University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: February 26, 2024

Group Number: 9

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Jacky Feng	45105616	k4a1k	Jfeng14@student.ubc.ca
Safiullah Kaleem	57876609	s9r9w	skaleem@student.ubc.ca
Mohammad Fakhir	11939493	i2r8j	mfakhir@student.ubc.ca

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

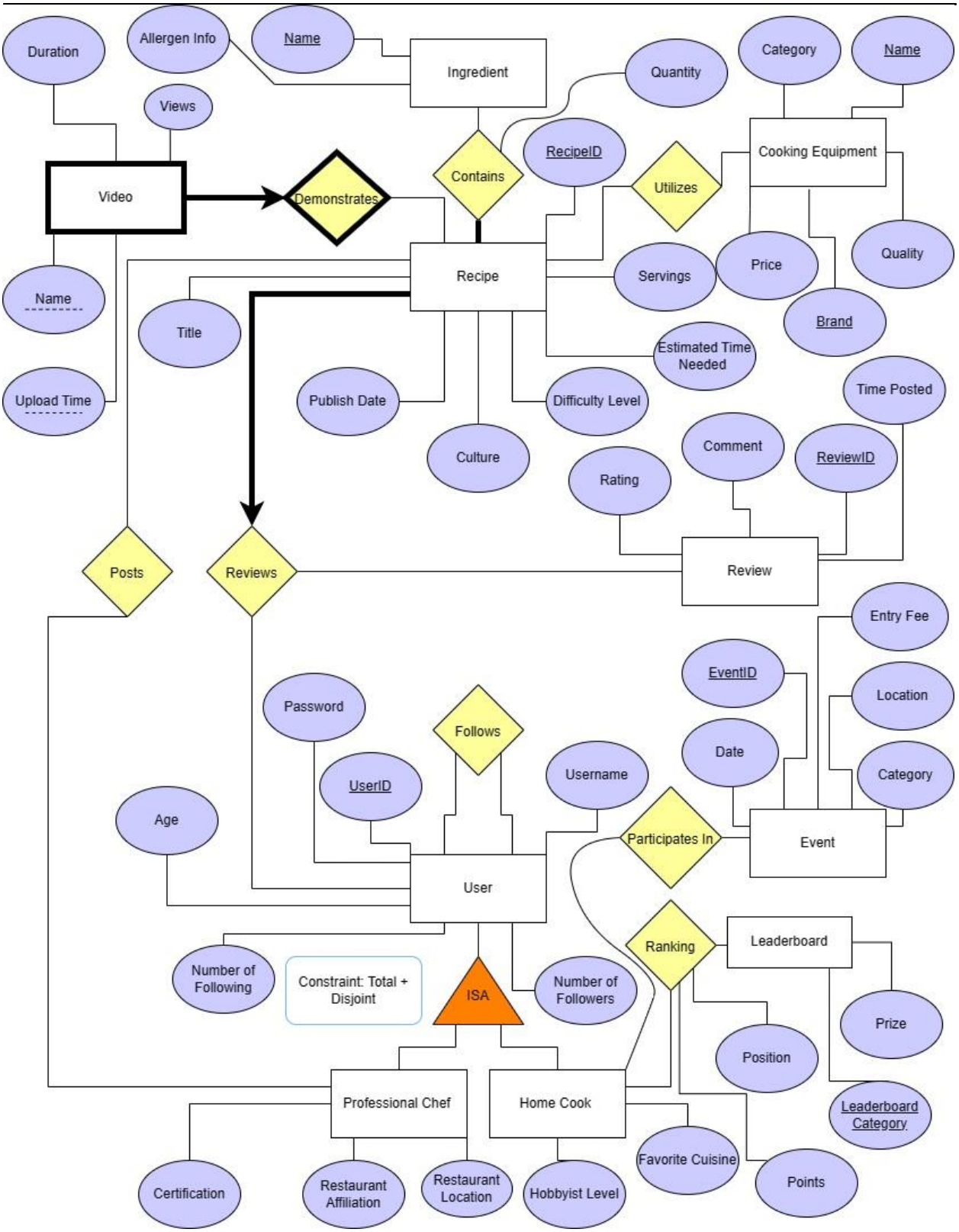
University of British Columbia, Vancouver
Department of Computer Science

Summary of Project:

The project is a community-centric cooking recipe platform, allowing users to access recipes from professional chefs, attend competitive events to earn points, and review recipes. Real-life applications include assisting home cooks in meal planning, chefs sharing signature recipes, and users with dietary needs finding suitable recipes. The database functionality includes analyzing user preferences and event popularity, while the application platform uses JavaScript, HTML, and CSS for the front-end, PHP for the back-end, and Oracle for the Database Management System (DBMS).

