Agriculture Database Management System

Problem Statement:-

In this project, I have designed a database management system to store information about the activities pertaining to the agricultural sector. The database will contain all the information related to farming which will be available to the admin/government and their officials to make schemes as per the status of the farmer and allot benefits as per his requirements.

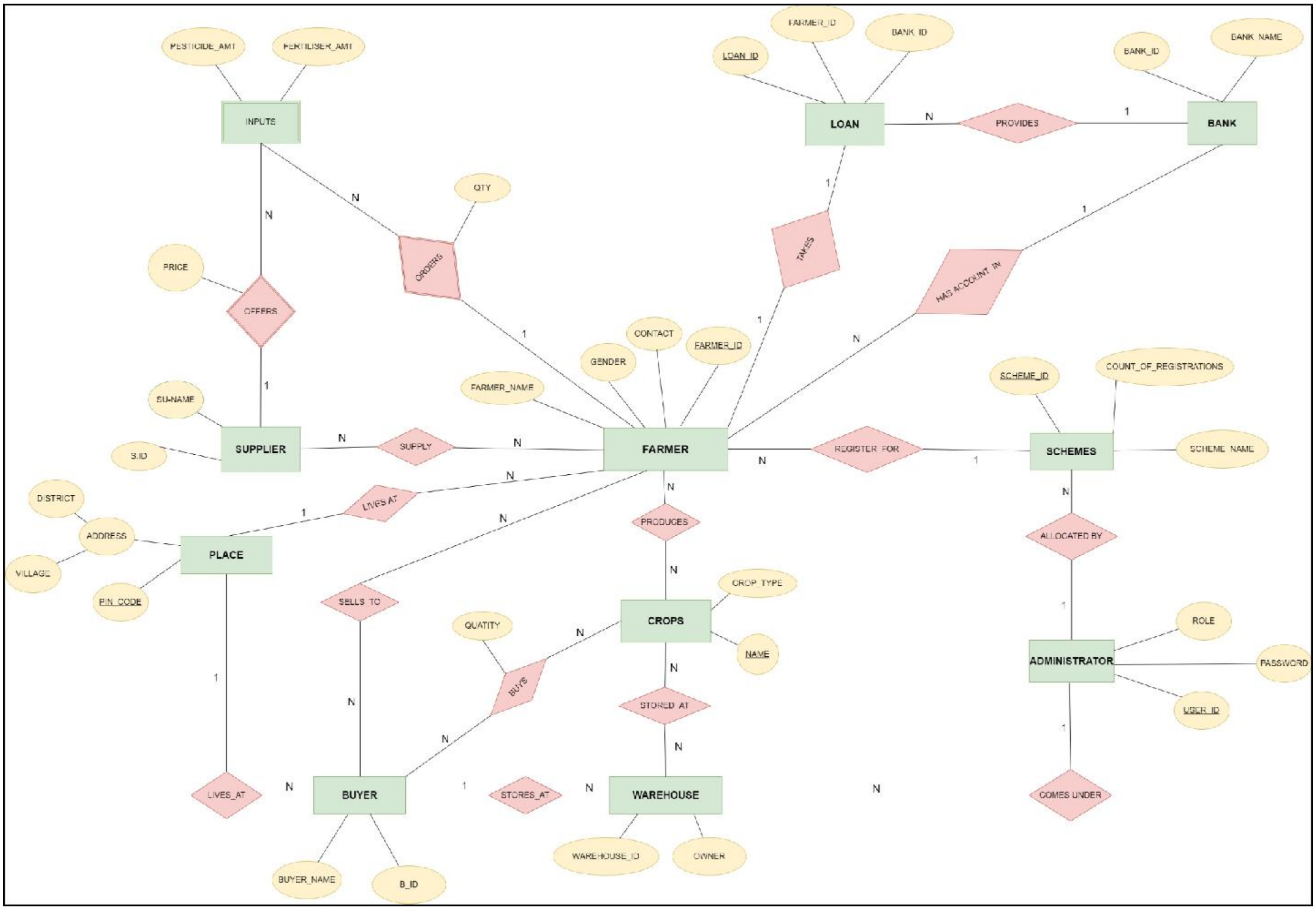
India is the fourth largest agricultural sector in the world. This sector employs over two-thirds of the country. Not only this, agriculture is a major source of income for more than 75% of Indians. Our project aims to solve farmers' problems through proper application of Information technology, which will increase the efficiency and profit in the agribusiness activities.

