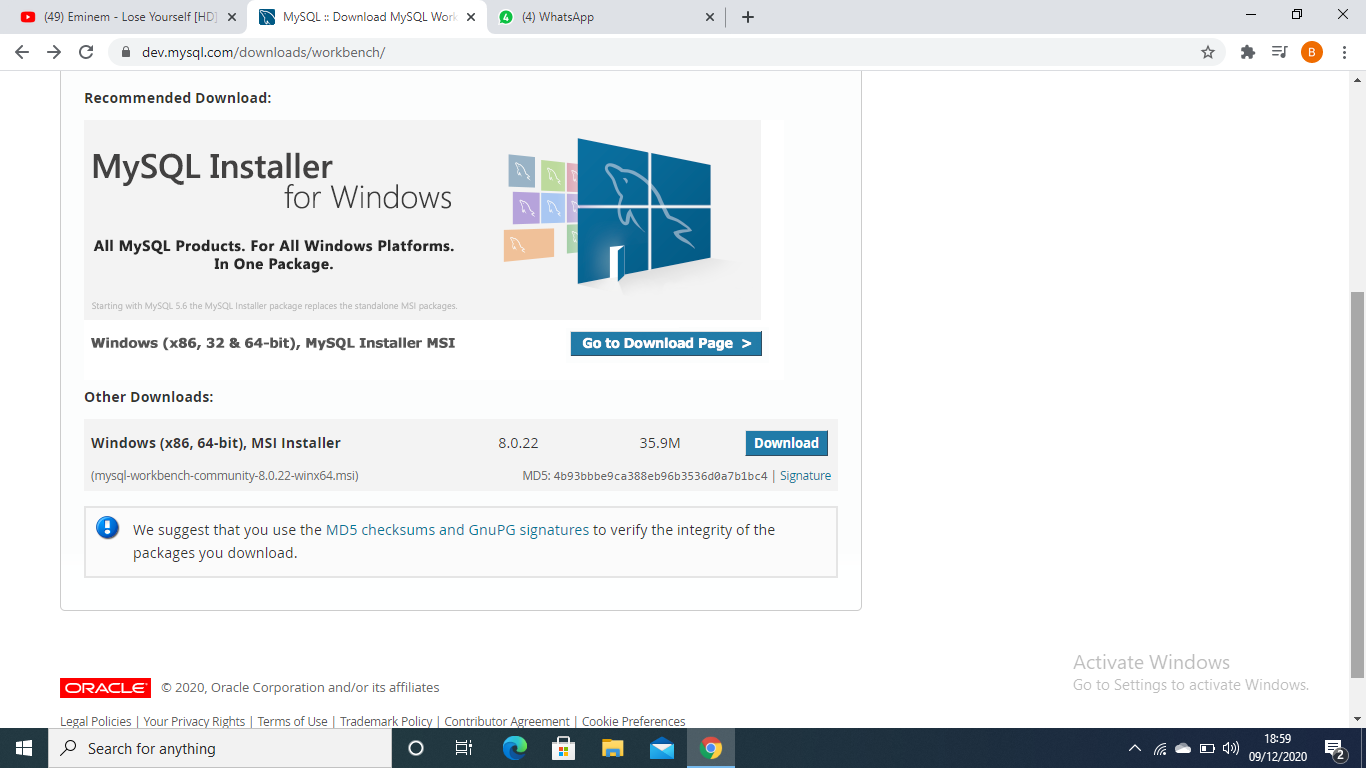


**Er diagram final updated**

# Installation of MYSQL workbench

* To make the database of our project and to do task given/assigned by faculty related to SQL queries, All the team members installed MYSQL workbench.
* Throughout the project we have run our queries on MYSQL workbench when needed and during the lab experiments also, we have used MYSQL workbench.
* We have downloaded MYSQL workbench from <https://dev.mysql.com/downloads/workbench/>.

Screenshot:



# Create table syntax

DROP DATABASE IF EXISTS `agriculture`; CREATE DATABASE IF NOT EXISTS `agriculture`;

USE `agriculture`;

🡪Create Table command for Table named ‘Farmer’

CREATE TABLE `farmer` (

`F\_ID` varchar(10) NOT NULL,

`Password` varchar(15) DEFAULT NULL,

`First\_name` varchar(30) DEFAULT NULL,

`Last\_name` varchar(30) DEFAULT NULL,

`state` varchar(25) DEFAULT NULL,

`district` varchar(25) DEFAULT NULL,

`Registration\_date` date DEFAULT NULL,

`Birth\_date` date DEFAULT NULL,

`Contact\_No` varchar(12) DEFAULT NULL,

`Yearly\_income` double DEFAULT NULL,

PRIMARY KEY (`F\_ID`),

CONSTRAINT `farmer\_chk\_1` CHECK ((length(\_cp850'Password') > 6))

);

🡪Create Table command for Table named ‘Farmer’

CREATE TABLE `farm` (

`F\_ID` varchar(10) NOT NULL,

`Farm\_ID` varchar(10) NOT NULL,

`Area\_of\_farm` float NOT NULL,

PRIMARY KEY (`Farm\_ID`),

KEY `F\_ID` (`F\_ID`),

CONSTRAINT `farm\_ibfk\_1` FOREIGN KEY (`F\_ID`) REFERENCES `farmer` (`F\_ID`)

);

🡪Create Table command for Table named ‘crop\_info’

CREATE TABLE `crop\_info` (

`crop\_ID` varchar(10) NOT NULL,

`crop\_name` varchar(30) NOT NULL,

PRIMARY KEY (`crop\_ID`)

);

🡪Create Table command for Table named ‘current\_season\_crop’

CREATE TABLE `current\_season\_crop` (

`F\_ID` varchar(10) NOT NULL,

`crop\_ID` varchar(10) NOT NULL,

`Farm\_ID` varchar(10) NOT NULL,

`Expected\_production` int NOT NULL,

`Expected\_price` float NOT NULL,

`sowing\_Month` int NOT NULL,

`harvest\_month` int NOT NULL,

`crop\_delivery\_month` int NOT NULL,

PRIMARY KEY (`F\_ID`,`Farm\_ID`,`sowing\_Month`,`harvest\_month`),

KEY `crop\_ID` (`crop\_ID`),

KEY `Farm\_ID` (`Farm\_ID`),

CONSTRAINT `current\_season\_crop\_ibfk\_1` FOREIGN KEY (`F\_ID`) REFERENCES `farmer` (`F\_ID`),

CONSTRAINT `current\_season\_crop\_ibfk\_2` FOREIGN KEY (`crop\_ID`) REFERENCES `crop\_info` (`crop\_ID`),

CONSTRAINT `current\_season\_crop\_ibfk\_3` FOREIGN KEY (`Farm\_ID`) REFERENCES `farm` (`Farm\_ID`)

);

🡪Create Table command for Table named ‘crop\_info’

CREATE TABLE `circulation\_deal` (

`CB\_ID` varchar(10) NOT NULL,

`TP\_ID` varchar(10) NOT NULL,

`Source\_latitude` decimal(9,7) NOT NULL,

`Source\_longitude` decimal(9,7) NOT NULL,

`Destination\_latitude` decimal(9,7) NOT NULL,

`Destination\_longitude` decimal(9,7) NOT NULL,

`item` varchar(30) NOT NULL,

`Payment\_type` varchar(7) NOT NULL,

`Amount\_Paid` float NOT NULL,

`Time\_of\_payment` datetime NOT NULL,

PRIMARY KEY (`CB\_ID`,`TP\_ID`,`Time\_of\_payment`),

KEY `TP\_ID` (`TP\_ID`),

CONSTRAINT `circulation\_deal\_ibfk\_1` FOREIGN KEY (`CB\_ID`) REFERENCES `crop\_buyer` (`CB\_ID`),

CONSTRAINT `circulation\_deal\_ibfk\_2` FOREIGN KEY (`TP\_ID`) REFERENCES `transportation\_provider` (`TP\_ID`)

);

🡪Create Table command for Table named ‘crop\_info’

CREATE TABLE `crop\_buyer` (

`CB\_ID` varchar(10) NOT NULL,

`Password` varchar(15) NOT NULL,

`First\_Name` varchar(30) NOT NULL,

`Last\_Name` varchar(30) NOT NULL,

`district` varchar(25) NOT NULL,

`state` varchar(25) NOT NULL,

`contact\_No` varchar(10) NOT NULL,

`Yearly\_income` float NOT NULL,

`Ratings` int NOT NULL,

`Total\_deals\_done` int NOT NULL,

`Registration\_Date` date NOT NULL,

`Birth\_date` date NOT NULL,

PRIMARY KEY (`CB\_ID`),

CONSTRAINT `crop\_buyer\_chk\_1` CHECK ((length(\_cp850'Password') > 6))

);

🡪Create Table command for Table named ‘crop\_buyer\_need’

CREATE TABLE `crop\_buyer\_need` (

`CB\_ID` varchar(10) NOT NULL,

`crop\_id` varchar(10) NOT NULL,

`Needed\_quantity` float NOT NULL,

`Affordable\_Price` float NOT NULL,

`Deadline\_month` varchar(13) NOT NULL,

PRIMARY KEY (`CB\_ID`),

KEY `crop\_id` (`crop\_id`),

CONSTRAINT `crop\_buyer\_need\_ibfk\_1` FOREIGN KEY (`CB\_ID`) REFERENCES `crop\_buyer` (`CB\_ID`),

CONSTRAINT `crop\_buyer\_need\_ibfk\_2` FOREIGN KEY (`crop\_id`) REFERENCES `crop\_info` (`crop\_ID`)

);

🡪Create Table command for Table named ‘deals\_with’

CREATE TABLE `deals\_with` (

`F\_ID` varchar(10) NOT NULL,

`CB\_ID` varchar(10) NOT NULL,

`crop\_ID` varchar(10) NOT NULL,

`quantity\_of\_crop` float NOT NULL,

`payment\_type` varchar(7) NOT NULL,

`Amount\_paid` float NOT NULL,

`deal\_time` datetime NOT NULL,

PRIMARY KEY (`F\_ID`,`CB\_ID`,`crop\_ID`,`deal\_time`),

KEY `crop\_ID` (`crop\_ID`),

KEY `CB\_ID` (`CB\_ID`),

CONSTRAINT `deals\_with\_ibfk\_1` FOREIGN KEY (`F\_ID`) REFERENCES `farmer` (`F\_ID`),

CONSTRAINT `deals\_with\_ibfk\_2` FOREIGN KEY (`crop\_ID`) REFERENCES `crop\_info` (`crop\_ID`),

CONSTRAINT `deals\_with\_ibfk\_3` FOREIGN KEY (`CB\_ID`) REFERENCES `crop\_buyer` (`CB\_ID`)

);

🡪Create Table command for Table named ‘fertilizer’

CREATE TABLE `fertilizer` (

`Fertilizer\_ID` varchar(10) NOT NULL,

`Fertilizer\_name` varchar(100) NOT NULL,

PRIMARY KEY (`Fertilizer\_ID`)

);

🡪Create Table command for Table named ‘fertilizer\_availability’

CREATE TABLE `fertilizer\_availability` (

`Fd\_ID` varchar(10) NOT NULL,

`Fertilizer\_ID` varchar(10) NOT NULL,

`Price` int NOT NULL,

`Available\_stock` int NOT NULL,

PRIMARY KEY (`Fd\_ID`,`Fertilizer\_ID`),

KEY `Fertilizer\_ID` (`Fertilizer\_ID`),

CONSTRAINT `fertilizer\_availability\_ibfk\_1` FOREIGN KEY (`Fd\_ID`) REFERENCES `fertilizer\_dealer` (`Fd\_ID`),

CONSTRAINT `fertilizer\_availability\_ibfk\_2` FOREIGN KEY (`Fertilizer\_ID`) REFERENCES `fertilizer` (`Fertilizer\_ID`)

);

🡪Create Table command for Table named ‘fertilizer\_deal’

CREATE TABLE `fertilizer\_deal` (

`F\_ID` varchar(10) NOT NULL,

`Fd\_ID` varchar(10) NOT NULL,

`Quantity\_of\_Fertilizer` float NOT NULL,

`Amount\_paid` int NOT NULL,

`Fertilizer\_ID` varchar(10) NOT NULL,

`Deal\_time` time NOT NULL,

PRIMARY KEY (`F\_ID`,`Fd\_ID`,`Fertilizer\_ID`,`Deal\_time`),

KEY `Fertilizer\_ID` (`Fertilizer\_ID`),

KEY `Fd\_ID` (`Fd\_ID`),

CONSTRAINT `fertilizer\_deal\_ibfk\_1` FOREIGN KEY (`Fertilizer\_ID`) REFERENCES `fertilizer` (`Fertilizer\_ID`),

CONSTRAINT `fertilizer\_deal\_ibfk\_2` FOREIGN KEY (`F\_ID`) REFERENCES `farmer` (`F\_ID`),

CONSTRAINT `fertilizer\_deal\_ibfk\_3` FOREIGN KEY (`Fd\_ID`) REFERENCES `fertilizer\_dealer` (`Fd\_ID`)

);

🡪Create Table command for Table named ‘fertilizer\_dealer’

CREATE TABLE `fertilizer\_dealer` (

`Fd\_ID` varchar(10) NOT NULL,

`First\_name` varchar(30) NOT NULL,

`Last\_name` varchar(30) NOT NULL,

`State` varchar(25) NOT NULL,

`District` varchar(26) NOT NULL,

`Yearly\_Income` float NOT NULL,

`Contact\_No` varchar(13) NOT NULL,

PRIMARY KEY (`Fd\_ID`)

);

🡪Create Table command for Table named ‘irrigation\_system’

