CPSC 304 Project Cover Page

Milestone #: _2__

Date: _01-03-2024__

Group Number: __45__

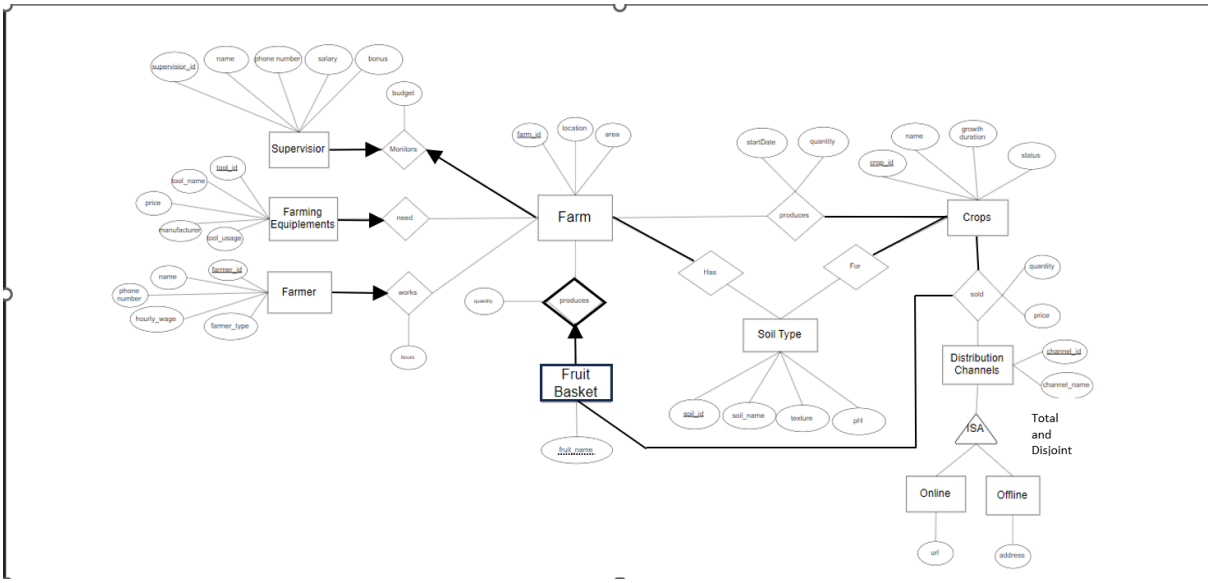
Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Pranjali Lal Das	62309620	d7j0y	pranjalidas2201@gmail.com
Chen Tong	69184950	h8r8i	tc0822@student.ubc.ca
Chang Huanfei	26638593	a6k2b	changhuanfei@163.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

2. A brief (~2-3 sentences) summary of your project.
The project is based on the database management of an agricultural firm. The model can be used by agricultural companies for farm and farmer management, crop production and crop distribution.

3. The ER diagram you are basing your item #3 (below) on.



4. The schema derived from your ER diagram (above). For the translation of the ER diagram to the relational model, follow the same instructions as in your lectures. The process should be reasonably straightforward. For each table:

a. **List the table definition (e.g., Table1(attr1: domain1, attr2: domain2, ...)). Make sure to include the domains for each attribute.**
b. **Specify the primary key (PK), candidate key, (CK) foreign keys (FK), and other constraints (e.g., not null, unique, etc.) that the table must maintain.**

Monitored_Farms(farm_id: char[20], location: char[40], area: varchar[20], supervisior_id: char[20], supervisior_name: char [20], phone_number: char[20], salary: integer, bonus: integer, budget: integer)

1. PK - farm_id
2. CK - supervisior_id, phone_number, location
3. FK - no FK
4. Constraints
    - supervisior_id is NOT NULL
    - supervisior_id is UNIQUE
    - location is UNIQUE
    - phone_number is UNIQUE

Farming_Equipments (tool_id: char[20], tool_name: char[20], tool_usage: char[20], manufacturer: char[30], price: float, **farm_id**: char[20])

1. PK - tool_id
2. CK - no CK
3. FK - farm_id

4. Constraints
    - farm_id is NOT NULL

Working_Famers (farmer_id: char[20], farmer_name: char[20], farmer_phone_number: char[20], hourly_wage: float, farmer_type: char[20], **farm_id**: char[20], farmer_hours: float)

1. PK - farmer_id
2. CK - farmer_phone_number
3. FK - farm_id
4. Constraint
    - farm_id is NOT NULL
    - farmer_phone_number is UNIQUE

FruitBasket (farm_id: char[20], fruit_name: char[20], quantity: integer)

1. PK - farm_id, fruit_name
2. CK - no CKs
3. FK - farm_id
4. Constraints - no constraint

Crops(crop_id: char[20], crop_name: char[20], growth_duration: varchar(20), crop_status: char[20])

1. PK - crop_id
2. CK - no CK
3. FK - no FKs
4. Constraints - no constraint

Produced (**crop_id**: char[20], **farm_id**: char [20], quantity: integer, sartDate: date, area: varchar(20), crop_name: char[20])

1. PK - crop_id, farm_id
2. CK - no CK
3. FK - crop_id, farm_id
4. Constraint - no constraint

Soil_Type(soil_id: char[20], soil_name: char[20], texture: char[20], pH: decimal(10,2))

1. PK - soil_id
2. CK - soil_name
3. FK - no FK
4. Constraint
    - soil_name is UNIQUE

Farm_Soil (**farm_id**: char[20], **soil_id**: char[20])