This database will have the farmer's personal details, schemes available for him, crops produced, inputs required by him, etc. This database management system will help the Government to monitor the status of all the farmers, whether they are small-scale, medium-scale or large-scale. Knowing about the farmers would give an overview to the government to make the agricultural budget and provide schemes that could benefit them. Farmers will be able to register for schemes as per their current condition and avail of corresponding benefits. Our database also keeps track of the farmers' bank accounts and loans to get relaxations if needed. Moreover, it also manages the quantity of crops sold and bought by them at MSP. This would make it flexible to keep a specific fixed value for crops, profitable for both farmers and the government.

Finally, in order to increase efficiency and profit in agribusiness, this database could be used to collect, retrieve and analyse data to continually improve the existing system.

Contents:

ER DIAGRAM:-



ER Model Assumptions:

- Farmers order only fertilizers and pesticides as crop inputs for agriculture.
- Multiple suppliers can supply the farmers with inputs and farmers can order from multiple suppliers.
- There is a direct link between buyers and sellers(farmers).
- Buyers utilize government owned warehouses to keep their stock.
- A warehouse has its owner and is managed by one of the government administrator's.
- Multiple schemes are launched by administrators (a government body) , but each farmer can avail facilities of only one scheme .
- For the sake of simplicity, we have assumed that one farmer can take a maximum of only one loan from a bank.
- Each crop is bought by multiple buyers .

Tables :-

1. FARMER

ATTRIBUTE	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
FARMER_ID	NUMBER	PRIMARY KEY
FARMER_NAME	VARCHAR(20)	NOT NULL
GENDER	VARCHAR(1)	NOT NULL
CONTACT	NUMBER	NOT NULL
PIN_CODE	NUMBER	FOREIGN KEY, NOT NULL
SCHEME_ID	NUMBER	FOREIGN KEY, NOT NULL

2. CROPS

ATTRIBUTE	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
NAME	VARCHAR(30)	PRIMARY KEY
CROP_TYPE	VARCHAR(30)	NOT NULL
FARMER_ID	NUMBER	FOREIGN KEY, NOT NULL

3. WAREHOUSE

ATTRIBUTE	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
WAREHOUSE_ID	NUMBER	PRIMARY KEY
OWNER	VARCHAR(30)	NOT NULL
B_ID	NUMBER	FOREIGN KEY, NOT NULL
USER_ID	NUMBER	FOREIGN KEY, NOT NULL

4. SCHEME

ATTRIBUTE	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
SCHEME_ID	NUMBER	PRIMARY KEY
NO_OF_REGISTRATIONS	NUMBER	NOT NULL
SCHEME_NAME	VARCHAR(30)	NOT NULL
USER_ID	NUMBER	FOREIGN KEY, NOT NULL

5. BUYER:

ATTRIBUTE	**DATATYPE**	**CONSTRAINT AND CHARACTERISICS**
B_ID	NUMBER	PRIMARY KEY
BUYER_NAME	VARCHAR2(40)	NOT NULL
PIN CODE	NUMBER	FOREIGN KEY, NOT NULL

6. BUYS

ATTRIBUTE	**DATATYPE**	**CONSTRAINT AND CHARACTERISICS**
B_ID	NUMBER	FOREIGN KEY, NOT NULL
CROP_NAME	NUMBER	FOREIGN KEY, NOT NULL
QUANTITY	NUMBER	NOT NULL

7. PRODUCES

ATTRIBUTE	**DATATYPE**	**CONSTRAINT AND CHARACTERISICS**
FARMER_ID	NUMBER	FOREIGN KEY, NOT NULL
CROP_NAME	NUMBER	FOREIGN KEY, NOT NULL

8. ADMINISTRATOR :

ATTRIBUTE	**DATATYPE**	**CONSTRAINT AND CHARACTERISTICS**
USER_ID	NUMBER	PRIMARY KEY
TYPE	VARCHAR2(20)	NOT NULL
PASSWORD	VARCHAR2(50)	NOT NULL

9. BANK

ATTRIBUTES	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
BANK_ID	NUMBER	PRIMARY KEY, NOT NULL
BANK_NAME	VARCHAR2(20)	NOT NULL

10. LOAN

ATTRIBUTES	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
LOAN_ID	NUMBER	PRIMARY KEY, NOT NULL
FARMER_ID	NUMBER	NOT NULL
BANK_ID	NUMBER	FORIEGN KEY, NOT NULL

11. INPUTS

ATTRIBUTES	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
PESTICIDE_AMT	NUMBER	NOT NULL
FERTILISER_AMT	NUMBER	NOT NULL
SUP_ID	NUMBER	FOREIGN KEY
PRICE	NUMBER	NOT NULL