CREATE TABLE `irrigation\_system` (

`temperature` float NOT NULL,

`humidity` float NOT NULL,

`methane\_amount` float NOT NULL,

`Farm\_ID` varchar(10) NOT NULL,

PRIMARY KEY (`Farm\_ID`),

CONSTRAINT `irrigation\_system\_ibfk\_1` FOREIGN KEY (`Farm\_ID`) REFERENCES `farm` (`Farm\_ID`)

);

🡪Create Table command for Table named ‘provides\_soil\_analysis’

CREATE TABLE `provides\_soil\_analysis` (

`Test\_ID` varchar(10) NOT NULL,

`payment\_type` varchar(7) NOT NULL,

`amount\_paid` int NOT NULL,

`sample\_given\_date` date NOT NULL,

`Report\_recieved\_date` date NOT NULL,

`date\_of\_payment` date NOT NULL,

`Lab\_ID` varchar(10) NOT NULL,

`F\_ID` varchar(10) NOT NULL,

PRIMARY KEY (`Lab\_ID`,`F\_ID`,`Test\_ID`,`sample\_given\_date`),

KEY `Test\_ID` (`Test\_ID`),

KEY `F\_ID` (`F\_ID`),

CONSTRAINT `provides\_soil\_analysis\_ibfk\_1` FOREIGN KEY (`Lab\_ID`) REFERENCES `soil\_analytical\_lab` (`Lab\_ID`),

CONSTRAINT `provides\_soil\_analysis\_ibfk\_2` FOREIGN KEY (`Test\_ID`) REFERENCES `test` (`Test\_ID`),

CONSTRAINT `provides\_soil\_analysis\_ibfk\_3` FOREIGN KEY (`F\_ID`) REFERENCES `farmer` (`F\_ID`)

);

🡪Create Table command for Table named ‘soil\_analytical\_lab’

CREATE TABLE `soil\_analytical\_lab` (

`Lab\_ID` varchar(10) NOT NULL,

`Password` varchar(15) NOT NULL,

`Lab\_Name` varchar(100) NOT NULL,

`Contact\_NO` varchar(13) NOT NULL,

`state` varchar(100) NOT NULL,

`district` varchar(25) NOT NULL,

`Pin\_code` int NOT NULL,

`opening\_time` time NOT NULL,

`closing\_time` time NOT NULL,

PRIMARY KEY (`Lab\_ID`)

);

🡪Create Table command for Table named ‘test’

CREATE TABLE `test` (

`Test\_name` varchar(200) NOT NULL,

`Test\_ID` varchar(10) NOT NULL,

`Price` int DEFAULT NULL,

PRIMARY KEY (`Test\_ID`)

);

🡪Create Table command for Table named ‘test\_info’

CREATE TABLE `test\_info` (

`Lab\_ID` varchar(10) NOT NULL,

`Test\_ID` varchar(10) NOT NULL,

`Test\_price` float NOT NULL,

`Time\_to\_complete\_test` int NOT NULL,

PRIMARY KEY (`Lab\_ID`,`Test\_ID`),

KEY `Test\_ID` (`Test\_ID`),

CONSTRAINT `test\_info\_ibfk\_1` FOREIGN KEY (`Lab\_ID`) REFERENCES `soil\_analytical\_lab` (`Lab\_ID`),

CONSTRAINT `test\_info\_ibfk\_2` FOREIGN KEY (`Test\_ID`) REFERENCES `test` (`Test\_ID`)

);

🡪Create Table command for Table named ‘transportation\_provider’

CREATE TABLE `transportation\_provider` (

`First\_name` varchar(30) NOT NULL,

`Last\_name` varchar(30) NOT NULL,

`TP\_ID` varchar(10) NOT NULL,

`Registration\_date` date NOT NULL,

`Ratings` decimal(2,1) NOT NULL,

`Availability` varchar(11) NOT NULL,

`location\_latitude` decimal(9,7) NOT NULL,

`location\_longitude` decimal(9,7) NOT NULL,

PRIMARY KEY (`TP\_ID`)

);

🡪Create Table command for Table named ‘transportation\_vehicle’

CREATE TABLE `transportation\_vehicle` (

`TP\_ID` varchar(10) NOT NULL,

`Rc\_book\_number` varchar(10) NOT NULL,

`Vehicle\_number` varchar(10) NOT NULL,

PRIMARY KEY (`Vehicle\_number`),

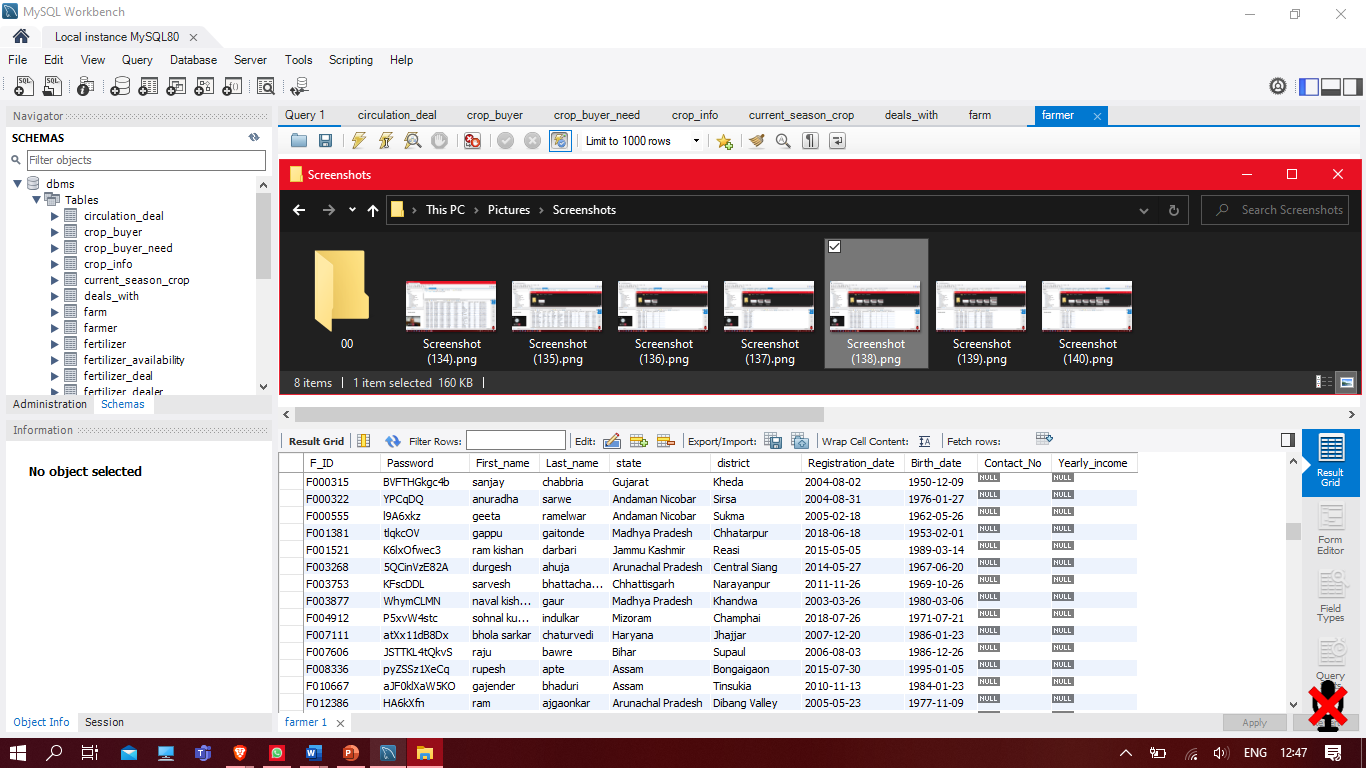
KEY `TP\_ID` (`TP\_ID`),

CONSTRAINT `transportation\_vehicle\_ibfk\_1` FOREIGN KEY (`TP\_ID`) REFERENCES `transportation\_provider` (`TP\_ID`)

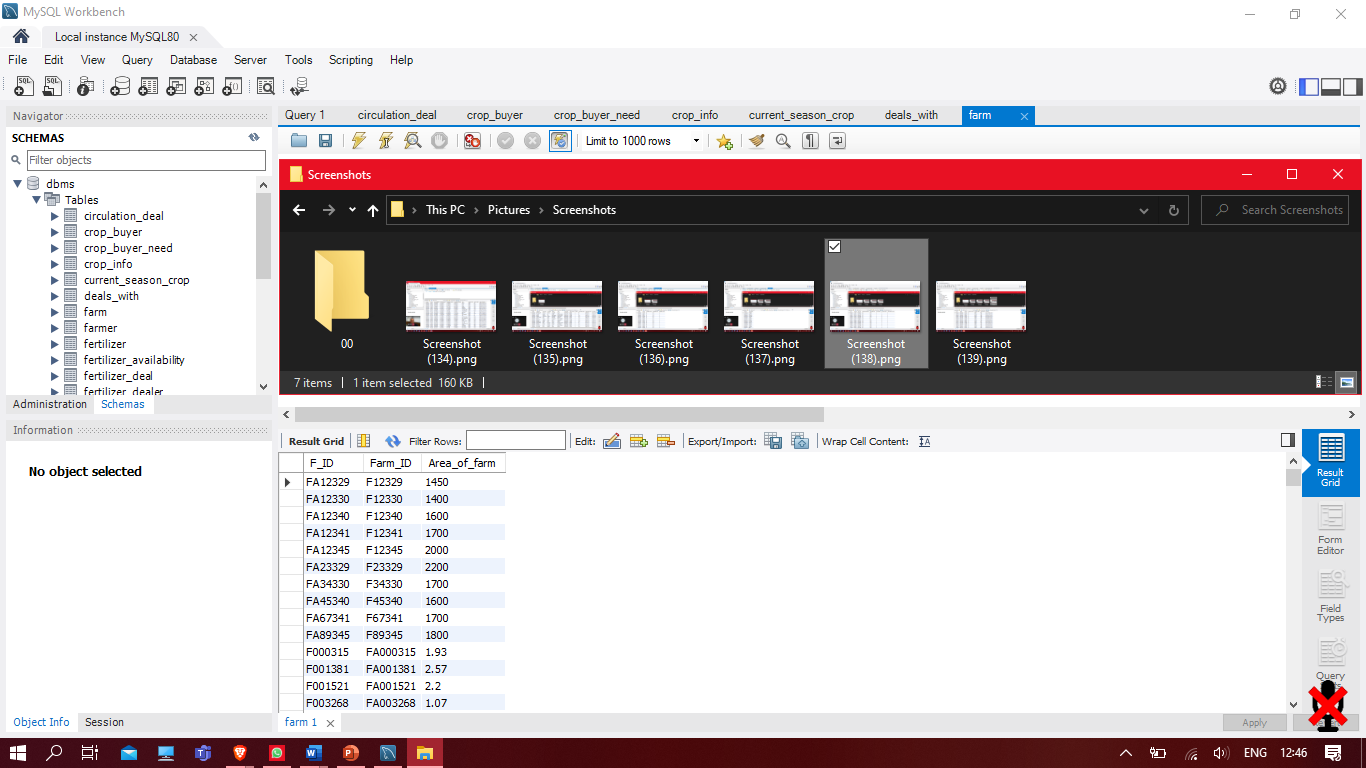
);