The ER diagram is on the following page.

University of British Columbia, Vancouver
Department of Computer Science



University of British Columbia, Vancouver
Department of Computer Science

SUMMARY OF CHANGES

The database schema underwent several changes based on feedback and requirements that emerged as we discussed our project with our mentor TA. Firstly, the relationship set (Demonstrates) in the weak entity was bolded to match the conventions used in the class as it wasn't previously (pointed out by the TA). To address the mentor's concerns about the primary key, the Cooking Equipment entity was revised to include more detailed attributes, ensuring a more meaningful representation of Cooking Equipment. Furthermore, the total count of entity types was increased to match the minimum requirements as pointed out in the rubric (as pointed out by our mentor TA). Additionally, various adjustments were made to enhance data representation and functionality. The Review relationship between Recipe and User was modified from many-to-many to one-to-many, to model realistic cases in the real world (one recipe can have many reviews, but one review cannot be linked to many recipes). Price was incorporated as an attribute into the Cooking Equipment entity, while Quantity was relocated as an attribute of the Contains relationship for improved data modeling. Name and Upload Time attributes replaced VideoID in the Video entity, enhancing data clarity. A ternary relationship was introduced among Recipe, User, and a new entity called Review, featuring attributes such as Rating, Comment, and ReviewID, for more comprehensive data organization. Professional Chef now includes a Restaurant Location attribute, enriching entity details, while the Event entity was given an Entry Fee attribute for enhanced event management functionality.

SCHEMAS

HomeCook (Userid: integer (10), UserName: Varchar (30) NOT NULL UNIQUE, Password: Binary (64) NOT NULL, Number of Followers: Integer (10) DEFAULT 0, Number of Following: Integer (10) DEFAULT 0, Age Integer (3) NOT NULL, HobbyistLevel: Varchar (30) DEFAULT "Amateur", FavoriteCuisine: Varchar (30))

Professional Chef (Userid: integer (10), UserName: Varchar (30) NOT NULL UNIQUE, Password: Binary (64) NOT NULL, Number of Followers: Integer (10) DEFAULT 0, Number of Following: Integer (10) DEFAULT 0, Age Integer (3) NOT NULL, Restaurant Affiliation: Varchar (30), Restaurant Location: Varchar (30), Certification: Varchar (30))

User (Userid: integer (10), UserName: Varchar (30) NOT NULL UNIQUE, Password: Binary (64) NOT NULL, Number of Followers: Integer (10) DEFAULT 0, Number of Following: Integer (10) DEFAULT 0, Age Integer (3))

Note: Even though its total + disjoint because of the special relationship we will have a table for user.

Leaderboard (Leaderboard Category: Varchar (30), Prize: Varchar (30))

Event (EventID: Integer (10), Date: Date (20) NOT NULL, Location: Varchar (30) NOT NULL, Category: Varchar (30) NOT NULL, Entry Fee: Integer (5))

Review (ReviewID: Integer (10), Time Posted: Date NOT NULL, Rating: Integer (5) DEFAULT 0, Comment: Varchar (30))

Cooking Equipment (Name: Varchar (30), Brand: Varchar (20) NOT NULL, Category: Varchar (20) NOT NULL, Price: Integer (5), Quality: Varchar (20))

Recipe (RecipeID: Integer (10), **ReviewID**: integer (10), **UserID**: integer (10) NOT NULL, Publish Date: Date (20) NOT NULL, Culture: Varchar (20) NOT NULL, Difficulty: Varchar (20) NOT NULL, Estimated Time: Integer (10) NOT NULL, Servings: Integer (20), Title: Varchar (20) NOT NULL)

Video (**RecipeID**: Integer (10), Name: Varchar (30), Upload time: Date (20), Views: Integer (20) DEFAULT 0, Duration: Integer (10) NOT NULL)

Ingredient (Name: Varchar (20), Allergen Info: Varchar (30) DEFAULT “None”)