12. SUPPLIER

ATTRIBUTES	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
SUP_ID	NUMBER	PRIMARY KEY , NOT NULL
SUP_NAME	VARCHAR2(30)	NOT NULL

13. PLACE

ATTRIBUTES	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
PINCODE	NUMBER	PRIMARY KEY , NOT NULL
VILLAGE	VARCHAR2(30)	NOT NULL
DISTRICT	VARCHAR2(30)	NOT NULL

14. SUPPLY

ATTRIBUTES	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
SUP_ID	NUMBER	FORIEGN KEY, NOT NULL
FARMER_ID	NUMBER	FORIEGN KEY, NOT NULL

15. SELLS

ATTRIBUTES	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
FARMER_ID	NUMBER	FORIEGN KEY, NOT NULL
B_ID	NUMBER	FORIEGN KEY, NOT NULL

16. STORED-AT

ATTRIBUTES	**DATA TYPE**	**CONSTRAINT AND CHARACTERISTICS**
CROP_NAME	VARCHAR2(20)	FORIEGN KEY, NOT NULL
WAREHOUSE_ID	NUMBER	FORIEGN KEY, NOT NULL

Functional Dependencies and Primary keys :

1. FARMER :

FARMER_ID -> { FARMER_NAME , GENDER , CONTACT , PIN_CODE , SCHEME_ID }
Since all fields depend on the farmer_id , (FARMER_ID $)^+$ -> R .
Hence , Farmer_id is a primary key .

2. CROPS :

NAME -> { CROP_TYPE }
Since all fields depend on the NAME, (NAME$)^+$ -> R .
Hence , NAMEis a primary key .

3. LOANS

LOAN_ID-> { FARMER_ID,,BANK_ID }
Since all fields depend on the LOAN_ID, (LOAN_ID$)^+$ -> R . Hence ,
LOAN_ID is a primary key .

4. BANK

BANK_ID-> { BANK_NAME }
Since all fields depend on the BANK_ID, (BANK_ID$)^+$ -> R . Hence
, BANK_ID is a primary key .

5. SCHEMES

SCHEME_ID-> { SCHEME_NAME, COUNT_OF_REGISTRATIONS }
Since all fields depend on the SCHEME_ID, (SCHEME_ID$)^+$ -> R .
Hence , SCHEME_ID is a primary key .

6. ADMINISTRATOR:

USER_ID-> { TYPE, PASSWORD}
Since all fields depend on the USER_ID, i.e. (USER_ID$)^+$ -> R .
Hence , USER_ID is a primary key .

7. SELLS RELATION

8. WAREHOUSE

WAREHOUSE_ID -> { OWNER , BUYER_ID }

Since all fields depend on the WAREHOUSE_ID, (WAREHOUSE_ID $)^{+}$ -> R .

Hence , WAREHOUSE_ID is a primary key .

9. STORED AT RELATION

10. PLACE

PIN_CODE ->{ VILLAGE, DISTRICT}

Since all fields depend on the BANK_ID, i.e.(PIN_CODE$)^{+}$ -> R .

Hence , PIN_CODE is a primary key .

11. SUPPLIER

SUP_ID -> { SUP_NAME }

Since all fields depend on the SUP_ID, (SUP_ID$)^{+}$ -> R .

Hence , SUP_ID is a primary key .

12. INPUTS

SUP_ID-> {PESTICIDE_AMT, FERTILISER_AMT, PRICE}

Since all fields depend on the SUP_ID, (SUP_ID$)^{+}$ -> R .

Hence , SUP_ID is a primary key which is Foreign Key from SUPPLIER Relation.

13. SUPPLY RELATION

14. PRODUCES RELATION

15. BUYS RELATION

16. BUYER

B_ID -> { BUYER_NAME}

Since all fields depend on the B_ID, (B_ID$)^{+}$ -> R . Hence ,

B_ID is a primary key .

Normalisations:

1) FARMER

Primary key: FARMER_ID
All attributes depend on the FARMER_ID, hence the table is 2NF.
All attributes depend directly on FARMER_ID, hence the table is in 3NF.
All determinants (FARMER_ID) are candidate keys, hence the table is in BCNF.

2) CROPS

Primary key: NAME
All attributes depend on the NAME, hence the table is 2NF.
All attributes depend directly on NAME, hence the table is in 3NF.
All determinants (NAME) are candidate keys, hence the table is in BCNF.

3) LOANS

Primary key: LOAN_ID
All attributes depend on the LOAN_ID, hence the table is 2NF.
All attributes depend directly on LOAN_ID, hence the table is in 3NF.
All determinants (LOAN_ID) are candidate keys, hence the table is in BCNF.