# Filled Tables

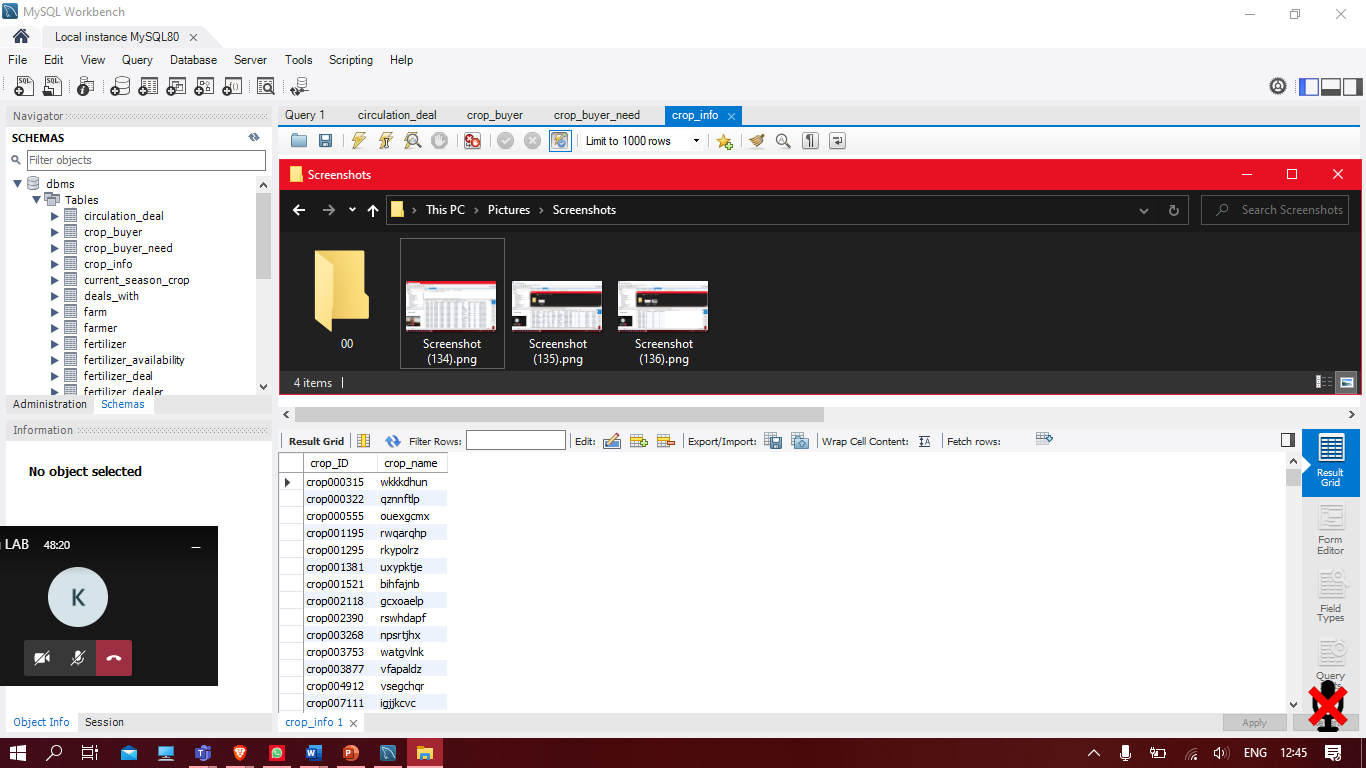
1. **farmer**



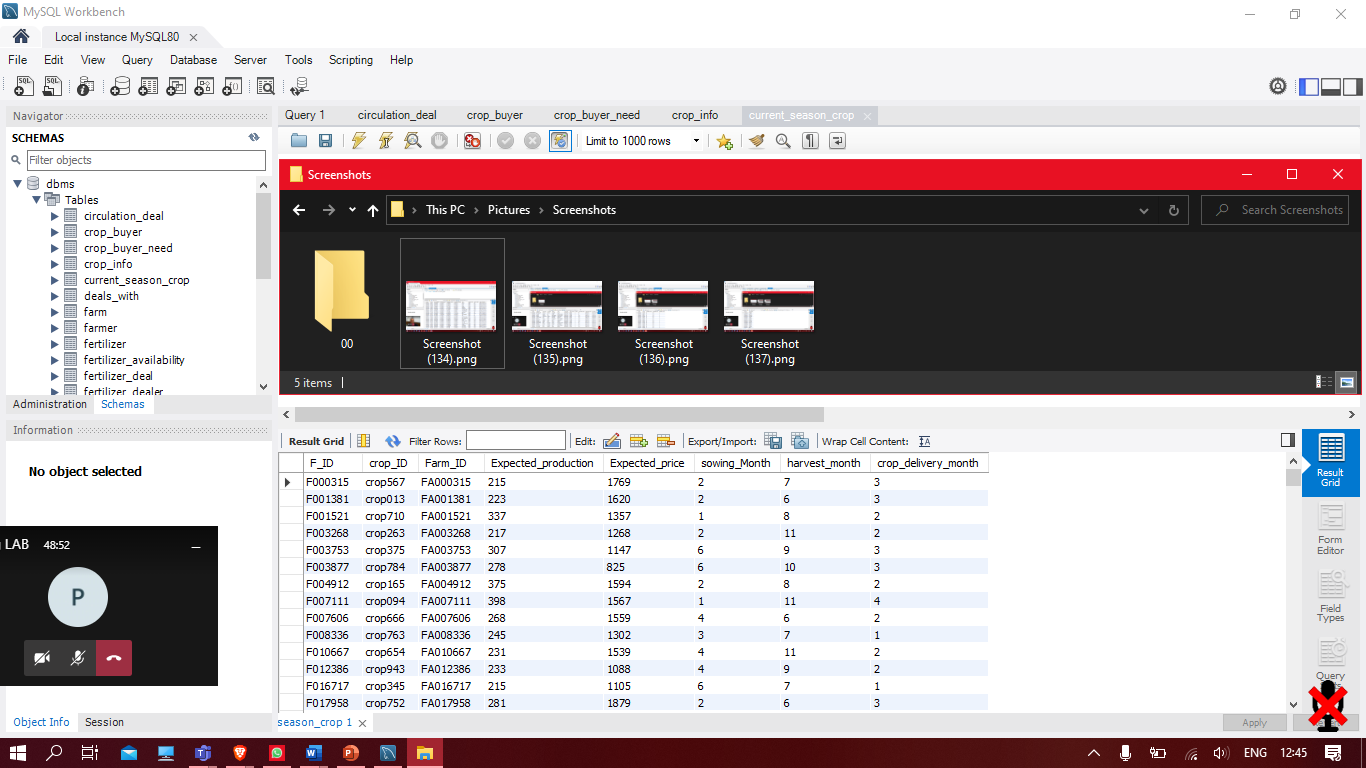
1. **farm**



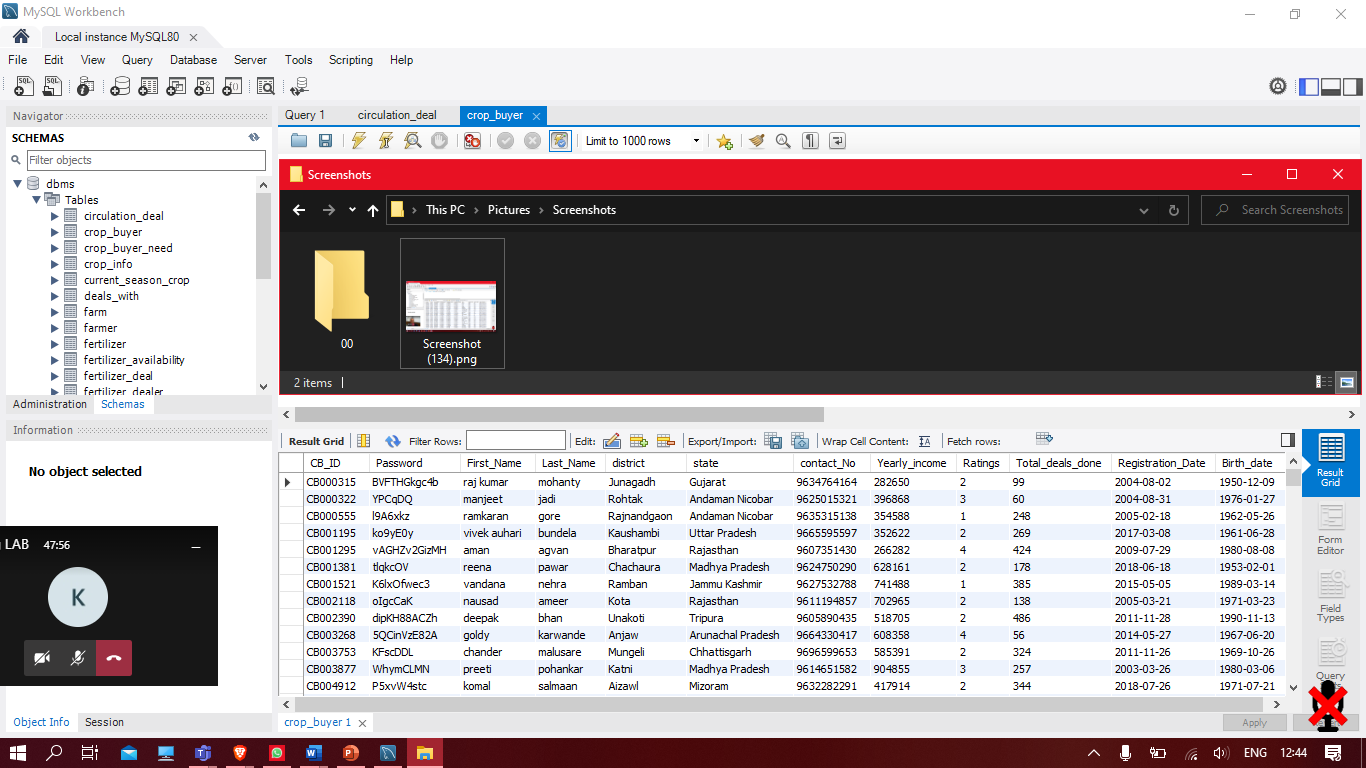
1. **crop\_info**



1. **current\_season\_crop**



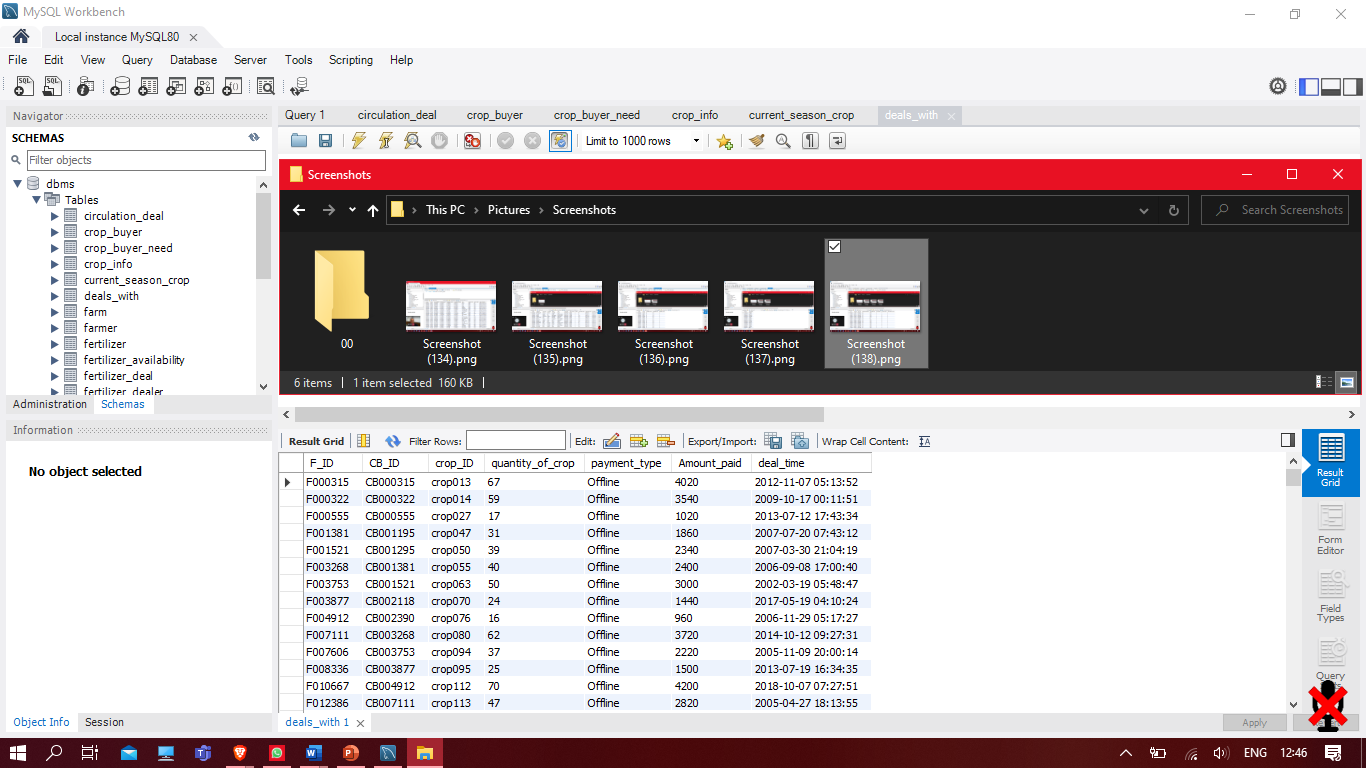
1. **crop\_buyer**



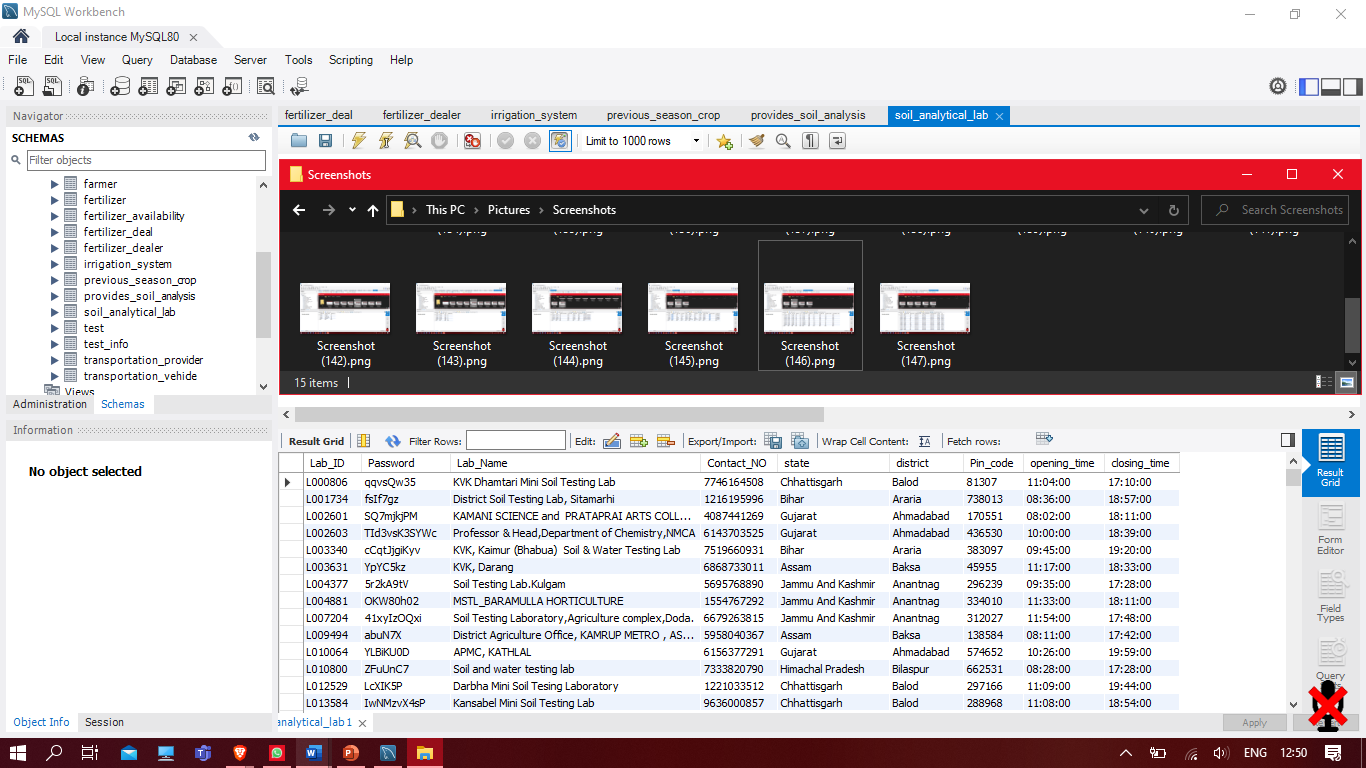
1. **crop\_buyer\_need**



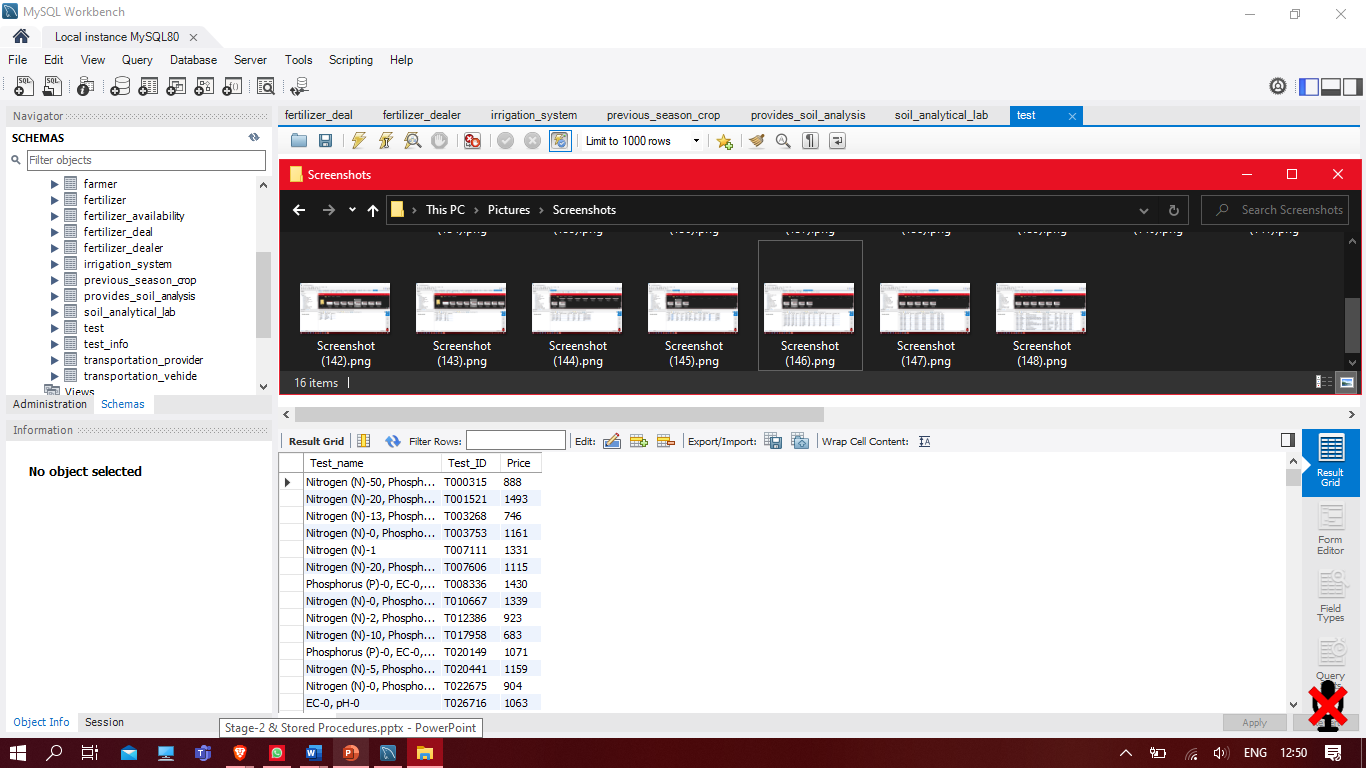
1. **deals\_with**



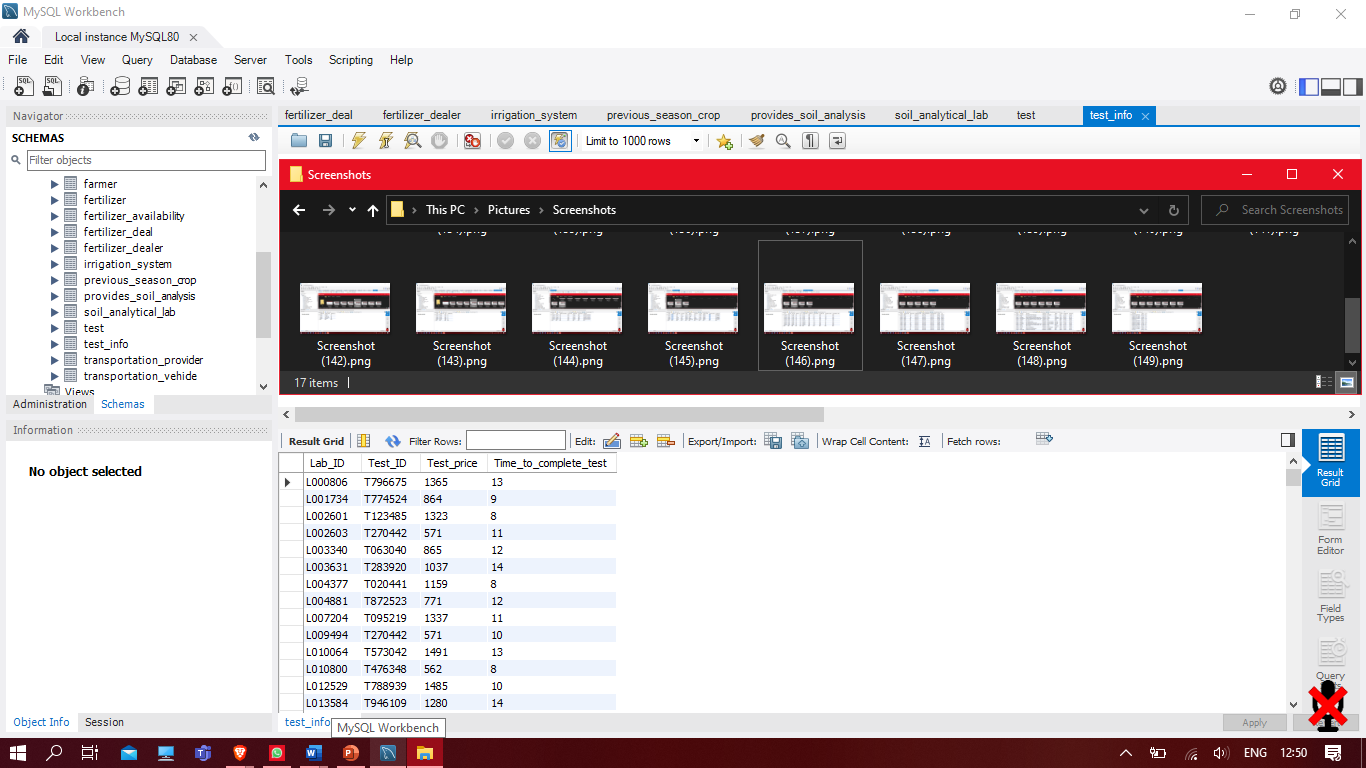
1. **soil\_analytical\_lab**



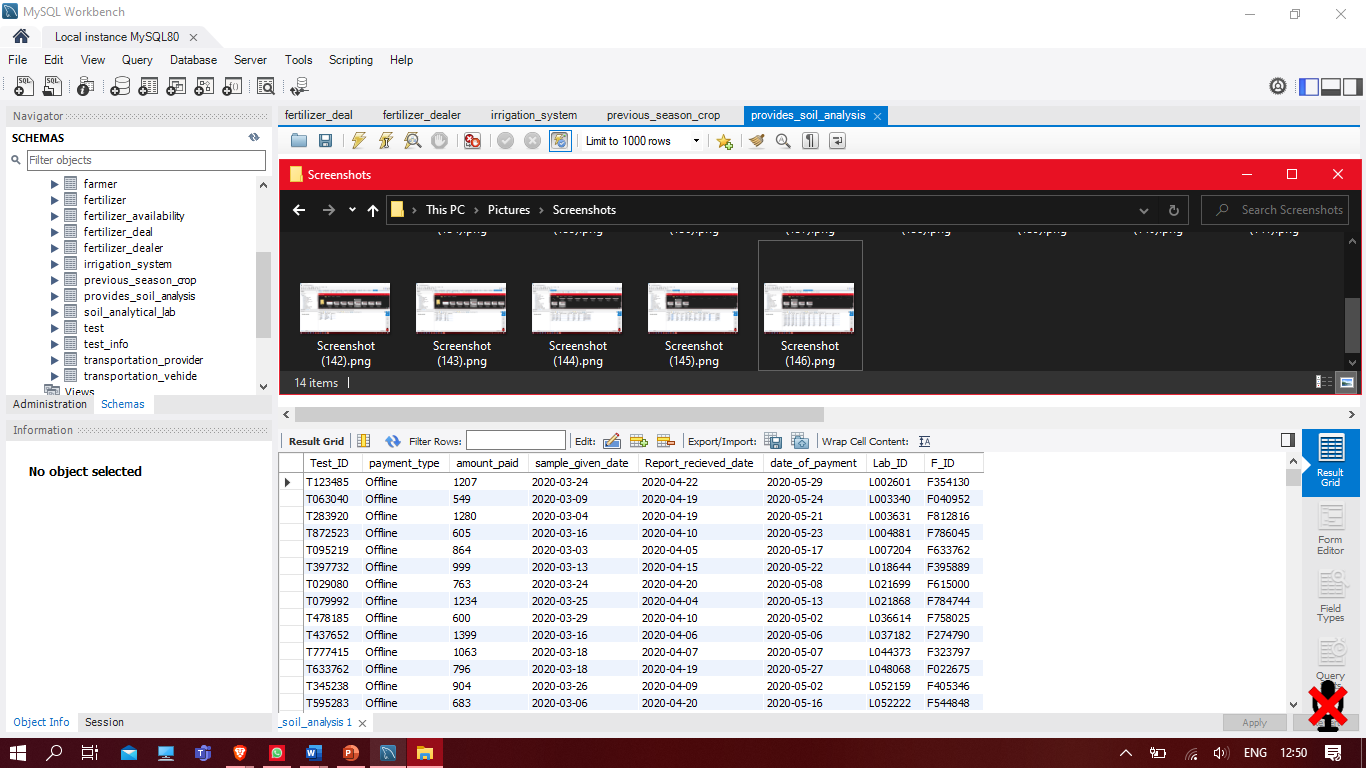
1. **test**



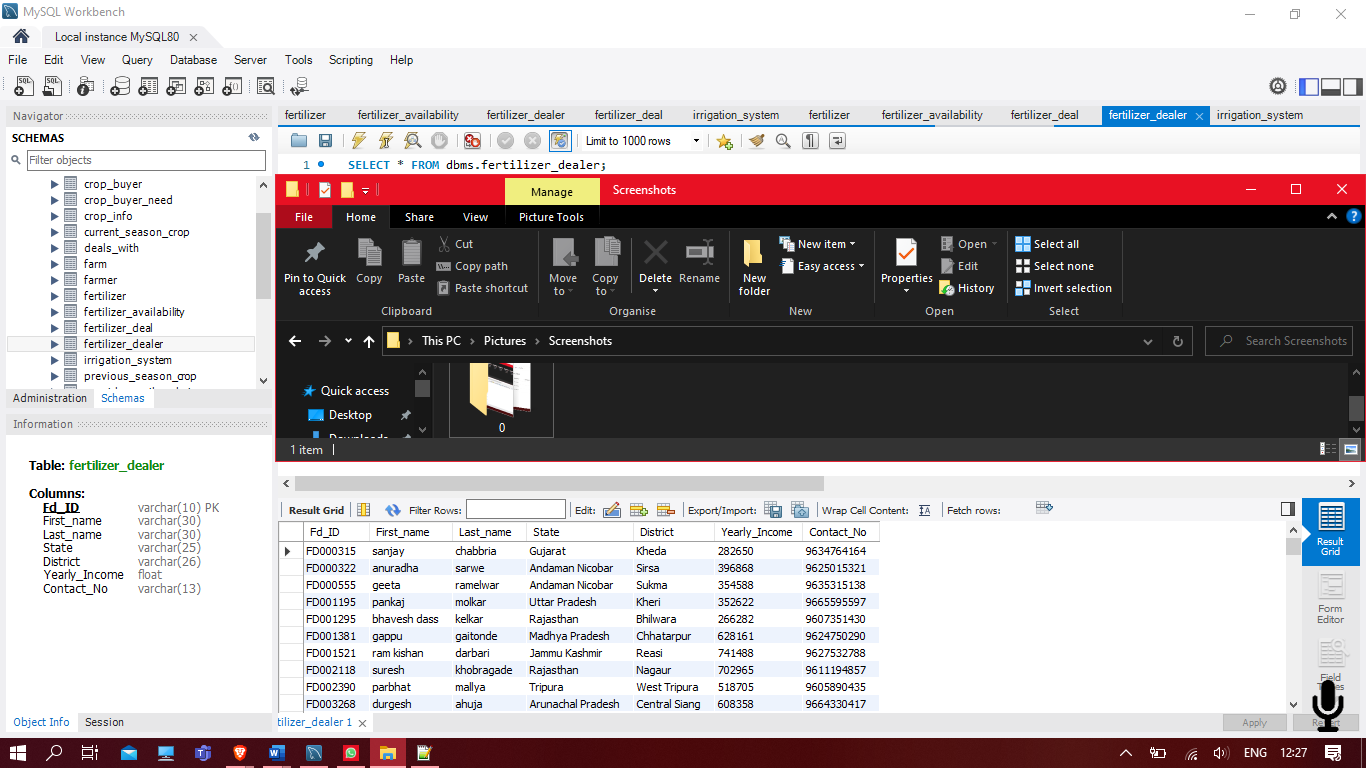
1. **test\_info**



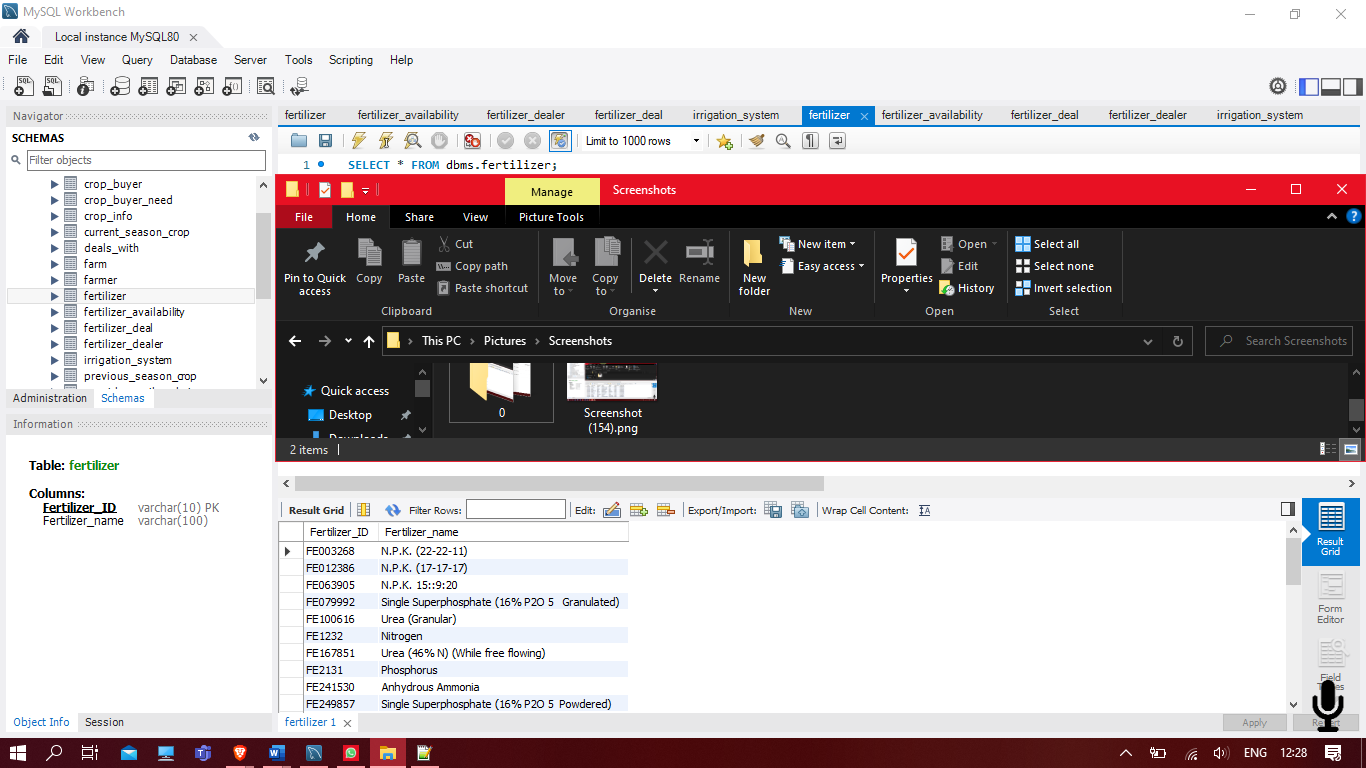
1. **provides\_soil\_analysis**



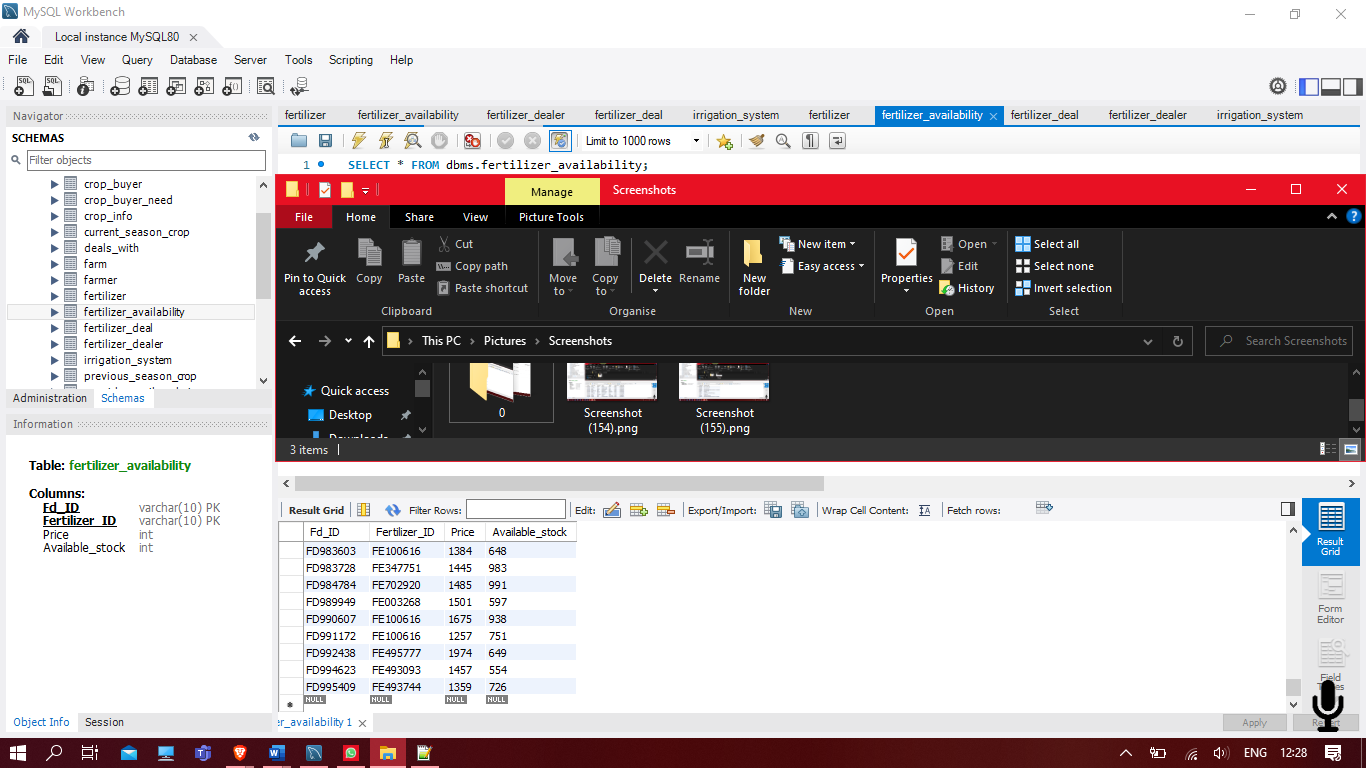
1. **fertilizer\_dealer**



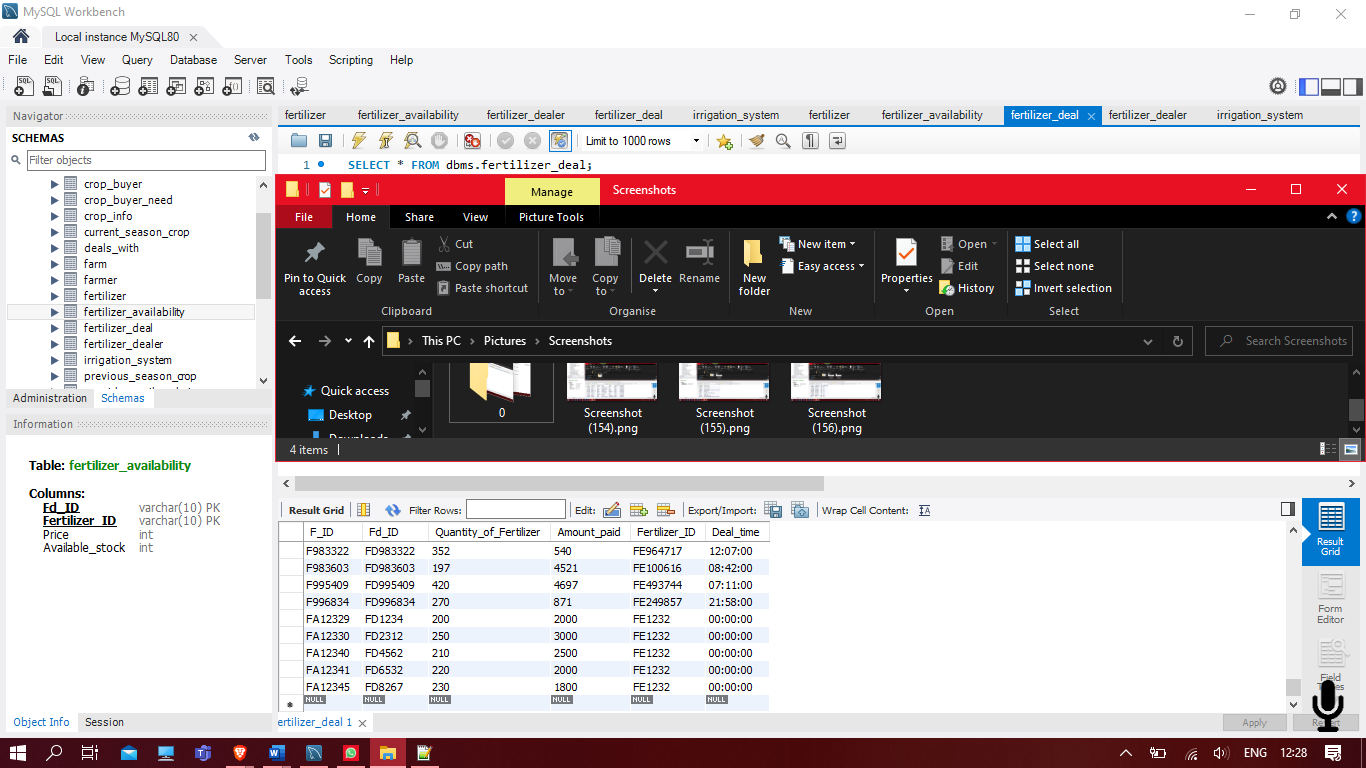
1. **fertilizer**



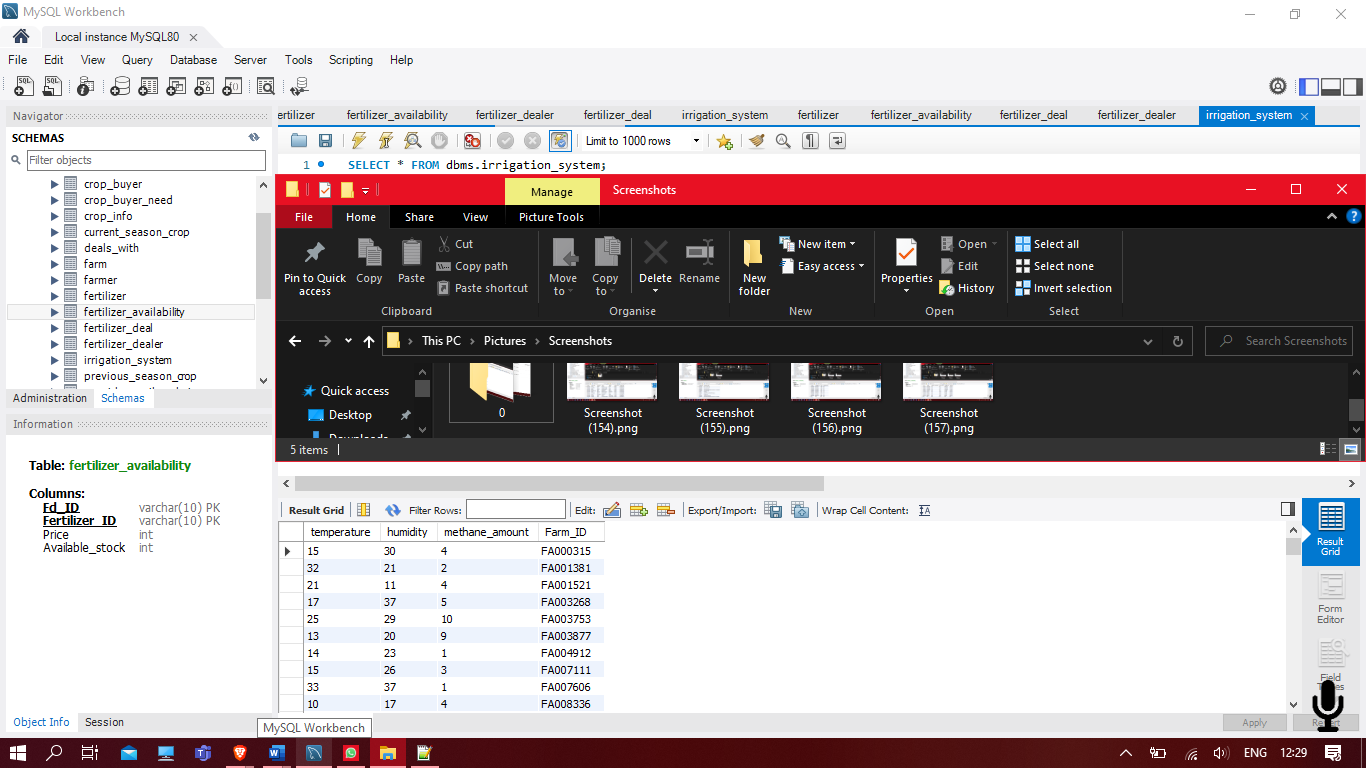
1. **fertilizer\_availability**



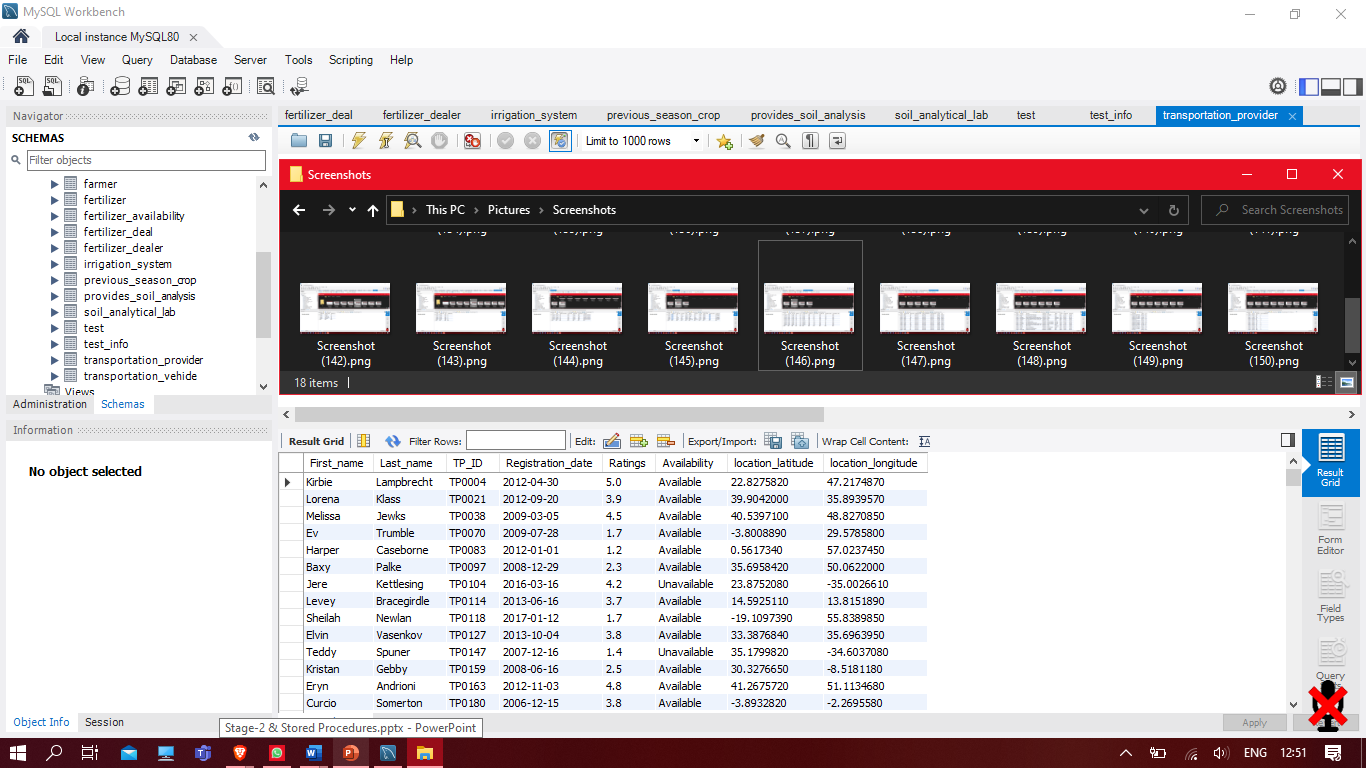
1. **fertilizer\_deal**



1. **irrigation\_system**



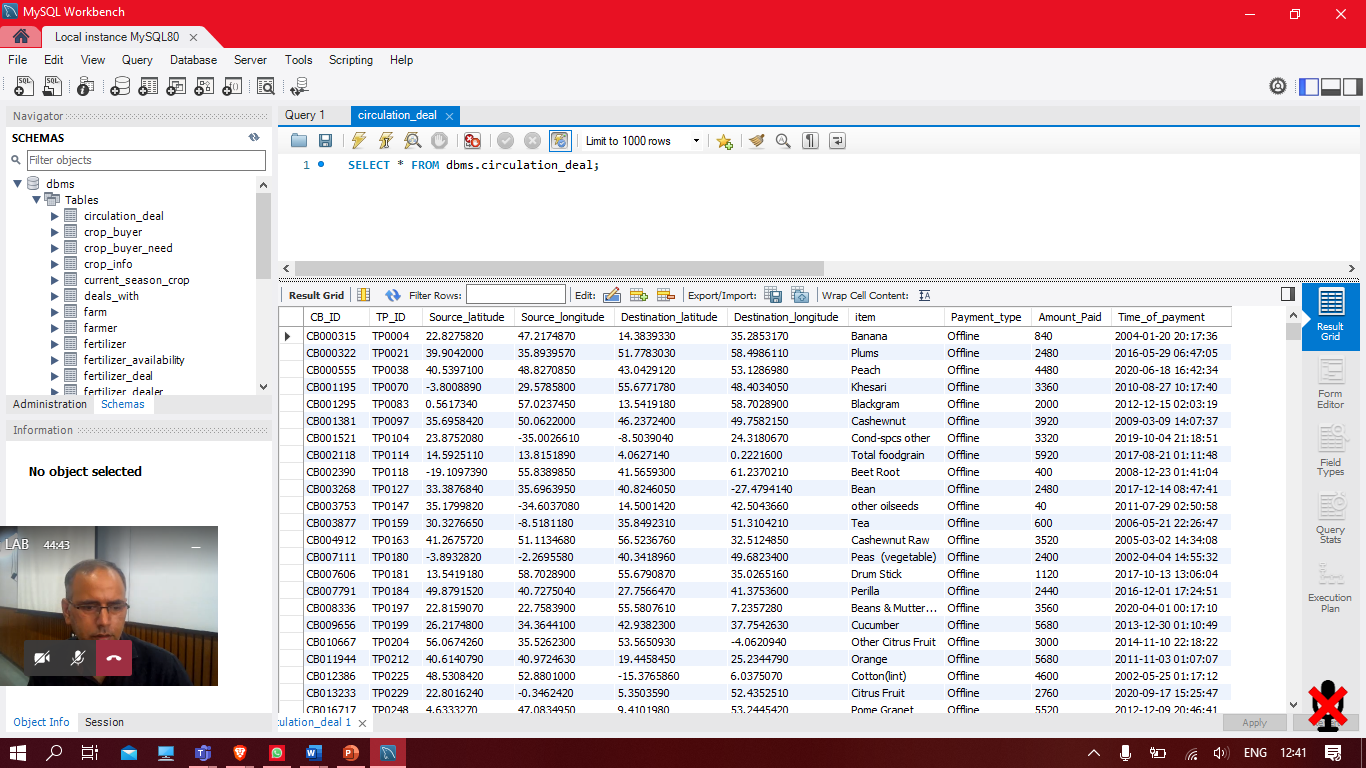
1. **transportation\_provider**



1. **transportation\_vehicle**



1. **circulation\_deal**