1. PK - farm_id, soil_id
2. CK - no CK
3. FK - farm_id, soil_id
4. Constraint - no constraint

Crop_Soil (**crop_id**: char[20], **soil_id**: char[20])

1. PK - crop_id, soil_id
2. CK - no CK
3. FK - crop_id, soil_id

4. Constraint - no constraint

Sold (**crop_id**: char[20], **channel_id**: char[20], quantity: integer, price: float, crop_name: char[20])

1. PK - crop_id, channel_id
2. CK - no CK
3. FK - crop_id, channel_id
4. Constraint - no constraint

Sold_Fruits(**fruit_name**: char[20], **farm_id**: char[20], **channel_id**: char[20], quantity: integer, price: float)

1. PK - fruit_name, farm_id, channel_id
2. CK - no CK
3. FK - fruit_name, farm_id, channel_id
4. Constraint - no constraint

Distribution_Channel (channel_id : char[20], channel_name: char[20])

1. PK - channel_id
2. CK - no CK
3. FK - no FK
4. Constraint - no constraint

Online(channel_id: char[20], url: varchar)

1. PK - channel_id
2. CK - url
3. FK - channel_id
4. Constraint
    - url is UNIQUE

Offline(**channel_id**: char[20], address: char[40])

1. PK - channel_id
2. CK - address
3. FK - channel_id
4. Constraint
    - address is UNIQUE

5. Functional Dependencies (FDs)

(a) Identify the functional dependencies in your relations, including the ones involving all candidate keys (including the primary key)

Monitored_Farms

1. farm_id → location, area, supervisior_id, phone_number, supervisior_name, salary, bonus, budget
2. location → phone_number , area, farm_id, supervisior_id, supervisior_name, salary, bonus, budget
3. supervisior_id → location, area, farm_id, phone_number, supervisior_name, salary, bonus, budget
4. phone_number → location, area, farm_id, supervisior_id, supervisior_name, salary, bonus, budget

5. salary → bonus
6. area → budget

Farming_Equipment

1. tool_id → tool_name, tool_usage, manufacturer, price, farm_id
2. manufacturer, tool_name → price
3. tool_name → tool_usage

Working_Famers

1. farmer_id → farmer_name, farmer_phone_number, hourly_wage, famer_type, farm_id, farmer_hours
2. farmer_phone_number → farmer_id, farmer_name, farmer_phone_number, hourly_wage, famer_type, farm_id, farmer_hours
3. farm_id, farmer_type → hourly_wage

FruitBasket

1. fruit_name, farm_id → quantity

Crops

1. crop_id→ crop_name, growth_duration, crop_status
2. crop_name → growth_duration

Produced

1. crop_id, farm_id → quantity, startDate, area, crop_name, crop_status
2. area → quantity
3. crop_name, startDate → crop_status

Soil_Type

1. soil_id → soil_name, texture, pH
2. soil_name → soil_id, texture, pH
3. texture → pH

Farm_Soil

1. farm_id, soil_id -> farm_id, soil_id

Crop_Soil

1. crop_id, soil_id -> crop_id, soil_id

Sold

1. channel_id, crop_id → crop_name, quantity, price
2. channel_id, crop_name → price, quantity

Sold_Fruits

1. channel_id, fruit_name, farm_id → quantity, price

Distribution_Channel

1. channel_id → channel_name

Online

1. channel_id → url
2. url → channel_id

Offline

1. channel_id → address
2. address→channel_id

6. Normalization

a. Normalize each of your tables to be in 3NF or BCNF. Give the list of tables, their primary keys, their candidate keys, and their foreign keys after normalization.

Monitored_Farms:

Keys

- farm_id
- supervisior_id
- phone_number
- location

Functional Dependencies

1. *farm_id → location, area, supervisior_id, phone_number, supervisior_name, salary, bonus, budget*
   (Does not violate BCNF as *farm_id* is a superkey.)

2. *location → phone_number , area, farm_id, supervisior_id, supervisior_name, salary, bonus, budget*
   (Does not violate BCNF as *location* is a superkey.)

3. *supervisior_id → location, area, farm_id, phone_number, supervisior_name, salary, bonus, budget*
   (Does not violate BCNF as *supervisior_id* is a superkey.)

4. *phone_number → location, area, farm_id, supervisior_id, supervisior_name, salary, bonus, budget*
   (Does not violate BCNF as *phone_number* is a superkey.)

5. *salary → bonus*
   (Violates BCNF as this is a non-trivial dependency and *salary* is not a superkey.)

6. *area → budget*
   (Violates BCNF as this is a non-trivial dependency and *area* is not a superkey.)

Decomposition

1. Monitored_Farms1 (farm_id, location, area, supervisior_id, phone_number, supervisior_name, salary)
   - PK - farm_id
   - CK - supervisior_id, phone_number, location
   - FK - no FK
2. Monitored_Farms2 (salary, bonus)
   - PK - salary
   - CK - no CK
   - FK - no FK
3. Monitored_Farms3 (area, budget)

- PK - area
- CK - no CK
- FK - no FK

Farming_Equipment

Keys

1. tool_id

Functional Dependencies

1. *tool_id → tool_name, tool_usage, manufacturer, price, farm_id*
   (Does not violate BCNF as *tool_id* is a superkey.)

2. *manufacturer , tool_ name → price*
   (Violates BCNF as this is a non-trivial dependency and *manufacturer* is not a superkey.)