4) BANK

Primary key: BANK_ID
All attributes depend on the BANK_ID, hence the table is 2NF.
All attributes depend directly on BANK_ID, hence the table is in 3NF.
All determinants (BANK_ID) are candidate keys, hence the table is in BCNF.

5) SCHEME

Primary key: SCHEME_ID
All attributes depend on the SCHEME_ID, hence the table is 2NF.
All attributes depend directly on SCHEME_ID, hence the table is in 3NF.
All determinants (SCHEME_ID) are candidate keys, hence the table is in BCNF.

6) ADMINISTRATOR

Primary key: USER_ID
All attributes depend on the USER_ID, hence the table is 2NF.
All attributes depend directly on USER_ID, hence the table is in 3NF.

All determinants (USER_ID) are candidate keys, hence the table is in BCNF.

7) WAREHOUSE

Primary key: WAREHOUSE_ID
All attributes depend on the WAREHOUSE_ID, hence the table is 2NF.
All attributes depend directly on WAREHOUSE_ID, hence the table is in 3NF. All determinants (WAREHOUSE_ID) are candidate keys, hence the table is in BCNF.

8) PLACE

Primary key: PIN_CODE
All attributes depend on the PIN_CODE, hence the table is 2NF.
All attributes depend directly on PIN_CODE, hence the table is in 3NF.
All determinants (PIN_CODE) are candidate keys, hence the table is in BCNF.

9) SUPPLIER

Primary key: SUP_ID
All attributes depend on the SUP_ID, hence the table is 2NF.
All attributes depend directly on SUP_ID, hence the table is in 3NF.
All determinants (SUP_ID) are candidate keys, hence the table is in BCNF.