# SQL Queries

# (Using aggregate functions)

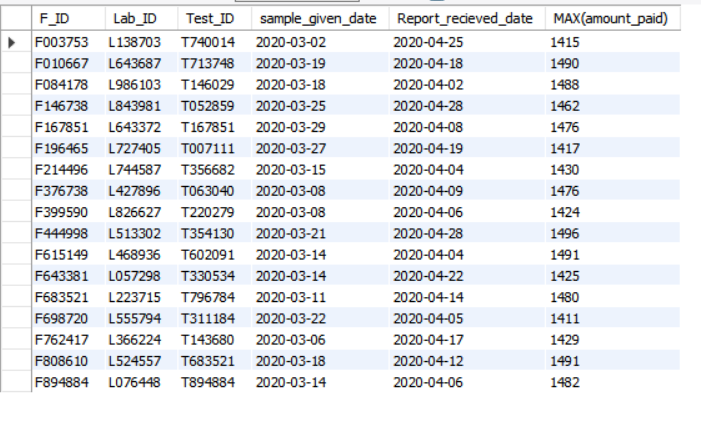
1. Display highest amount\_paid with their F\_ID , Lab\_ID , Test\_ID ,sample\_given\_date , Report\_recieved\_date for those farmers who have a higher amount\_paid in a day is within the range 1400 and 1500.

* SELECT F\_ID , Lab\_ID , Test\_ID , sample\_given\_date , Report\_recieved\_date , MAX(amount\_paid)

FROM provides\_soil\_analysis

GROUP BY F\_ID , Lab\_ID

HAVING MAX(amount\_paid) BETWEEN 1400 AND 1500;

**Output:**

1. Display the number of deals happened between Crop buyer and transportation provider in 2015 to 2016 year.

* SELECT COUNT(\*)

FROM circulation\_deal

WHERE Time\_of\_payment >= '2015-01-01 00:00:00.000' and Time\_of\_payment <' 2016-01-01 00:00:00.000 ' ;

**Output:**

1. Display average amount earned within a month 2020-01-01 to 2020-02-01 of farmers with their F\_ID.

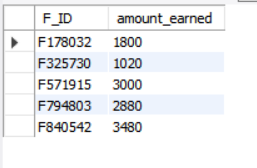
* SELECT F\_ID , AVG(Amount\_paid) as amount\_earned

FROM deals\_with

WHERE deal\_time >= '2020-01-01 00:00:00.000' and deal\_time <' 2020-02-01 00:00:00.000 '

GROUP BY F\_ID ;

**Output:**

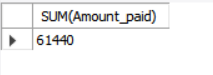