3. *tool_name → tool_usage*
   (Violates BCNF as this is a non-trivial dependency and *tool_name* is not a superkey.)

Decomposition

1. Farming_Equipment1(tool_id, tool_name, manufacturer, **farm_id**)
   - PK - tool_id
   - CK - no CK
   - FK - farm_id
2. Farming_Equipment2 (tool_name, manufacturer, price)
   - PK - tool_usage, manufacturer
   - CK - no CK
   - FK - no FK
3. Farming_Equipment3 (tool_name, tool_usage)
   - PK - tool_name
   - CK - no CK
   - FK - no FK

Working_Famers

Keys

1. farmer_id
2. farmer_phone_number

Functional Dependencies

1. *farmer_id → farmer_name, farmer_phone_number, hourly_wage, famer_type, farm_id, farmer_hours*
   (Does not violate BCNF as *farmer_id* is a superkey.)

2. *farmer_phone_number → farmer_id, farmer_name, farmer_phone_number, hourly_wage, famer_type, farm_id, farmer_hours*
   (Does not violate BCNF as *farmer_phone_number* is a superkey.)

3. farm_id, farmer_type → hourly_wage
   (Violates BCNF as this is a non-trivial dependency and (farm_id, farmer_type) is not a superkey.)

Decomposition

1. Working_Farmers1 (farmer_id, farmer_name, farmer_phone_number, famer_type, **farm_id**, farmer_hours)
   - PK - farmer_id

- CK - farmer_phone_number
- FK - farm_id

2. Working_Farmers2 (famer_type**, farm_id**, hourly_wage)
    - PK - famer_type, farm_id
    - CK - no CK
    - FK - farm_id

FruitBasket

Keys

1. fruit_name, farm_id

Functional Dependencies

1. *fruit_name, farm_id → quantity*
   (Does not violate BCNF as *(fruit_name, farm_id)* is a superkey.)

Decomposition

- No decomposition
- FruitBasket (fruit_name, **farm_id**, quantity)
- PK - fruit_name, farm_id
- CK - no CK
- FK - farm_id

Crops

Keys

1. crop_id

Functional Dependencies

1. *crop_id→ crop_name, growth_duration, crop_status*
   (Does not violate BCNF as *crop_id* is a superkey.)

2. *crop_name → growth_duration*
   (Violates BCNF as this is a non-trivial dependency and crop_name is not a superkey.)

Decomposition

1. Crops1 (crop_id, crop_name, crop_status)
    - PK - crop_id
    - CK - no CK
    - FK - no FK

2. Crops2 (crop_name, growth_duration)
    - PK - crop_name
    - CK - no CK
    - FK - no FK

Produced

Keys

1. crop_id, farm_id

Functional Dependencies

1. *crop_id, farm_id → quantity, startDate, area, crop_name*
   (Does not violate BCNF as crop_id, farm_id is a superkey.)

2. *area → quantity*
   (Violates BCNF as this is a non-trivial dependency and area is not a superkey.)

3. *crop_name, startDate → crop_status*
   (Violates BCNF as this is a non-trivial dependency and (crop_name, startDate) is not a superkey.)

Decomposition

1. Produced1 (**crop_id, farm_id**, startDate, area, crop_name)
   - PK - crop_id, farm_id
   - CK - no CK
   - FK - crop_id, farm_id
2. Produced2 (area, quantity)
   - PK - area
   - CK - no CK
   - FK - no FK
3. Produced3 (crop_name, startDate, crop_status)
   - PK - crop_name, startDate
   - CK - no CK
   - FK - no FK

Soil_Type

Keys

1. soil_id

Functional Dependencies

1. *soil_id → soil_name, texture, pH*
   (Does not violate BCNF as soil_id is a superkey.)

2. *soil_name → soil_id, texture, pH*
   (Does not violate BCNF as soil_name is a superkey.)

3. *texture → pH*
   (Violates BCNF as this is a non-trivial dependency and texture is not a superkey.)

Decomposition

1. Soil_Type1 (soil_id, soil_name, texture)
   - PK - soil_id
   - CK - no CK
   - FK - no FK
2. Soil_Type2 (texture, pH)
   - PK - texture
   - CK - no CK
   - FK - no FK

Farm_Soil

Keys

1. farm_id, soil_id

Functional Dependencies

1. *farm_id, soil_id -> farm_id, soil_id*
   (Does not violate BCNF as it is a trivial functional dependency)

Decomposition

- No decomposition
- Farm_Soil(**farm_id, soil_id**)
- PK - farm_id, soil_id
- CK - no CK
- FK - farm_id, soil_id

Crop_Soil

Keys

1. crop_id, soil_id

Functional Dependencies

1. *crop_id, soil_id -> farm_id, soil_id*
   (Does not violate BCNF as it is a trivial functional dependency)

Decomposition

- No decomposition
- Crop_Soil(**crop_id**, **soil_id**)
- PK - crop_id, soil_id
- CK - no CK
- FK - crop_id, soil_id
-

Sold

Keys

1. channel_id, crop_id

Functional Dependencies

1. channel_id, crop_id → crop_name, quantity, price
   (Does not violate BCNF as channel_id, crop_id is a superkey.)

2. channel_id, crop_name → price, quantity
   (Violates BCNF as this is a non-trivial dependency and (channel_id, crop_name) is not a superkey.)