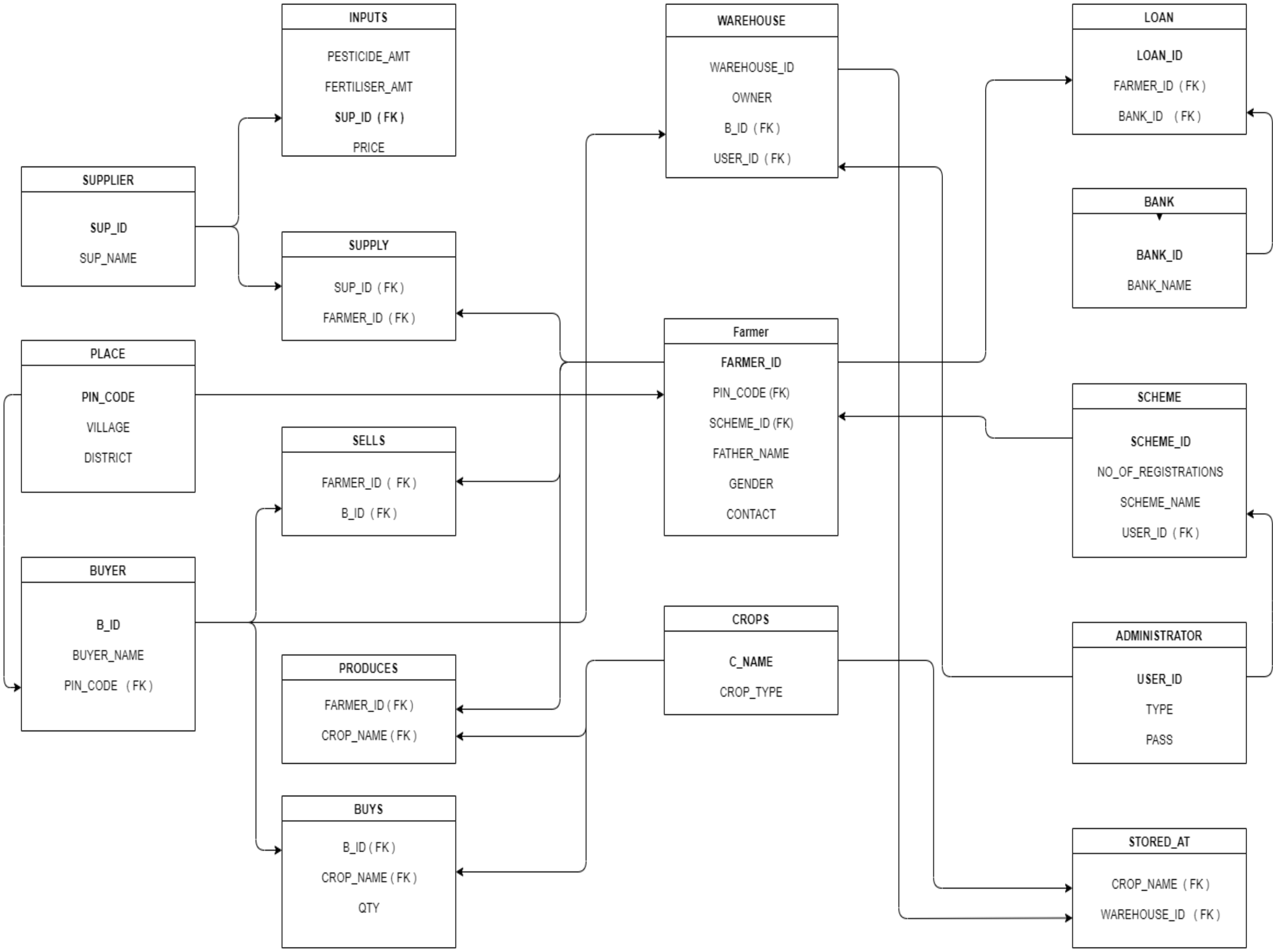
10) BUYER

Primary key: B_ID
All attributes depend on the B_ID, hence the table is 2NF.
All attributes depend directly on B_ID, hence the table is in 3NF.
All determinants (B_ID) are candidate keys, hence the table is in BCNF.

11) INPUTS

Primary key: NONE
All attributes depend on the SUP_ID , hence the table is 2NF.
All attributes depend directly on SUP_ID, hence the table is in 3NF.
All determinants (SUP_ID) are candidate keys, hence the table is in BCNF.

Relational Schema with Normalized tables



SQL CODE :

1 -> CREATION OF TABLES:-

```
CREATE TABLE
GOVERMENT (
USER_ID NUMBER,
TYPE_
VARCHAR(30),
PASSWORD_ VARCHAR(30),
PRIMARY KEY(USER_ID)
);

CREATE TABLE
PLACE (
PINCODE NUMBER,
VILLAGE VARCHAR(30),
STATE_ VARCHAR(30),
PRIMARY KEY(PINCODE)
);

CREATE TABLE
SCHEME (
SCHEME_ID NUMBER,
NO_OF_REGISTRATIONS NUMBER,
SCHEME_NAME VARCHAR(30),
USER_ID NUMBER,
PRIMARY KEY(SCHEME_ID),
FOREIGN KEY (USER_ID) REFERENCES GOVERMENT(USER_ID)
);

CREATE TABLE
FARMER (
FARMER_ID NUMBER,
FARMER_NAME
VARCHAR(30), GENDER
VARCHAR(30), CONTACT
NUMBER,
PINCODE NUMBER,
SCHEME_ID NUMBER,
PRIMARY
KEY(FARMER_ID),
FOREIGN KEY (SCHEME_ID) REFERENCES SCHEME(SCHEME_ID)
);

CREATE TABLE
BUYER (
B_ID NUMBER,
BUYER_NAME
VARCHAR(30), PINCODE
NUMBER, PRIMARY
KEY(B_ID),
FOREIGN KEY (PINCODE) REFERENCES PLACE(PINCODE)
);

CREATE TABLE
CROPS (
NAME_ VARCHAR(30),
CROP_TYPE
VARCHAR(30), B_ID
NUMBER,
FARMER_ID NUMBER,
PRIMARY
KEY(NAME_),