1. Display Sum of Amount\_Paid done on online within 2019 to 2020 year between transportation provider and crop buyer.

* SELECT SUM(Amount\_paid)

FROM circulation\_deal

WHERE Payment\_type = 'Online' and Time\_of\_payment >= '2019-01-01 00:00:00.000' and Time\_of\_payment <' 2020-01-01 00:00:00.000 ' ;

**Output:**

# SQL Queries

1. This query is for finding work\_status of lab on the basis of time taken to complete soil analytical test and price taken by lab. This query can be used to check labs are doing their work properly or not.

* SELECT provides\_soil\_analysis.Lab\_ID, Lab\_Name, Contact\_No, Area\_Name, provides\_soil\_analysis.Test\_ID,

amount\_paid, F\_ID, DATEDIFF(Report\_recieved\_date,sample\_given\_date) as days\_for\_test,

CASE WHEN DATEDIFF(Report\_recieved\_date,sample\_given\_date)>14 AND amount\_paid>1800 AND

provides\_soil\_analysis.Test\_ID='T30789' THEN 'Expensive and late'

WHEN DATEDIFF(Report\_recieved\_date,sample\_given\_date)>14 AND amount\_paid<=1800 AND

provides\_soil\_analysis.Test\_ID='T30789' THEN 'late with ok price'

WHEN DATEDIFF(Report\_recieved\_date,sample\_given\_date)<14 AND amount\_paid>1800 AND provides\_soil\_analysis.Test\_ID='T30789' THEN 'On time and expensive'