Decomposition

1. Sold1(**channel_id**, **crop_id**, crop_name)
   - PK - channel_id
   - CK - no CK
   - FK - channel_id, crop_id
2. Sold2 (**channel_id,** crop_name, quantity, price)
   - PK - channel_id, crop_name
   - CK - no CK
   - FK - channel_id

Sold_Fruits

Keys

1. channel_id, fruit_name, farm_id

Functional Dependencies

1. *channel_id, fruit_name, farm_id → quantity, price*
   (Does not violate BCNF as *channel_id, fruit_name, farm_id* is a superkey.)

Decomposition

- No decomposition
- Sold_Fruits(**channel_id, fruit_name, farm_id**, quantity, price)
- PK - channel_id, fruit_name, farm_id
- CK - no CK
- FK - channel_id, fruit_name, farm_id

Distribution_Channel

Keys

1. channel_id

Functional Dependencies

1. *channel_id → channel_name*
   (Does not violate BCNF as channel_id is a superkey.)

Decomposition

- No decomposition
- Distribution_Channel(channel_id, channel_name)
- PK - channel_id
- CK - no CK
- FK - no FK

Online

Keys

1. channel_id
2. url

Functional Dependencies

1. *channel_id → url*
   (Does not violate BCNF as channel_id is a superkey.)
2. *url → channel_id*
   (Does not violate BCNF as url is a superkey.)

Decomposition

- No decomposition
- Online(**channel_id**, url)
- PK - channel_id
- CK - url
- FK - channel_id

Offline

Keys

1. channel_id
2. address

Functional Dependencies

1. *channel_id → address*
   (Does not violate BCNF as channel_id is a superkey.)
2. *address → channel_id*
   (Does not violate BCNF as *address* is a superkey.)

Decomposition

- No decomposition
- Offline (**channel_id**, address)
- PK - channel_id
- CK - address
- FK - channel_id

All Relations

- Monitored_Farms1 (farm_id, location, area, supervisior_id, phone_number, supervisior_name, salary)
- Monitored_Farms2 (salary, bonus)
- Monitored_Farms3 (area, budget)
- Farming_Equipment1(tool_id, tool_name, manufacturer, **farm_id**)
- Farming_Equipment2 (tool_name, manufacturer, price)
- Farming_Equipment3 (tool_name, tool_usage)

- Working_Farmers1 (farmer_id, farmer_name, farmer_phone_number, famer_type, **farm_id**, farmer_hours)
- Working_Farmers2 (famer_type**, farm_id**, hourly_wage)
- FruitBasket (fruit_name, **farm_id**, quantity)
- Crops1 (crop_id, crop_name, crop_status)
- Crops2 (crop_name, growth_duration)
- Produced1 (**crop_id, farm_id**, startDate, area, crop_name)
- Produced2 (area, quantity)
- Produced3 (crop_name**,** startDate, crop_status)
- Soil_Type1 (soil_id, soil_name, texture)
- Soil_Type2 (texture, pH)
- Farm_Soil(**farm_id, soil_id**)
- Crop_Soil(**crop_id**, **soil_id**)
- Sold1(**channel_id**, **crop_id**, crop_name)
- Sold2 (**channel_id,** crop_name, quantity, price)
- Sold_Fruits(**channel_id, fruit_name, farm_id**, quantity, price)
- Distribution_Channel(channel_id, channel_name)
- Online(**channel_id**, url)
- Offline (**channel_id**, address)

7. The SQL DDL statements required to create all the tables from item #6. The statements should use the appropriate foreign keys, primary keys, UNIQUE constraints, etc.

1. *Monitored_Farms1(farm_id, location, area, supervisior_id, phone_number, supervisior_name, salary)*

```
CREATE TABLE Monitored_Farms1 (
   farm_id CHAR(20) PRIMARY KEY,
   location CHAR(40) UNIQUE,
   area VARCHAR(20),
   supervisior_id CHAR(20) NOT NULL UNIQUE,
   phone_number CHAR(20)  UNIQUE,
   supervisior_name CHAR(20),
   salary INTEGER
);
```

2. *Monitored_Farms2(salary, bonus)*

```
CREATE TABLE Monitored_Farms2 (
   salary INTEGER PRIMARY KEY,
   bonus INTEGER
);
```

3. *Monitored_Farms3(area, budget)*

```
CREATE TABLE Monitored_Farms3 (
   area VARCHAR(20) PRIMARY KEY,
   budget INTEGER
);
```

4. *Farming_Equipment1(tool_id, tool_name, manufacturer, **farm_id**)*

```
CREATE TABLE Farming_Equipment1 (
   tool_id CHAR(20) PRIMARY KEY,
   tool_name CHAR(20),
   manufacturer CHAR(30),
   farm_id CHAR(20) NOT NULL,
   FOREIGN KEY (farm_id) REFERENCES Monitored_Farms1(farm_id)
      ON UPDATE CASCADE
      ON DELETE CASCADE

);
```

5. *Farming_Equipment2 (tool_name, manufacturer, price)*

```
CREATE TABLE Farming_Equipment2 (
   tool_name CHAR(20),
   manufacturer CHAR(30),
   price FLOAT,
   PRIMARY KEY (tool_usage, manufacturer)
);
```

6. *Farming_Equipment3 (tool_name, tool_usage)*

```
CREATE TABLE Farming_Equipment3 (
   tool_name CHAR(20),
   tool_usage CHAR(20),
   PRIMARY KEY (tool_name),
);
```