Follows (UserID1: Integer (10, **UserID2:** Integer (10))

Note: USERID1 means follower id and USERID2 means follows id

Participates (Userid: integer (10), **EventID:** Integer (10))

Ranking (Userid: integer (10), **Leaderboard Category:** Varchar (30), Position: Integer (10), Points: Integer (20))

Utilizes (RecipeID: Integer (10), **Name:** Varchar (30) NOT NULL, **Brand:** Varchar (20))

Posts (RecipeID: Integer (10), **Userid: integer (10)**)

For contains I can't model a schema right now since it needs assertion since its total participation for many. Assertions haven't been covered in class.

University of British Columbia, Vancouver
Department of Computer Science

FUNCTIONAL DEPENDENCIES

Ranking:

- UserID, Leaderboard Category → Position, Points
- Points → Position

Participate In:

- No Non-Trivial FDS, since there are only primary keys in this relationship

Follows:

- No Non-Trivial FDS, since there are only primary keys in this relationship

Reviews:

- No Non-Trivial FDS, since there are only primary keys in this relationship

Posts:

- No Non-Trivial FDS, since there are only primary keys in this relationship

Contains:

- Name, RecipeID → Quantity

There is no schema written for contains however, we are assuming the contains relationship would look something for this

Utilizes:

- No Non-Trivial FDS, since there are only primary keys in this relationship

Video:

- No Non-Trivial FDS, since there are only primary keys in this relationship

User:

- UserID →Username, Number of Followers, Password, Number of Following, Age
- Username, Password → Number of Followers, Password, Number of Following, Age

Home Cook:

- UserID → Username, Number of Followers, Number of Following, Password, Hobbyist Level, Favourite Cuisine, Age
- Username, Password → Number of Followers, Number of Following, Hobbyist Level, Favourite Cuisine, Age

- Number of followers → Hobbyist Level

Professional Chef:

- UserID → Number of Followers, Username, Restaurant Affiliation, Certification, Restaurant location, Age
- Username, Password → Number of Followers, Number of Following, Age, Restaurant Affiliation, Certification, Restaurant location
- Restaurant Affiliation, Restaurant Location → Certification

Event:

- EventID → Date, Location, Category, Entry Fee
- Entry Fee, Category → Location
- Date, Location → Category

Leaderboard:

- Leaderboard Category -> Prize

Review:

- ReviewID → Time Posted, Comment, Rating
- Time Posted, Comment → Rating

Recipe:

- RecipeID → Title, Publish Date, Culture, Difficulty Level, Servings, Estimated Time, ReviewID, UserID
- Publish Date, Title → Culture, Difficulty Level, Servings
- Estimated Time → Difficulty Level

Video:

- Name, Upload Time, RecipeID → Duration, Views

Cooking Equipment:

- Name, Brand → Category, Price, Quality
- Price, Category, Quality → Brand

Ingredient:

- Name → Allergen Info

NORMALIZATION

We will use the lossless-join BCNF decomposition algorithm to ensure that every functional dependency is in BCNF:

User (UserID, UserName, Number of Followers, Number of Following, Password, Age)

1.) UserID →Username, Number of Followers, Password, Age, Number of Following
2.) Username, Password → Number of Followers, Number of Following, Age

We can decompose by (Username, Password → Number of Following, Password, Age, Number of Followers) to be:

User 1 (Username, Password, Number of Followers, Number of Following, Age)

User 2 (Username, Password, UserID)

Home Cook (UserID, Password, Username, Number of Followers, Hobbyist Level, Favorite Cuisine, Number of Following, Age)

1.) UserID → Username, Number of Followers, Number of Following, Password, Hobbyist Level, Number of Following, Age, Favorite Cuisine
2.) Username, Password → Number of Followers, Number of Following, Password, Hobbyist Level, Age, Favorite Cuisine
3.) Number of followers → Hobbyist Level

We can decompose by (Username, Password → Number of Followers, Number of Following, Password, Hobbyist Level, Age, Favorite Cuisine) to be:

Home Cook 1 (Username, Password, Number of Followers, Number of Following, Password, Hobbyist Level, Age, Favorite Cuisine)
Home Cook 2 (Username, Password, **UserID**)

We can decompose further by (Number of Followers → Hobbyist Level) to be:

Home Cook 1 (Username, Password, Number of Followers, Number of Following, Password, Age, Favorite Cuisine)

Home Cook 2 (Username, Password, **UserID**)

Home Cook 3 (Number of Followers, Hobbyist Level)

Professional Chef (UserID, Password, Username, Number of Followers, Number of Following, Restaurant Affiliation, Restaurant Location, Age)

1.) UserID → Number of Followers, Username, Restaurant Affiliation, Certification, Restaurant Location, Age, Number of Following, Certification
2.) Username, Password → Number of Followers, Number of Following, Restaurant Affiliation, Restaurant Location, Age, Certification
3.) Restaurant Affiliation, Restaurant Location → Certification

We can decompose by (Username, Password → Number of Followers, Number of Following, Restaurant Affiliation, Restaurant Location, Age, Certification) to be:

Professional Chef 1 (Username, Password, Number of Followers, Number of Following, Restaurant Affiliation, Restaurant Location, Age, Certification)

Professional Chef 2 (Username, Password, **UserID**)

We can decompose further by (Restaurant Affiliation, Restaurant Location → Certification) to be:

Professional Chef 1 (Username, Password, Number of Followers, Number of Following, Age, Restaurant Affiliation, Restaurant Location)

Professional Chef 2 (Username, Password, **UserID**)

Professional Chef 3 (Restaurant Affiliation, Restaurant Location, Certification)

Event (EventID, Entry Fee, Location, Category, Date)

1.) EventID → Date, Location, Category, Entry Fee
2.) Entry Fee, Category → Location
3.) Date, Location → Category

We can decompose by (Entry Fee, Category → Location) to be:

Event 1 (Entry Fee, Category, Location)

Event 2 (Category, Entry Fee, EventID, Date)

Review (ReviewID, Time Posted, Comment, Rating)

1.) ReviewID → Time Posted, Comment, Rating
2.) Time Posted, Comment → Rating

We can decompose by (Time Posted, Comment → Rating) to be:

Review 1 (Time Posted, Comment, Rating)

Review 2 (Time Posted, Comment, ReviewID)

Recipe (RecipeID, Title, Publish Date, Culture, Difficulty Level, Servings, Estimated time, **UserId, ReviewId**)

1.) RecipeID → Title, Publish Date, Culture, Difficulty Level, Servings, Estimated Time, **UserId, ReviewId**
2.) Publish Date, Title → Culture, Difficulty Level, Servings
3.) Estimated time → Difficulty Level

We can decompose by (Publish Date, Title → Culture, Difficulty Level, Servings) to be:

Recipe 1 (Publish Date, Title, Culture, Difficulty Level, Servings)

Recipe 2 (Publish Date, Title, Estimated Time, **UserId, ReviewId,** RecipeID)

Cooking Equipment (Name, Brand, Category, Price, Quality)

1.) Name, Brand → Category, Price, Quality
2.) Price, Category, Quality → Brand

We can decompose by (Price, Category, Quality → Brand) to be:

Cooking Equipment 1 (Price, Category, Quality, Brand)

Cooking Equipment 2 (Price, Category, Quality, Name)

Ranking (UserID, Leaderboard Category, Position, Prize)

1.) UserID, Leaderboard Category → Position, Points
2.) Points → Position

We can decompose by (Points → Position) to be:

Ranking 1 (Points, Position)

Ranking 2 (**UserID**, **Leaderboard Category**, Points)

All other Functional dependencies are in BCNF.

Final Table

Homecook1 (UserName: Varchar (30), Password: Binary (64), Number of Followers: Integer (10) DEFAULT 0, Number of Following: Integer (10) DEFAULT 0, Age Integer (3) NOT NULL, FavoriteCuisine: Varchar (30))

Homecook2 (**Userid:** integer (10), UserName: Varchar (30) NOT NULL UNIQUE, Password: Binary (64) NOT NULL)

Homecook3 (Number of Followers: Integer (10), HobbyistLevel: Varchar (30) DEFAULT "Amateur")

Professional Chef 1 (UserName: Varchar (30), Password: Binary (64) NOT NULL, Number of Followers: Integer (10) DEFAULT 0, Number of Following: Integer (10) DEFAULT 0, Age Integer (3) NOT NULL, Restaurant Affiliation: Varchar (30), Restaurant Location: Varchar (30))