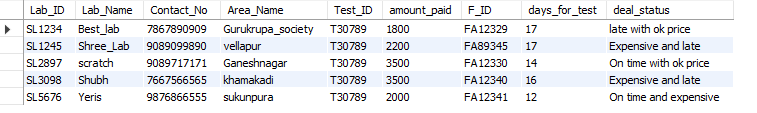
ELSE 'On time with ok price'

END as deal\_status

FROM soil\_analytical\_lab,provides\_soil\_analysis

WHERE soil\_analytical\_lab.Lab\_ID=provides\_soil\_analysis.Lab\_ID;

**Output:**



1. This query gives farmer's current season's wheat production details, price\_status and give the output in the ascending order of expected\_price. This query will be used by crop\_buyer.

* SELECT current\_season\_crop.F\_ID,First\_name, Contact\_No, Expected\_production, Expected\_price,

CASE WHEN Expected\_price>220 THEN 'expensive'

WHEN Expected\_price<=220 THEN 'affordable'

END AS price\_status

FROM farmer, current\_season\_crop

WHERE state='Gujarat' AND Expected\_production>200 AND

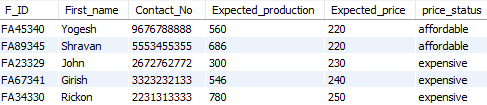
crop\_ID='WH123' AND crop\_delivery\_month='July' AND crop\_delievery\_year='2000' AND

current\_season\_crop.F\_ID=farmer.F\_ID

ORDER BY(Expected\_price) ASC

LIMIT 5;

**Output:**



1. This query gives farmer's list fulfilling some conditions about their last season crop details. This query can be used for analysis purpose.