7. *Working_Farmers1 (farmer_id, farmer_name, farmer_phone_number, famer_type, **farm_id**, farmer_hours)*

```
CREATE TABLE Working_Farmers1 (
   farmer_id CHAR(20) PRIMARY KEY,
   farmer_name CHAR(20),
   farmer_phone_number CHAR(20) NOT NULL UNIQUE,
   farmer_type CHAR(20),
   farm_id CHAR(20) NOT NULL,
   farmer_hours FLOAT,
   FOREIGN KEY (farm_id) REFERENCES Monitored_Farms1(farm_id)
   ON UPDATE CASCADE
  );
```

8. *Working_Farmers2(famer_type, **farm_id**, hourly_wage)*

```
CREATE TABLE Working_Farmers2 (
```

```
    farmer_type CHAR(20),
    farm_id CHAR(20),
    hourly_wage FLOAT,
    PRIMARY KEY (farmer_type, farm_id),
    FOREIGN KEY (farm_id) REFERENCES Monitored_Farms1(farm_id)
    ON UPDATE CASCADE
   );
```

9. *FruitBasket(fruit_name, **farm_id**, quantity)*

```
CREATE TABLE FruitBasket (
    fruit_name CHAR(20),
    farm_id CHAR(20),
    quantity INTEGER,
    PRIMARY KEY (fruit_name, farm_id),
    FOREIGN KEY (farm_id) REFERENCES Monitored_Farms1(farm_id)
    ON UPDATE CASCADE
    ON DELETE CASCADE
);
```

10. *Produced1 (**crop_id**, **farm_id**, startDate, area, crop_name)*

```
CREATE TABLE Produced1 (
    crop_id CHAR(20),
    farm_id CHAR(20),
    startDate DATE,
    area VARCHAR(20),
    crop_name CHAR(20),
    PRIMARY KEY (crop_id, farm_id),
    FOREIGN KEY (crop_id) REFERENCES Crops1(crop_id)
    ON UPDATE CASCADE,
    FOREIGN KEY (farm_id) REFERENCES Monitored_Farms1(farm_id)
    ON UPDATE CASCADE
);
```

11. *Produced2 (area, quantity)*

```
CREATE TABLE Produced2 (
    area VARCHAR(20)  PRIMARY KEY,
    quantity INTEGER
);
```

12. *Produced3 (crop_name, startDate, crop_status)*

```
CREATE TABLE Produced3 (
    crop_name CHAR(20),
    startDate DATE,
    crop_status CHAR(20),
    PRIMARY KEY (crop_name, startDate)
);
```

13. *Crops1(crop_id, crop_name, crop_status)*

```
CREATE TABLE Crops1 (
   crop_id CHAR(20) PRIMARY KEY,
   crop_name CHAR(20),
   crop_status CHAR(20)
);
```

14. *Crops2(crop_name, growth_duration)*

```
CREATE TABLE Crops2 (
   crop_name CHAR(20) PRIMARY KEY,
   growth_duration VARCHAR(20)
);
```

15. Soil_Type1 (soil_id, soil_name, texture)

```
CREATE TABLE Soil_Type1 (
   soil_id CHAR(20) PRIMARY KEY,
   soil_name CHAR(20) UNIQUE,
   texture CHAR(20)
);
```

16. Soil_Type2 (texture, pH)

```
CREATE TABLE Soil_Type2 (
   texture CHAR(20) PRIMARY KEY,
   pH DECIMAL(10,2)
);
```

17. Farm_Soil(**farm_id**, **soil_id**)

```
CREATE TABLE Farm_Soil (
   farm_id CHAR(20),
   soil_id CHAR(20),
   PRIMARY KEY (farm_id, soil_id),
   FOREIGN KEY (farm_id) REFERENCES Monitored_Farms1(farm_id)
   ON UPDATE CASCADE,
   FOREIGN KEY (soil_id) REFERENCES Soil_Type1(soil_id)
   ON UPDATE CASCADE
);
```

18. Crop_Soil(**crop_id**, **soil_id**)

```
CREATE TABLE Crop_Soil (
   crop_id CHAR(20),
   soil_id CHAR(20),
```

```
    PRIMARY KEY (crop_id, soil_id),
    FOREIGN KEY (crop_id) REFERENCES Crops1(crop_id)
    ON UPDATE CASCADE,
    FOREIGN KEY (soil_id) REFERENCES Soil_Type1(soil_id)
    ON UPDATE CASCADE
);
```

19. Sold1 (**channel_id**, **crop_id**, crop_name,)

```
CREATE TABLE Sold1 (
    channel_id CHAR(20),
    crop_id CHAR(20),
    crop_name CHAR(20),
    PRIMARY KEY (channel_id, crop_id),
    FOREIGN KEY (channel_id) REFERENCES Distribution_Channel(channel_id)
    ON UPDATE CASCADE,
    FOREIGN KEY (crop_id) REFERENCES Crops1(crop_id)
    ON UPDATE CASCADE
);
```

20. Sold2 (**channel_id**, crop_name, quantity, price)

```
CREATE TABLE Sold2 (
    channel_id CHAR(20),
    crop_name CHAR(20),
    quantity INTEGER,
    price FLOAT,
    PRIMARY KEY (channel_id, crop_name),
    FOREIGN KEY (channel_id) REFERENCES Distribution_Channel(channel_id)
    ON UPDATE CASCADE
);
```

21. Distribution_Channel(channel_id, channel_name)

```
CREATE TABLE Distribution_Channel (
    channel_id CHAR(20) PRIMARY KEY,
    channel_name CHAR(20)
);
```