FOREIGN KEY (B_ID) REFERENCES BUYER(B_ID),
FOREIGN KEY (FARMER_ID) REFERENCES FARMER(FARMER_ID)
);

CREATE TABLE
WAREHOUSE (
WAREHOUSE_ID
NUMBER, OWNER_
VARCHAR(30), B_ID
NUMBER,
USER_ID NUMBER,
PRIMARY KEY(WAREHOUSE_ID),
FOREIGN KEY (B_ID) REFERENCES BUYER(B_ID),
FOREIGN KEY (USER_ID) REFERENCES GOVERMENT(USER_ID)
);
```

```
CREATE TABLE
BANK (
BANK_ID NUMBER,
BANK_NAME
VARCHAR(30), PRIMARY
KEY(BANK_ID)
);


CREATE TABLE
LOAN (
LOAN_ID NUMBER,
BORROWER_ID NUMBER,
BANK_ID NUMBER,
PRIMARY KEY(LOAN_ID),
FOREIGN KEY (BANK_ID) REFERENCES BANK(BANK_ID)
);


CREATE TABLE
SUPPLIER (
SUP_ID NUMBER,
SUP_NAME
VARCHAR(30), PRIMARY
KEY(SUP_ID)
);


CREATE TABLE
SUPPLY (
SUP_ID NUMBER,
FARMER_ID NUMBER,
FOREIGN KEY (SUP_ID) REFERENCES SUPPLIER(SUP_ID),
FOREIGN KEY (FARMER_ID) REFERENCES
FARMER(FARMER_ID)
);
```

```
CREATE TABLE
SELLS (
FARMER_ID NUMBER,
B_ID NUMBER,
FOREIGN KEY (B_ID) REFERENCES BUYER(B_ID),
FOREIGN KEY (FARMER_ID) REFERENCES FARMER(FARMER_ID)
);

CREATE TABLE
STORED_AT (
CROP_NAME
VARCHAR(20),
WAREHOUSE_ID NUMBER,
FOREIGN KEY (CROP_NAME) REFERENCES CROPS(NAME_),
FOREIGN KEY (WAREHOUSE_ID) REFERENCES WAREHOUSE(WAREHOUSE_ID)
);

CREATE TABLE
PRODUCES (
FARMER_ID NUMBER,
CROP_NAME
VARCHAR(20),
FOREIGN KEY (FARMER_ID) REFERENCES
FARMER(FARMER_ID), FOREIGN KEY (CROP_NAME)
REFERENCES CROPS(NAME_)
);

CREATE TABLE
BUYS (
B_ID NUMBER,
CROP_NAME
VARCHAR(20), QTY
NUMBER,
FOREIGN KEY (B_ID) REFERENCES BUYER(B_ID),
FOREIGN KEY (CROP_NAME) REFERENCES CROPS(NAME_)
);
```