* SELECT farmer.F\_ID,First\_name, Contact\_No,Address, previous\_season\_crop.production,state,district

FROM farmer,previous\_season\_crop

WHERE sowing\_month='January' AND harvest\_month='May' AND sowing\_year=2000

AND harvest\_year=2000 AND previous\_season\_crop.crop\_ID='WH123' AND previous\_season\_crop.F\_ID=farmer.F\_ID

AND production= ANY (SELECT production

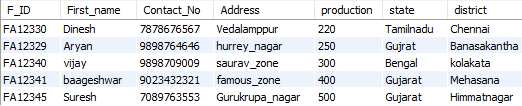
FROM previous\_season\_crop

WHERE production

BETWEEN 200.0 AND 500.0)

ORDER BY(production) ASC;

**Output:**



1. This query gives detail about those fertilizer dealer selling fertilizer than normal price and having stalk more than normal.

* SELECT fertilizer\_dealer.Fd\_ID, Contact\_No, First\_name, fertilizer\_availability.Price, Available\_stock, District, Area\_Name

FROM fertilizer\_dealer, fertilizer\_availability

WHERE fertilizer\_availability.Fertilizer\_ID='FE1232'

AND EXISTS (SELECT price

FROM fertilizer\_availability

UNION

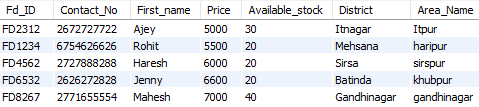
SELECT Available\_stock

FROM fertilizer\_availability

WHERE price>500 AND Available\_stock>=20) AND fertilizer\_availability.Fd\_ID= fertilizer\_dealer.Fd\_ID AND state='gujarat'

ORDER BY (fertilizer\_availability.Price) ASC;

**Output:**



# Normalization

**Table: farmer**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| F\_ID(PK) | Password | | First\_name | | Last\_name | state | district | Registration\_date |
| Birth\_date | | Contact\_No | | Yearly\_income | |

|  |  |
| --- | --- |
| FD | Minimal cover |
| F\_ID🡪First\_name  F\_ID 🡪Last\_name  F\_ID 🡪 State  F\_ID 🡪Birth\_date  F\_ID 🡪 Contact\_no  F\_ID 🡪 Registration\_date  F\_ID 🡪 Yearly\_income | F\_ID🡪 First\_name  F\_ID 🡪Last\_name  F\_ID 🡪State  F\_ID 🡪Birth\_date  F\_ID 🡪Contact\_no  F\_ID 🡪Registration\_date  F\_ID 🡪Yearly\_income |

**Candidate Key:-** F\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: farm**

|  |  |  |
| --- | --- | --- |
| F\_ID | Farm\_ID (PK) | Area\_of\_farm |

|  |  |
| --- | --- |
| FD | Minimal cover |
| Farm\_ID 🡪 F\_ID  Farm\_ID 🡪 Area | Farm\_ID 🡪 F\_ID  Farm\_ID 🡪 Area |

**Candidate Key:-** Farm\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: crop\_info**

|  |  |
| --- | --- |
| crop\_ID (PK) | crop\_name |

|  |  |
| --- | --- |
| FD | Minimal cover |
| crop\_ID 🡪 crop\_name | crop\_ID 🡪 crop\_name |

**Candidate Key:-** crop\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: current\_season\_crop**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| F\_ID (PK) | crop\_ID | | Farm\_ID (PK) | Expected\_production | Expected\_price |
| sowing\_Month (PK) | | harvest\_month (PK) | | crop\_delivery\_month |

|  |  |
| --- | --- |
| FD | Minimal cover |
| Farm\_ID 🡪 F\_ID  Farm\_ID 🡪 crop\_ID  Farm\_ID 🡪 sowing\_month  Farm\_ID 🡪 sowing\_year  Farm\_ID 🡪 harvest\_month  Farm\_ID 🡪 harvest\_year | Farm\_ID 🡪 F\_ID  Farm\_ID 🡪 crop\_ID  Farm\_ID 🡪 sowing\_month  Farm\_ID 🡪 sowing\_year  Farm\_ID 🡪 harvest\_month  Farm\_ID 🡪 harvest\_year |

**Candidate Key:-** Farm\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: crop\_buyer**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CB\_ID (PK) | | Password | | First\_name | | Last\_name | state | | district | Registration\_date |
| Birth\_date | Ratings | | Yearly\_income | | Total\_deals\_done | | |

|  |  |
| --- | --- |
| FD | Minimal cover |
| CB\_ID 🡪 Ratings  CB\_ID 🡪 Total\_deals\_done  CB\_ID🡪First\_name  CB\_ID🡪Last\_name  CB\_ID 🡪 State  CB\_ID 🡪 Birth\_date  CB\_ID 🡪 Contact\_no  CB\_ID 🡪 Registration\_date  CB\_ID 🡪 Yearly\_income | CB\_ID 🡪 Ratings  CB\_ID 🡪 Deals\_done  CB\_ID🡪First\_name  CB\_ID🡪Last\_name  CB\_ID 🡪 State  CB\_ID 🡪 Birth\_date  CB\_ID 🡪 Contact\_no  CB\_ID 🡪 Registration\_date  CB\_ID 🡪 Yearly\_income |

**Candidate Key:-** CB\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: crop\_buyer\_need**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CB\_ID(PK) | crop\_id | Needed\_quantity | Affordable\_Price | Deadline\_month |

|  |  |
| --- | --- |
| FD | Minimal cover |
| (CB\_ID)(crop\_ID) 🡪 needed\_quantity  (CB\_ID)(crop\_ID) 🡪 Affordable\_price  (CB\_ID)(crop\_ID) 🡪 Deadline\_month  (CB\_ID)(crop\_ID) 🡪 Deadline\_year | (CB\_ID)(crop\_ID) 🡪 needed\_quantity  (CB\_ID)(crop\_ID) 🡪 Affordable\_price  (CB\_ID)(crop\_ID) 🡪 Deadline\_month  (CB\_ID)(crop\_ID) 🡪 Deadline\_year |

**Candidate Key:-** (CB\_ID)(crop\_ID)

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: deals\_with**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| F\_ID(PK) | CB\_ID (PK) | crop\_ID (PK) | quantity\_of\_crop | payment\_type | Amount\_paid | deal\_time (PK) |

|  |  |
| --- | --- |
| FD | Minimal cover |
| (CB\_ID)(crop\_ID)(F\_ID)(deal\_time) 🡪quantity\_of\_crop  (CB\_ID)(crop\_ID)(F\_ID)(deal\_time)  🡪payment\_type  (CB\_ID)(crop\_ID)(F\_ID)(deal\_time)  🡪Amount\_paid | (CB\_ID)(crop\_ID)(F\_ID)(deal\_time) 🡪quantity\_of\_crop  (CB\_ID)(crop\_ID)(F\_ID)(deal\_time)  🡪payment\_type  (CB\_ID)(crop\_ID)(F\_ID)(deal\_time)  🡪Amount\_paid |

**Candidate Key:-** (CB\_ID)(crop\_ID)(F\_ID)(deal\_time)

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: soil\_analytical\_lab**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Lab\_ID (PK) | Password | Lab\_Name | Contact\_NO | state | district | Pin\_code | opening\_time |
| closing\_time |

|  |  |
| --- | --- |
| FD | Minimal cover |
| Lab\_ID 🡪 Lab\_name  Lab\_ID 🡪 Contact\_NO  Lab\_ID 🡪 state  Lab\_ID 🡪 district  Lab\_ID 🡪 opening\_time  Lab\_ID 🡪closing\_time | Lab\_ID 🡪 Lab\_name  Lab\_ID 🡪 contact\_no  Lab\_ID 🡪 state  Lab\_ID 🡪 district  Lab\_ID 🡪 opening\_time  Lab\_ID 🡪closing\_time |

**Candidate Key:-** Lab\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: test**

|  |  |  |
| --- | --- | --- |
| Test\_name | Test\_ID (PK) | Price |

|  |  |
| --- | --- |
| FD | Minimal cover |
| Test\_ID 🡪 Test\_name | Test\_ID 🡪 Test\_name |

**Candidate Key:-** Test\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: test\_info**

|  |  |  |  |
| --- | --- | --- | --- |
| Lab\_ID(PK) | Test\_ID (PK) | Test\_price | Time\_to\_complete\_test |

|  |  |
| --- | --- |
| FD | Minimal cover |
| (Test\_ID)(Lab\_ID) 🡪 Test\_price  (Test\_ID)(Lab\_ID) 🡪Days\_for\_test | (Test\_ID)(Lab\_ID) 🡪 Test\_price  (Test\_ID)(Lab\_ID) 🡪Days\_for\_test |

**Candidate Key:-** (Test\_ID)(Lab\_ID)

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: provides\_soil\_analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| Test\_ID(PK) | payment\_type | amount\_paid | sample\_given\_date (PK) |
| Report\_recieved\_date | date\_of\_payment | Lab\_ID (PK) | F\_ID (PK) |

|  |  |
| --- | --- |
| FD | Minimal cover |
| (Test\_ID)(sample\_given\_date)(Lab\_ID)(F\_ID) 🡪 (Payment\_type)(amount) (Report\_received\_date) | (Test\_ID)(sample\_given\_date)(Lab\_ID)(F\_ID) 🡪 (Payment\_type)(amount) (Report\_received\_date) |

**Candidate Key:-** (Test\_ID)(sample\_given\_date)(Lab\_ID)(F\_ID)

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: fertilizer\_dealer**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fd\_ID(PK) | First\_name | Last\_name | State | District | Pin\_code | Yearly\_Income | Contact\_No |

|  |  |
| --- | --- |
| FD | Minimal cover |
| FD\_ID🡪First\_name  FD\_ID 🡪Last\_name  FD\_ID 🡪 State  FD\_ID 🡪Birth\_date  FD\_ID 🡪 Contact\_no  FD\_ID 🡪 Registration\_date  FD\_ID 🡪 Yearly\_income | FD\_ID🡪First\_name  FD\_ID 🡪Last\_name  FD\_ID 🡪 State  FD\_ID 🡪Birth\_date  FD\_ID 🡪 Contact\_no  FD\_ID 🡪 Registration\_date  FD\_ID 🡪 Yearly\_income |

**Candidate Key:-** FD\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: fertilizer**

|  |  |
| --- | --- |
| Fertilizer\_ID (PK) | Fertilier\_name |

|  |  |
| --- | --- |
| FD | Minimal cover |
| Fertilizer\_ID 🡪 Fertilizer\_name | Fertilizer\_ID 🡪 Fertilizer\_name |

**Candidate Key:-** Fertilizer\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: fertilizer\_availability**

|  |  |  |  |
| --- | --- | --- | --- |
| Fd\_ID (PK) | Fertilizer\_ID (PK) | Price | Available\_stock |

|  |  |
| --- | --- |
| FD | Minimal cover |
| (FD\_ID)(Fertilizer\_ID) 🡪 Price  (FD\_ID)(Fertilizer\_ID) 🡪 Available\_stock | (FD\_ID)(Fertilizer\_ID) 🡪 Price  (FD\_ID)(Fertilizer\_ID) 🡪 Available\_stock |

**Candidate Key:-** (FD\_ID)(Fertilizer\_ID)

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: fertilizer\_deal**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| F\_ID(PK) | Fd\_ID (PK) | Quantity\_of\_Fertilizer | Amount\_paid | Fertilizer\_ID (PK) | Payment\_type | Deal\_time (PK) |

|  |  |
| --- | --- |
| FD | Minimal cover |
| (F\_ID)(FD\_ID)(Fertilizer\_ID)  (Deal\_time)🡪  (Amount\_paid)(Payment\_type)  (quantity\_of\_fertilizer) | (F\_ID)(FD\_ID)(Fertilizer\_ID)  (Deal\_time)🡪  (Amount\_paid)(Payment\_type)  (quantity\_of\_fertilizer) |

**Candidate Key:-** (F\_ID)(FD\_ID)(Fertilizer\_ID)(Deal\_time)

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: irrigation\_system**

|  |  |  |  |
| --- | --- | --- | --- |
| Farm\_ID (PK) | temperature | humidity | methane\_amount |

|  |  |
| --- | --- |
| FD | Minimal cover |
| Farm\_ID 🡪 temperature  Farm\_ID 🡪 humidity  Farm\_ID 🡪 methane\_amount | Farm\_ID 🡪 temperature  Farm\_ID 🡪 humidity  Farm\_ID 🡪 methane\_amount |

**Candidate Key:-** Farm\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: transportation\_provider**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| First\_name | Last\_name | | TP\_ID (PK) | Registration\_date | Ratings | Availability |
| location\_latitude | | location\_longitude | |

|  |  |
| --- | --- |
| FD | Minimal cover |
| TP\_ID 🡪 First\_name  TP\_ID 🡪 Last\_name  TP\_ID 🡪 Registration\_date  TP\_ID 🡪 Ratings  TP\_ID 🡪Availability  TP\_ID 🡪location\_latitude  TP\_ID 🡪 location\_longitude | TP\_ID 🡪 First\_name  TP\_ID 🡪 Last\_name  TP\_ID 🡪 Registration\_date  TP\_ID 🡪 Ratings  TP\_ID 🡪Availability  TP\_ID 🡪location\_latitude  TP\_ID 🡪 location\_longitude |

**Candidate Key:-** TP\_ID

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: transportation\_vehicle**

|  |  |  |
| --- | --- | --- |
| TP\_ID | Rc\_book\_number | Vehicle\_number (PK) |

|  |  |
| --- | --- |
| FD | Minimal cover |
| Vehicle\_number 🡪  TP\_ID  Vehicle\_number🡪  Vehicle\_model  Vehicle\_number🡪  Rc\_book\_number | Vehicle\_number 🡪  TP\_ID  Vehicle\_number 🡪  Vehicle\_model  Vehicle\_number 🡪  Rc\_book\_number |