22. Online(**channel_id**, url)

```
CREATE TABLE Online (
    channel_id CHAR(20) PRIMARY KEY,
    url VARCHAR UNIQUE,
    FOREIGN KEY (channel_id) REFERENCES Distribution_Channel(channel_id)
    ON UPDATE CASCADE
);
```

23. Offline (**channel_id**, address)

```
CREATE TABLE Offline (
```

```
    channel_id CHAR(20) PRIMARY KEY,
    address CHAR(40) UNIQUE,
    FOREIGN KEY (channel_id) REFERENCES Distribution_Channel(channel_id)
    ON UPDATE CASCADE
);
```

8. INSERT statements to populate each table with at least 5 tuples. You will likely want to have more than 5 tuples so that you can have meaningful queries later.

```
INSERT INTO Monitored_Farms1 (farm_id, location, area, supervisior_id, phone_number, supervisior_name, salary) VALUES('FARM001', '1050 West 14th Ave', '100', 'SUP001', '555-0101', 'John Doe', 70000);
INSERT INTO Monitored_Farms1 (farm_id, location, area, supervisior_id, phone_number, supervisior_name, salary) VALUES('FARM002', '2020 East 7th Ave', '150', 'SUP002', '555-0102', 'Jane Smith', 75000);
INSERT INTO Monitored_Farms1 (farm_id, location, area, supervisior_id, phone_number, supervisior_name, salary) VALUES('FARM003', '3080 North 3rd St', '120', 'SUP003', '555-0103', 'Emily White', 72000);
INSERT INTO Monitored_Farms1 (farm_id, location, area, supervisior_id, phone_number, supervisior_name, salary) VALUES('FARM004', '4500 South 12th St', '200', 'SUP004', '555-0104', 'Michael Brown', 68000);
INSERT INTO Monitored_Farms1 (farm_id, location, area, supervisior_id, phone_number, supervisior_name, salary) VALUES('FARM005', '5190 West 20th Ave', '180', 'SUP005', '555-0105', 'Alex Johnson', 71000);

INSERT INTO Monitored_Farms2 (salary, bonus) VALUES (70000, 5000);
INSERT INTO Monitored_Farms2 (salary, bonus) VALUES (75000, 5500);
INSERT INTO Monitored_Farms2 (salary, bonus) VALUES (72000, 5200);
INSERT INTO Monitored_Farms2 (salary, bonus) VALUES (68000, 4800);
INSERT INTO Monitored_Farms2 (salary, bonus) VALUES (71000, 5100);

INSERT INTO Monitored_Farms3 (area, budget) VALUES ('100', 100000);
INSERT INTO Monitored_Farms3 (area, budget) VALUES ('150', 150000);
INSERT INTO Monitored_Farms3 (area, budget) VALUES ('120', 120000);
INSERT INTO Monitored_Farms3 (area, budget) VALUES ('200', 200000);
INSERT INTO Monitored_Farms3 (area, budget) VALUES ('180', 180000);
```

```
INSERT INTO Farming_Equipment1 (tool_id, tool_name, manufacturer, farm_id) VALUES
('TOOL001', 'Tractor', 'John Deere', 'FARM001');
INSERT INTO Farming_Equipment1 (tool_id, tool_name, manufacturer, farm_id) VALUES
('TOOL002', 'Plough', 'Caterpillar', 'FARM002');
INSERT INTO Farming_Equipment1 (tool_id, tool_name, manufacturer, farm_id) VALUES
('TOOL003', 'Harvester', 'Kubota', 'FARM003');
INSERT INTO Farming_Equipment1 (tool_id, tool_name, manufacturer, farm_id) VALUES
('TOOL004', 'Sprayer', 'Case IH', 'FARM004');
INSERT INTO Farming_Equipment1 (tool_id, tool_name, manufacturer, farm_id) VALUES
('TOOL005', 'Seeder', 'New Holland', 'FARM005');


INSERT INTO Farming_Equipment2 (tool_usage, manufacturer, price) VALUES ('Tilling', 'John
Deere', 25000.00);
INSERT INTO Farming_Equipment2 (tool_usage, manufacturer, price) VALUES  ('Ploughing',
'Caterpillar', 15000.00);
INSERT INTO Farming_Equipment2 (tool_usage, manufacturer, price) VALUES ('Harvesting',
'Kubota', 35000.00);
INSERT INTO Farming_Equipment2 (tool_usage, manufacturer, price) VALUES ('Spraying',
'Case IH', 20000.00);
INSERT INTO Farming_Equipment2 (tool_usage, manufacturer, price) VALUES ('Seeding', 'New
Holland', 18000.00);

INSERT INTO Farming_Equipment3 (tool_name, tool_usage) VALUES ('Tractor', 'Tilling');
INSERT INTO Farming_Equipment3 (tool_name, tool_usage) VALUES ('Plough', 'Ploughing');
INSERT INTO Farming_Equipment3 (tool_name, tool_usage) VALUES ('Harvester',
'Harvesting');
INSERT INTO Farming_Equipment3 (tool_name, tool_usage) VALUES ('Sprayer', 'Spraying');
INSERT INTO Farming_Equipment3 (tool_name, tool_usage) VALUES ('Seeder', 'Seeding');