2 -> INSERTION OF VALUES:-

```
Insert into administrator values(100001, 'Central' , 'itsPMOofINDIA');
Insert into administrator values(100002, 'Maharashtra state' , '293dne83DH'); Insert into
administrator values(100003, 'Punjab state' , '582kee48RJ');
Insert into administrator values(100004, 'Delhi UT' , '572jne28RH'); Insert into
administrator values(100005, 'Bihar state' , '683fhi82YJ');
Insert into administrator values(100006, 'Jharkhand state' , '479feu95UW'); Insert into
administrator values(100007, 'Kerela state' , '581fiq74IR');
Insert into administrator values(100008, 'Mizoram state' , '792iex38IC'); Insert into
administrator values(100009, 'Odisha state' , '582ian72IS');
```

```
Insert into place values (124303 , 'Jindran' , 'Rohtak' ); Insert
into place values (141104 , 'Badhel' , 'Ludhiana' ); Insert into
place values (110001 , 'Delhi' , 'Delhi' ); Insert into place
values (841101 , 'Saraiya' , 'Saran' ); Insert into place values
(845401 , 'Garahia' , 'Motihari' ); Insert into place values
(424304 , 'Balsane' , 'Dhule' );
Insert into place values (416216 , 'Karnoor' , 'Kolhapur' );
Insert into place values (825405 , 'Jitpur' , 'Hazaribagh' ); Insert
into place values (829206 , 'Bhusar' , 'Latehar' ); Insert into
place values (673579 , 'Pulpalli' , 'Wayanaad' ); Insert into
place values (691307 , 'Edamon' , 'Kollam' ); Insert into place
values (796261 , 'Phullen' , 'Aizwal' ); Insert into place values
(796431 , 'Belkhai' , 'Mamit' ); Insert into place values
(754200 , 'Ganapur' , 'Cuttack' ); Insert into place values
(752121 , 'Sairi' , 'Puri' );
```

```
Insert into scheme values(200001, 89264 , 'PM-Kishan Scheme' , 100001 );
Insert into scheme values(200002, 46293 , 'Solar water pump Scheme' ,100002 ); Insert
into scheme values(200003, 57292 , 'Jan Dhan Yojna' , 100003 );
Insert into scheme values(200004, 82664 , 'Kerala-Kishor-Fund' , 100004 ); Insert into
scheme values(200005, 54829 , 'Maharshtra Kranti' , 100005 ); Insert into scheme
values(200006, 68264 , 'Kisan Bima Yojna' , 100006 ); Insert into scheme
values(200007, 18234 , 'Kishan Samridhi' , 100007 ); Insert into scheme
values(200008, 59132 , 'Krishi udyog Yojna' , 100008 ); Insert into scheme
values(200009, 71837, 'Krishi Vikas' , 100009 );

Insert into farmer values (900001 , 'Diljeet Singh' , 'M' , 815739679 , 124303 , 100003 );
Insert into farmer values (900002 , 'Shivshankar Tambe' , 'M' , 817825679 , 416216 ,
100002 );
Insert into farmer values (900003 , 'Harpreet Sahu' , 'M' , 8123468264 , 141104 , 100001
);
Insert into farmer values (900004 , 'Meiyang Dumphut' , 'M' , 6824572849 , 796261 ,
100008 );
Insert into farmer values (900005 , 'Ramadhir Singh' , 'M' , 9452758257 , 110001 , 100004
);
Insert into farmer values (900006 , 'Ashok Jadhav' , 'M' , 9342857556 , 110001 , 100001 );
Insert into farmer values (900007 , 'Bibhu Pandey' , 'M' , 6201768943 , 829206 , 100006 );
Insert into farmer values (900008 , 'Mohan Lal' , 'M' , 9428537194 , 841101 , 100001 );
Insert into farmer values (900009 , 'Ram Prasad' , 'M' , 920831958 , 845401 , 100005 );
Insert into farmer values (900010 , 'Rempuia Baleng' , 'M' , 8351958365 , 752121 , 100009 );
Insert into farmer values (900011 , 'Mrin Moy' , 'M' , 9027461853 , 796431 , 100008 );
Insert into farmer values (900012 , 'Sayooj Nair' , 'M' , 9191243827 , 673579 , 100007 );
Insert into farmer values (900013 , 'Ramalinga Swamy' , 'M' , 8204745109 , 691307 ,
100001 );
```

```
Insert into farmer values (900014 , 'Tarun Yadav' , 'M' , 9829692547 , 845401 , 100005 );
Insert into farmer values (900015 , 'Nikhil Gope' , 'M' , 9430173985 , 825405 , 100006 );
Insert into farmer values (900016 , 'Gurkeerat Singh' , 'M' , 9036837159 , 141104 , 100003
);
Insert into farmer values (900017 , 'Raman Das' , 'M' , 9302947518 , 754200 , 100009 );
Insert into farmer values (900018 , 'Anil Ganguly' , 'M' , 9438194326 , 754200 , 100001 );

Insert into buyer values( 400001 , 'Raghav' , 141104 ); Insert into
buyer values( 400002 , 'Ashutosh' , 841101 ); Insert into buyer
values( 400003 , 'Deepak' , 829206 ); Insert into buyer values(
400004 , 'Avishek' , 845401 ); Insert into buyer values( 400005 ,
'Shubhank' , 110001 ); Insert into buyer values( 400006 ,
'Kulshrestha' , 124303 ); Insert into buyer values( 400007 ,
'Krishnanshu' , 754200 ); Insert into buyer values( 400008 ,
'Chirantan' , 673579 ); Insert into buyer values( 400009 ,
'Raghvendra' , 796261 );

Insert into bank values (600001, 'NITW Bank' );
Insert into bank values (600002, 'SBI' );
Insert into bank values (600003, 'Central Bank' );
Insert into bank values (600004, 'Canara Bank' );
Insert into bank values (600005, 'Kerala Bank' );
Insert into bank values (600006, 'Punjab National Bank' );
Insert into bank values (600007, 'Bank of Mizo' );
Insert into bank values (600008, 'Dena Bank' ); Insert
into bank values (600009, 'Bank of India' );

Insert into crops values ( 'Arhar' , 'pulses' ); Insert