**Candidate Key:-** Vehicle\_number

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

**Table: circulation\_deal**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CB\_ID (PK) | | TP\_ID (PK) | Source\_latitude | | Source\_longitude | | Destination\_latitude | | Destination\_longitude |
| item | Payment\_type | | | Amount\_Paid | | Time\_of\_payment (PK) | |

|  |  |
| --- | --- |
| FD | Minimal cover |
| (CB\_ID)(TP\_ID)(Time\_of\_payment) 🡪Source  (CB\_ID)(TP\_ID)(Time\_of\_payment) 🡪Destination,item  (CB\_ID)(TP\_ID)(Time\_of\_payment)  🡪 Payment\_type  (CB\_ID)(TP\_ID)(Time\_of\_payment)  🡪 Amount\_Paid | (CB\_ID)(TP\_ID)(Time\_of\_payment) 🡪Source  (CB\_ID)(TP\_ID)(Time\_of\_payment) 🡪Destination,item  (CB\_ID)(TP\_ID)(Time\_of\_payment)  🡪 Payment\_type  (CB\_ID)(TP\_ID)(Time\_of\_payment)🡪 Amount |

**Candidate Key:-** (CB\_ID)(TP\_ID)(Time\_of\_payment)

**Normalization:**

1NF 🡪 Yes

2NF 🡪 Yes

3NF 🡪 Yes

BCNF 🡪 Yes

# Assignment Topic

# (Stored Procedures)

1. This query will merge dataset of farmer's current and past season's crop state wise.

CREATE PROCEDURE `statewise\_farmer\_detail`(IN current\_cropID VARCHAR(10),

IN previous\_cropID VARCHAR(10),IN state\_ VARCHAR(25))

BEGIN

SELECT farmer.First\_name, current\_season\_crop.crop\_ID as current\_crop\_ID,

previous\_season\_crop.crop\_ID as previous\_crop\_ID,current\_season\_crop.Expected\_production, previous\_season\_crop.production as produced\_crop

FROM ((farmer INNER JOIN current\_season\_crop ON farmer.F\_ID =

current\_season\_crop.F\_ID) INNER JOIN previous\_season\_crop ON farmer.F\_ID =

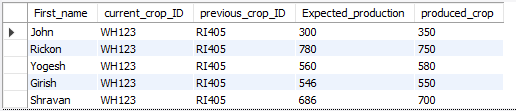
previous\_season\_crop.F\_ID) WHERE current\_season\_crop.crop\_ID= current\_cropID

AND previous\_season\_crop.crop\_ID= previous\_cropID AND state=state\_;

END

CALL statewise\_farmer\_detail ('gujarat')

**Output:**



1. This query will give the crop buyer detail with their experience status based on their ratings and total deals done.

CREATE PROCEDURE `Experienced\_crop\_buyer`(IN state\_ VARCHAR(25), IN District\_ VARCHAR(25))

BEGIN

SELECT CB\_ID, First\_Name,contact\_No, district,Ratings,Total\_deals\_done,

CASE WHEN Total\_deals\_done>=500 THEN 'experienced'

WHEN Total\_deals\_done>=300 AND Total\_deals\_done<500 THEN 'quite experienced'

ELSE 'not\_experienced'

END AS experience\_status FROM crop\_buyer

WHERE state='gujarat' AND crop\_buyer.CB\_ID= ANY (SELECT crop\_buyer.CB\_ID FROM

crop\_buyer WHERE Total\_deals\_done>200) AND state=state\_ AND district=district\_

ORDER BY(Total\_deals\_done);

END

**Output:**



1. This query will give average price and average stalk available per fertilizer dealer per state.

CREATE PROCEDURE `fertilizer\_detail`(IN state\_ VARCHAR(25),

IN Fertilizer\_ID\_ VARCHAR(10))

BEGIN

SELECT AVG(fertilizer\_availability.Price) As Avg\_price,

AVG(Available\_stock) As Avg\_available\_stock

FROM fertilizer\_availability, fertilizer\_dealer, fertilizer

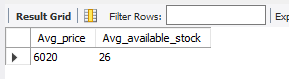
WHERE fertilizer.Fertilizer\_ID= fertilizer\_availability.Fertilizer\_ID=Fertilizer\_ID\_

AND Fertilizer\_dealer.state=state\_;

END

CALL fertilizer\_detail('gujarat','FE1232')

**Output:**



1. This query will give list of top fertilizer dealer in particular state on the basis of quantity of fertilizer they have sold.

CREATE PROCEDURE `top\_fertilizer\_dealers`(IN State\_ VARCHAR(25))

BEGIN

SELECT fertilizer\_dealer.Fd\_ID, Contact\_No, First\_name, District

FROM fertilizer\_dealer,fertilizer\_deal

WHERE fertilizer\_deal.Fertilizer\_ID='FE1232' AND

EXISTS (SELECT Quantity\_of\_Fertilizer FROM fertilizer\_deal WHERE

Quantity\_of\_Fertilizer>190 ) AND fertilizer\_dealer.state=state\_

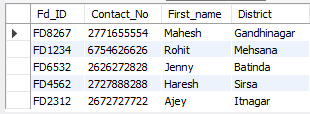
AND fertilizer\_dealer.Fd\_ID= fertilizer\_deal.Fd\_ID

ORDER BY (Amount\_paid) ASC;

END

CALL top\_fertilizer\_dealers('gujarat')

**Output:**

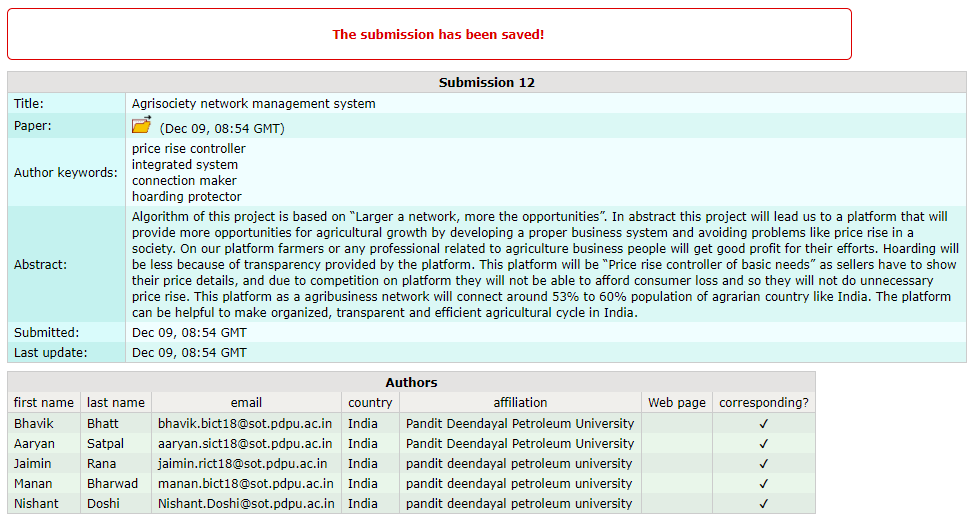


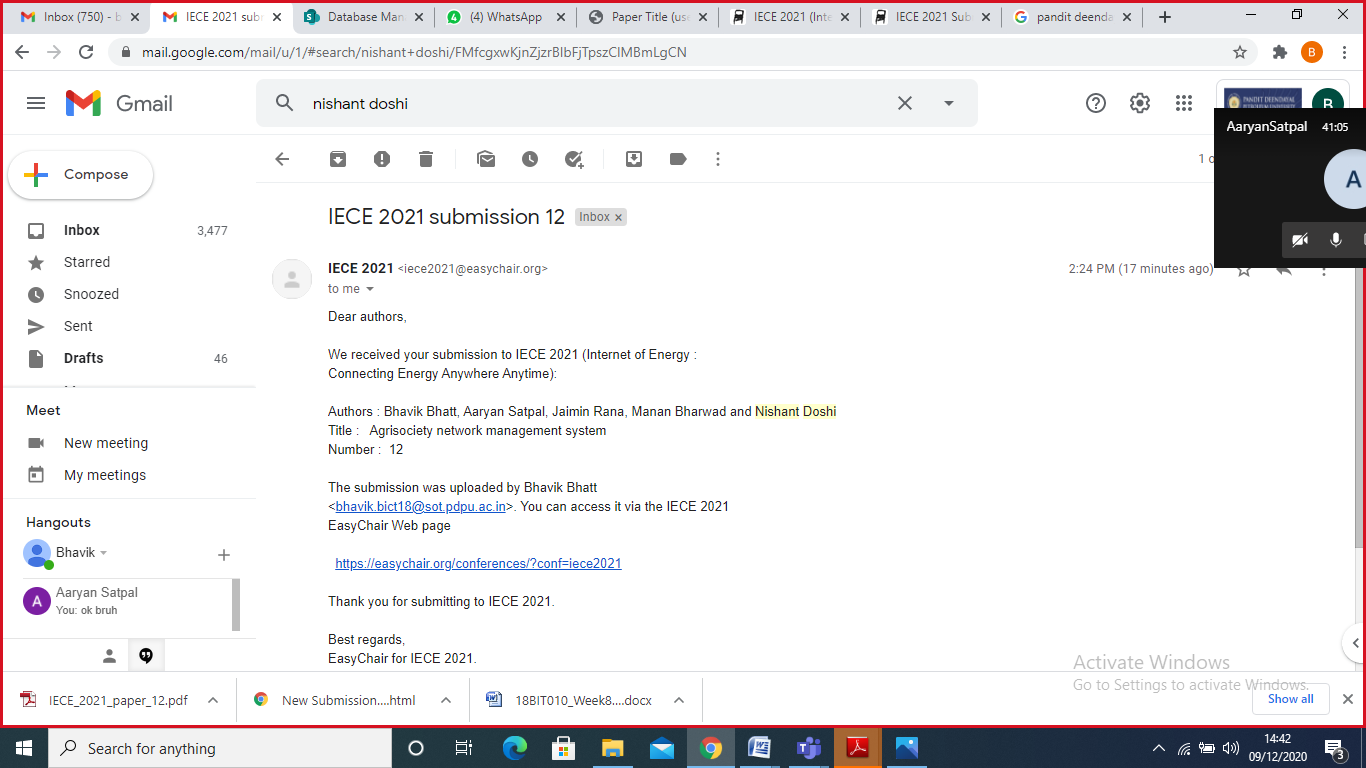
# Research Publication

# (Book Chapter)

We have submitted the book chapter of “Agrisociety Network Management System” on easy chair.

Snapshots of submission page and mail are given below:





# Our Contribution

* Before we started our work for project, we decided that we will work in a manner that each member will get to know about every corner of project and will be able to learn every topic attached to the DBMS project.
* So during the project, there were some topics in which, we simply divided our work into four parts, those tasks were:

1. DBMS Vs File system examples
2. SQL queries
3. Relational Algebra
4. Extra topic (Stored Procedure)

* But there were some tasks in which we distributed work table wise.
* Tasks like:

1. Creating ER diagram
2. Creating schema diagram
3. Creating relational diagram
4. create table commands

* During the project when the task was oriented or related to table/s, members completed tasks according to the tables they selected. Here, members and their selected tables’ information is given:

|  |  |
| --- | --- |
| Member Name | Allocated table Name |
| Bhavik | ‘Farmer’, ’Farm’, ’Crop\_info’, ’Current\_season\_crop’ ’Crop\_buyer’, ’Crop\_buyer\_need’ |
| Aaryan | ‘Lab\_owner’, ’soil\_analytical\_lab’, ’provides\_soil\_analysis’, ’test’,’test\_info’ |
| Jaimin | ‘circulation\_deal’,’deals\_with’,’transportation\_provider’,’transportation\_vehicle’ |
| Manan | ‘Fertilizer’,’Fertilizer \_dealer’, ’Fertilizer\_deal’, ’Fertilizer\_availability’, ’Irrigation\_system’ |

### In tasks like stage 1, stage 2 and research paper, it was never like one person is doing all the tasks, we made sure that each member can know everything going on in tasks, so we made several meetings, during these tasks and made sure that all members can contribution in every possible way.

### So in overall project, we can say that work done by each team member was neither more nor less and each member contributed equally.

* Throughout the project, we have divided the complex work like making queries for the project. And for the book chapter, we distributed topics among the team members.
* We divided each and every task considering the unique skills of each team member.

|  |  |
| --- | --- |
| Member Name | Work(Stage-2) |
| Bhavik | Stored procedure, My SQL queries |
| Aaryan | MY SQL QUERIES, Algebra queries |
| Jaimin | Algebra queries, Stored procedure |
| Manan | Algebra queries, My SQL queries |

|  |  |
| --- | --- |
| Member Name | Work(Book chapter) |
| Bhavik | Proposed work & Feature description of project |
| Aaryan | Literature survey and analysis of the papers |
| Jaimin | Introduction and abstraction of project |
| Manan | Query Analysis and Conclusion of project |

* How we added IOT devices to our database?

🡪In our sensor based irrigation system we embedded moisture sensor, temperature sensor and methane sensor.

🡪The reading those will be shown by these sensors will be stored in table named Irrigation\_system.

🡪And the developers will take the data from tables like farm\_area from farm table, moisture, temperature and methane information from irrigation\_system table and growing crop info from farm table and the needed levels of element will be compared with the existing and the call like watering crop will be taken.

* Here is some information about out IOT devices:

|  |  |
| --- | --- |
| IOT device name | In which table it is? |
| Moisture sensor | irrigation\_system |
| Temperature sensor | irrigation\_system |
| Methane sensor | irrigation\_system |
| IOT device name | What it does? |
| Moisture sensor | Shows moisture level |
| Temperature sensor | Shows temperature |
| Methane sensor | Methane element level |