INSERT INTO Working_Farmers1 (farmer_id, farmer_name, farmer_phone_number,
farmer_type, farm_id, farmer_hours) VALUES
('FARMER001', 'Tom Hardy', '555-0201', 'Junior Farmer', 'FARM001', 40);
INSERT INTO Working_Farmers1 (farmer_id, farmer_name, farmer_phone_number,
farmer_type, farm_id, farmer_hours) VALUES('FARMER002', 'Natalie Portman', '555-0202',
'Senior Farmer', 'FARM002', 45);
INSERT INTO Working_Farmers1 (farmer_id, farmer_name, farmer_phone_number,
farmer_type, farm_id, farmer_hours) VALUES('FARMER003', 'Chris Evans', '555-0203', 'Junior
Farmer', 'FARM003', 38);
INSERT INTO Working_Farmers1 (farmer_id, farmer_name, farmer_phone_number,
farmer_type, farm_id, farmer_hours) VALUES('FARMER004', 'Scarlett Johansson', '555-0204',
'Supervisor', 'FARM004', 50);
INSERT INTO Working_Farmers1 (farmer_id, farmer_name, farmer_phone_number,
farmer_type, farm_id, farmer_hours) VALUES ('FARMER005', 'Mark Ruffalo', '555-0205', 'Senior
Farmer', 'FARM005', 42);
```

```
INSERT INTO Working_Farmers2 (farmer_type, farm_id, hourly_wage) VALUE ('Junior Farmer', 'FARM001', 20);
INSERT INTO Working_Farmers2 (farmer_type, farm_id, hourly_wage) VALUES ('Senior Farmer', 'FARM002', 25);
INSERT INTO Working_Farmers2 (farmer_type, farm_id, hourly_wage) VALUES ('Junior Farmer', 'FARM003', 20);
INSERT INTO Working_Farmers2 (farmer_type, farm_id, hourly_wage) VALUES ('Supervisor', 'FARM004', 30);
INSERT INTO Working_Farmers2 (farmer_type, farm_id, hourly_wage) VALUES ('Senior Farmer', 'FARM005', 25);

INSERT INTO FruitBasket (fruit_name, farm_id, quantity) VALUES ('Apples', 'FARM001', 500);
INSERT INTO FruitBasket (fruit_name, farm_id, quantity) VALUES ('Oranges', 'FARM002', 300);
INSERT INTO FruitBasket (fruit_name, farm_id, quantity) VALUES ('Grapes', 'FARM003', 450);
INSERT INTO FruitBasket (fruit_name, farm_id, quantity) VALUES ('Bananas', 'FARM004', 600),
INSERT INTO FruitBasket (fruit_name, farm_id, quantity) VALUES ('Strawberries', 'FARM005', 400);

INSERT INTO Produced1 (crop_id, farm_id, startDate, area, crop_name) VALUES ('CROP001', 'FARM001', '2023-03-01', '100', 'Wheat');
INSERT INTO Produced1 (crop_id, farm_id, startDate, area, crop_name) VALUES('CROP002', 'FARM002', '2023-04-15', '150', 'Corn');
INSERT INTO Produced1 (crop_id, farm_id, startDate, area, crop_name) VALUES('CROP003', 'FARM003', '2023-05-20', '120', 'Soybeans');
INSERT INTO Produced1 (crop_id, farm_id, startDate, area, crop_name) VALUES('CROP004', 'FARM004', '2023-06-10', '200', 'Rice');
INSERT INTO Produced1 (crop_id, farm_id, startDate, area, crop_name) VALUES('CROP005', 'FARM005', '2023-07-05', '180', 'Barley');

INSERT INTO Produced2 (area, quantity) VALUES ('100', 10000);
INSERT INTO Produced2 (area, quantity) VALUES ('150', 15000);
INSERT INTO Produced2 (area, quantity) VALUES ('120', 12000);
INSERT INTO Produced2 (area, quantity) VALUES ('200', 20000);
INSERT INTO Produced2 (area, quantity) VALUES ('180', 18000);

INSERT INTO Produced3 (crop_name, startDate, crop_status) VALUES ('Wheat', '2023-03-01', 'Harvested');
INSERT INTO Produced3 (crop_name, startDate, crop_status) VALUES ('Corn', '2023-04-15', 'Growing');
INSERT INTO Produced3 (crop_name, startDate, crop_status) VALUES ('Soybeans', '2023-05-20', 'Planted');
INSERT INTO Produced3 (crop_name, startDate, crop_status) VALUES ('Rice', '2023-06-10', 'Harvested')l
INSERT INTO Produced3 (crop_name, startDate, crop_status) VALUE ('Barley', '2023-07-05', 'Growing');

INSERT INTO Crops1 (crop_id, crop_name, crop_status) VALUES ('CROP001', 'Wheat', 'Harvested');
INSERT INTO Crops1 (crop_id, crop_name, crop_status) VALUES ('CROP002', 'Corn', 'Growing');
```

```
INSERT INTO Crops1 (crop_id, crop_name, crop_status) VALUE ('CROP003', 'Soybeans',
'Planted');
INSERT INTO Crops1 (crop_id, crop_name, crop_status) VALUES ('CROP004', 'Rice',
'Harvested'),
INSERT INTO Crops1 (crop_id, crop_name, crop_status) VALUES ('CROP005', 'Barley',
'Growing');

INSERT INTO Crops2 (crop_name, growth_duration) VALUES ('Wheat', '120 days'),