Professional Chef 2 (**Userid:** integer (10), UserName: Varchar (30) NOT NULL UNIQUE, Password: Binary (64) NOT NULL)

Professional Chef 3 (Restaurant Affiliation: Varchar (30), Restaurant Location: Varchar (30), Certification: Varchar (30))

User 1 (UserName: Varchar (30), Password: Binary (64) NOT NULL, Number of Followers: Integer (10) DEFAULT 0, Number of Following: Integer (10) DEFAULT 0, Age Integer (3) NOT NULL)

User 2 (Userid**:** integer (10), UserName: Varchar (30) NOT NULL UNIQUE, Password: Binary (64) NOT NULL)

Leaderboard (Leaderboard Category: Varchar (30), Prize: Varchar (30))

Event1 (Category: Varchar (30), Entry Fee: Integer (5), Location: Varchar (30) NOT NULL)

Event2 (EventID: Integer (10), Date: Date (20) NOT NULL, Category: Varchar (30) NOT NULL, Entry Fee: Integer (5))

University of British Columbia, Vancouver
Department of Computer Science

Review1 (Time Posted: Date, Comment: Varchar (30), Rating: Integer (5) DEFAULT 0)

Review2 (ReviewID: Integer (10), Time Posted: Date NOT NULL, Comment: Varchar (30))

Cooking Equipment1 (Brand: Varchar (20) NOT NULL, Category: Varchar (20), Price: Integer (5), Quality: Varchar (20))

Cooking Equipment2 (Name: Varchar (30), Category: Varchar (20) NOT NULL, Price: Integer (5), Quality: Varchar (20))

Recipe1 (Publish Date: Date (20), Title: Varchar (20), Culture: Varchar (20) NOT NULL, Difficulty: Varchar (20) NOT NULL, Servings: Integer)

Recipe2 (RecipeID: Integer (10), **ReviewID**: integer (10), **UserID**: integer (10) NOT NULL, Publish Date: Date (20) NOT NULL, Estimated Time: Integer (10) NOT NULL, Title: Varchar (20) NOT NULL)

Video (**RecipeID**: Integer (10), Name: Varchar (30), Upload time: Date (20), Views: Integer (20) DEFAULT 0, Duration: Integer (10) NOT NULL)

Ingredient (Name: Varchar (20), Allergen Info: Varchar (30) DEFAULT "None")

Follows (UserID1: Integer (10), **UserID2:** Integer (10))

Note: USERID1 means follower id and USERID2 means follows id

Participates (Userid: integer (10), **EventID:** Integer (10))

Ranking1 (Points: Integer (20), Position: Integer (10))

Ranking2 (Userid: integer (10), **Leaderboard Category:** Varchar (30), Points: Integer (20))

Utilizes (RecipeID: Integer (10), **Name:** Varchar (30), **Brand:** Varchar (20))

Posts (RecipeID: Integer (10), **UserID:** integer (10))

For contains I can't model a schema right now since it needs assertion since its total participation for many. Assertions haven't been covered in class.

DDL STATEMENTS:

- User 1 table:

```
CREATE TABLE User1 (
  Number of Followers: Integer (10) DEFAULT 0,
  Number of Following: Integer (10) DEFAULT 0,
   Age Integer (3) NOT NULL,
   Username VARCHAR (30),
   Password BINARY (64),
   PRIMARY KEY (Username, Password)
);
```

- User 2 table:

```
CREATE TABLE User2 (
   Username VARCHAR (30) NOT NULL UNIQUE,
   Password BINARY (64) NOT NULL,
   UserID INTEGER (10),
   PRIMARY KEY (UserID),
   FOREIGN KEY (UserID) REFERENCES User
         ON DELETE CASCASE
);
```

- Home Cook 1 table:

```
CREATE TABLE HomeCook1 (
   FavouriteCuisine VARCHAR (30),
   Number of Followers: Integer (10) DEFAULT 0,
  Number of Following: Integer (10) DEFAULT 0,
   Age Integer (3) NOT NULL
   Username VARCHAR (30),
   Password BINARY (64),
   PRIMARY KEY (Username, Password)
);
```