into crops values ( 'Toor' , 'pulses' ); Insert into
crops values ( 'Rice' , 'Cereal' ); Insert into crops
values ( 'Wheat' , 'Cereals' ); Insert into crops values (
'Maize' , 'Cereals' );
```

```
Insert into crops values ( 'Jowar' , 'Cereals'); Insert
into crops values ( 'Bajra' , 'Cereals' ); Insert into crops
values ( 'Turmeric' , 'Spices' ); Insert into crops values (
'Pepper' , 'Spices' );
```

```
Insert into buys values (400008,  'Arhar',2000 ); Insert
into buys values (400002,  'Maize', 15000 ); Insert into
buys values (400004,  'Jowar', 8000 ); Insert into buys
values (400009,  'Bajra', 7000 ); Insert into buys values
(400001,  'Pepper' , 4000 ); Insert into buys values
(400006,  'Turmeric', 1000 );
```

```
Insert into produces values (900018,  'Arhar’ ); Insert
into produces values (900008,  ‘Maize’); Insert into
produces values (900001,  ‘Jowar’); Insert into
produces values (900002,  'Arhar’ ); Insert into
produces values (900007,  'Pepper’ ); Insert into
produces values (900005,  'Turmeric’ );
```

```
Insert into warehouse values(500001, 'Deepak' ,  400005 , 100008); Insert into
warehouse values(500002, 'Jamal' ,  400009 , 100005); Insert into warehouse
values(500003, 'Rishab' ,  400003 , 100009); Insert into warehouse
values(500004, 'Farhan' ,  400008 , 100002); Insert into warehouse
values(500005, 'Naman' ,  400004 , 100007); Insert into warehouse
values(500006, 'Devansh' ,  400005 , 100001); Insert into warehouse
values(500007, 'Tarun' ,  400009 , 100006); Insert into warehouse
values(500008, 'Aniket' ,  400004 , 100003); Insert into warehouse
values(500009, 'Piyush' ,  400006 , 100009); Insert into warehouse
values(500010, 'Krishna' ,  400001 , 100008); Insert into warehouse
values(500011, 'Saurabh' ,  400002 , 100004); Insert into warehouse
values(500012, 'Shaswat' ,  400006 , 100007); Insert into warehouse
values(500013, 'Devashish' , 400008 , 100005); Insert into warehouse
values(500014, 'Dipankar' ,  400007 , 100003); Insert into warehouse
values(500015, 'Manish' ,  400002 , 100001);
```

```
Insert into warehouse values(500016, 'Manav' ,  400001 , 100006); Insert into
warehouse values(500017, 'Madhav' ,  400007 , 100002); Insert into warehouse
values(500018, 'Sai' ,  400003 , 100004);
```

```
Insert into loan values ( 700001, 90001, 600003 );
Insert into loan values ( 700002, 90007, 600006 );
Insert into loan values ( 700003, 90009, 600001 );
Insert into loan values ( 700004, 90002, 600007 );
Insert into loan values ( 700005, 90005, 600002 );
Insert into loan values ( 700006, 90008, 600009 );
Insert into loan values ( 700007, 90003, 600004 );
Insert into loan values ( 700008, 90006, 600008 );
Insert into loan values ( 700009, 90004, 600005 );
```

```
Insert into supplier values (400001, 'Ashish' ); Insert
into supplier values (400002, 'Rahul' ); Insert into
supplier values (400003, 'Ranjan' ); Insert into
supplier values (400004, 'Aman' ); Insert into supplier
values (400005, 'Raushan' ); Insert into supplier values
(400006, 'Prateek' ); Insert into supplier values
(400007, 'Sahil' ); Insert into supplier values (400008,
'Farhan' ); Insert into supplier values (400009,
'Sukhbir' );
```

```
Insert into supply values (400001, 90007 );
Insert into supply values (400002, 90009 );
Insert into supply values (400003, 90005 );
Insert into supply values (400004, 90008 );
Insert into supply values (400005, 90003 );
Insert into supply values (400006, 90006 );
Insert into supply values (400007, 90004 );
Insert into supply values (400008, 90001 );
Insert into supply values (400009, 90002 );


Insert into sells values ( 90001 , 400009);
Insert into sells values ( 90002 , 400008);
Insert into sells values ( 90003 , 400007);
Insert into sells values ( 90004 , 400006);
Insert into sells values ( 90005 , 400005);
Insert into sells values ( 90006 , 400004);
Insert into sells values ( 90007 , 400003);
Insert into sells values ( 90008 , 400002);
Insert into sells values ( 90009 , 400001);

Insert into stored_at values( 'arhar' , 500006 ); Insert
into stored_at values( 'Jowar' , 500001 ); Insert into
stored_at values( 'Rice' , 500002 ); Insert into
stored_at values( 'Turmeric' , 500007 ); Insert into
stored_at values( 'Wheat' , 500009 ); Insert into
stored_at values( 'Toor' , 500003 ); Insert into
stored_at values( 'Pepper' , 500008 ); Insert into
stored_at values( 'Maize' , 500004 ); Insert into
stored_at values( 'Arhar' , 500005 );


Insert into inputs values ( 573 , 682 , 400001 , 4827);
Insert into inputs values ( 324 , 588 , 400002 , 5479);
Insert into inputs values ( 841 , 764 , 400003 , 3685);
Insert into inputs values ( 575 , 489 , 400004 , 5116);
Insert into inputs values ( 265 , 342 , 400005 , 7090);
Insert into inputs values ( 648 , 538 , 400006 , 3767);
Insert into inputs values ( 492 , 500 , 400007 , 6452);
Insert into inputs values ( 632 , 730 , 400008 , 4753);
Insert into inputs values ( 692 , 650 , 400009 , 5388);
```