INSERT INTO Crops2 (crop_name, growth_duration) VALUES ('Corn', '90 days'),
INSERT INTO Crops2 (crop_name, growth_duration) VALUES ('Soybeans', '100 days'),
INSERT INTO Crops2 (crop_name, growth_duration) VALUES ('Rice', '150 days'),
INSERT INTO Crops2 (crop_name, growth_duration) VALUES ('Barley', '70 days');

INSERT INTO Soil_Type1 (soil_id, soil_name, texture) VALUES ('SOIL001', 'Loamy', 'Smooth')
INSERT INTO Soil_Type1 (soil_id, soil_name, texture) VALUES ('SOIL002', 'Clay', 'Sticky')
INSERT INTO Soil_Type1 (soil_id, soil_name, texture) VALUES ('SOIL003', 'Sandy', 'Grainy')
INSERT INTO Soil_Type1 (soil_id, soil_name, texture) VALUES ('SOIL004', 'Peaty', 'Spongy')
INSERT INTO Soil_Type1 (soil_id, soil_name, texture) VALUES ('SOIL005', 'Silty', 'Silky')



INSERT INTO Soil_Type2 (texture, pH) VALUES ('Smooth', 6.5);
INSERT INTO Soil_Type2 (texture, pH) VALUES ('Sticky', 7.0);
INSERT INTO Soil_Type2 (texture, pH) VALUES ('Grainy', 5.5);
INSERT INTO Soil_Type2 (texture, pH) VALUES ('Spongy', 5.8);
INSERT INTO Soil_Type2 (texture, pH) VALUES ('Silky', 6.2);


INSERT INTO Farm_Soil (farm_id, soil_id) VALUES ('FARM001', 'SOIL001');
INSERT INTO Farm_Soil (farm_id, soil_id) VALUES ('FARM002', 'SOIL002');
INSERT INTO Farm_Soil (farm_id, soil_id) VALUES ('FARM004', 'SOIL004');
INSERT INTO Farm_Soil (farm_id, soil_id) VALUES ('FARM003', 'SOIL003');
INSERT INTO Farm_Soil (farm_id, soil_id) VALUES ('FARM005', 'SOIL005');


INSERT INTO Crop_Soil (crop_id, soil_id) VALUES ('CROP001', 'SOIL001');
INSERT INTO Crop_Soil (crop_id, soil_id) VALUES ('CROP002', 'SOIL002');
INSERT INTO Crop_Soil (crop_id, soil_id) VALUES ('CROP003', 'SOIL003');
INSERT INTO Crop_Soil (crop_id, soil_id) VALUES ('CROP004', 'SOIL004');
INSERT INTO Crop_Soil (crop_id, soil_id) VALUES ('CROP005', 'SOIL005');


INSERT INTO Sold1 (channel_id, crop_id, crop_name) VALUES ('CH001', 'CROP001', 'Wheat');
INSERT INTO Sold1 (channel_id, crop_id, crop_name) VALUES ('CH002', 'CROP002', 'Corn');
INSERT INTO Sold1 (channel_id, crop_id, crop_name) VALUES ('CH003', 'CROP003',
'Soybeans');
INSERT INTO Sold1 (channel_id, crop_id, crop_name) VALUES ('CH004', 'CROP004', 'Rice');
INSERT INTO Sold1 (channel_id, crop_id, crop_name) VALUES ('CH005', 'CROP005', 'Barley');
```

```
INSERT INTO Sold2 (channel_id, crop_name, quantity, price) VALUES ('CH001', 'Wheat', 200, 10.50);
INSERT INTO Sold2 (channel_id, crop_name, quantity, price) VALUES ('CH002', 'Corn', 150, 7.25);
INSERT INTO Sold2 (channel_id, crop_name, quantity, price) VALUES ('CH003', 'Soybeans', 100, 8.75);
INSERT INTO Sold2 (channel_id, crop_name, quantity, price) VALUES ('CH004', 'Rice', 250, 9.00);
INSERT INTO Sold2 (channel_id, crop_name, quantity, price) VALUES ('CH005', 'Barley', 180, 6.50);


INSERT INTO Distribution_Channel (channel_id, channel_name) VALUES ('CH001', 'Local Market');
INSERT INTO Distribution_Channel (channel_id, channel_name) VALUES ('CH002', 'Export');
INSERT INTO Distribution_Channel (channel_id, channel_name) VALUES ('CH003', 'Farmers Market');
INSERT INTO Distribution_Channel (channel_id, channel_name) VALUES ('CH004', 'Online Sales');
INSERT INTO Distribution_Channel (channel_id, channel_name) VALUES ('CH005', 'Direct to Consumer');


INSERT INTO Online (channel_id, url) VALUES ('CH004', 'www.onlineshoes.com');
INSERT INTO Online (channel_id, url) VALUES ('CH005', 'www.directtoconsumer.com');
INSERT INTO Online (channel_id, url) VALUES ('ON001', 'www.agriweb.com');
INSERT INTO Online (channel_id, url) VALUES ('ON002', 'www.farmgoods.org');
INSERT INTO Online (channel_id, url) VALUES ('ON003', 'www.cropsonline.net');


INSERT INTO Offline (channel_id, address) VALUES ('CH001', '123 Local Market St');
INSERT INTO Offline (channel_id, address) VALUES ('CH002', '456 Export Blvd');
INSERT INTO Offline (channel_id, address) VALUES ('CH003', '789 Farmers Market Ave');
INSERT INTO Offline (channel_id, address) VALUES ('OF004', '404 Harvest Home Rd');
INSERT INTO Offline (channel_id, address) VALUES ('OF005', '505 Tractor Trail');
```