- Home Cook 2 table

```
CREATE TABLE HomeCook2 (
   Username VARCHAR (30) NOT NULL UNIQUE,
   Password BINARY (64) NOT NULL,
   UserID INTEGER (10),
   PRIMARY KEY (UserID),
   FOREIGN KEY (UserID) REFERENCES User
```

```
        ON DELETE CASCASE
);

- Home Cook 3 table

CREATE TABLE HomeCook3 (
   Number of Followers Integer (10) DEFAULT 0,
   HobbyistLevel VARCHAR (30) DEFAULT “Amateur”
   PRIMARY KEY (Number of Followers)
);


- Professional chef 1 table:

CREATE TABLE ProfessionalChef1 (
   Restaurant Affiliation VARCHAR (30),
   Restaurant Location VARCHAR (30)),
   Number of Followers Integer (10) DEFAULT 0,
   Number of Following Integer (10) DEFAULT 0,
   Age Integer (3) NOT NULL,
   Username VARCHAR (30),
   Password BINARY (64),
   PRIMARY KEY (Username, Password)
);


- Professional chef 2 table

CREATE TABLE ProfessionalChef2 (
   Username VARCHAR (30) NOT NULL UNIQUE,
   Password BINARY (64)  NOT NULL,
   UserID INTEGER (10),
   PRIMARY KEY (UserID),
   FOREIGN KEY (UserID) REFERENCES User
        ON DELETE CASCASE
);

- Professional Chef 3 table

CREATE TABLE ProfessionalChef3 (
  Restaurant Affiliation VARCHAR (30),
  Restaurant Location VARCHAR (30),
  Certification: VARCHAR (30),
```

```
  PRIMARY KEY (Restaurant Affiliation, Restaurant Location)
);

- Leaderboard table

CREATE TABLE Leaderboard (
    Leaderboard Category VARCHAR (30) NOT NULL UNIQUE,
    Prize VARCHAR (30),
    PRIMARY KEY (Leaderboard Category)
);

- Event 1 table

CREATE TABLE Event1 (
    Category VARCHAR (30),
    Entry Fee INTEGER (5),
    Location VARCHAR (30) NOT NULL,
    PRIMARY KEY (Category, Entry Fee)
);

- Event 2 table

CREATE TABLE Event2 (
    Category VARCHAR (30) NOT NULL,
    EntryFee INTEGER (5),
    EventID VARCHAR (30),
    Date VARCHAR (30),
    PRIMARY KEY (EventID)
);


- Review 1 table

CREATE TABLE Review1 (
    TimePosted DATE,
    Comment VARCHAR (30),
    Rating INTEGER (5) DEFAULT 0,
    PRIMARY KEY (TimePosted, Comment)
);

- Review 2 table

CREATE TABLE Review2 (
```

```
    TimePosted DATE NOT NULL,
    Comment VARCHAR (30),
    ReviewID INTEGER (30),
    PRIMARY KEY (ReviewID),
);
```

- Cooking Equipment 1 table

```
CREATE TABLE CookingEquipment1 (
    Price INTEGER (5),
    Category VARCHAR (20),
    Quality VARCHAR (30),
    Brand VAR CHAR (20) NOT NULL,
    PRIMARY KEY (Price, Category, Quality)
);
```

- Cooking Equipment 2 table

```
CREATE TABLE CookingEquipment2 (
    Price INTEGER (5),
    Category VARCHAR (30),
    Quality VARCHAR (30),
    Name VARCHAR (30),
    PRIMARY KEY (Price, Category, Quality, Name)
);
```

- Recipe 1 table

```
CREATE TABLE Recipe1 (
    Publish Date DATE (20),
    Title: VARCHAR (20),
    Culture: VARCHAR (20) NOT NULL,
    Difficulty: VARCHAR (20) NOT NULL,
    Servings: INTEGER (20),
    PRIMARY KEY (Publish Date, Title)
);
```

- Recipe 2 Table

```
CREATE TABLE Recipe2 (
    RecipeID INTEGER (10),
    ReviewID INTEGER (10),
    UserID INTEGER (10) NOT NULL,
```

```
    Publish Date DATE (20) NOT NULL,
    Estimated Time INTEGER (10) NOT NULL,
    Title VARCHAR (20) NOT NULL
    PRIMARY KEY (RecipeID),
    FOREIGN KEY (ReviewID) REFERENCES Review
        ON DELETE CASCASE,
    FOREIGN KEY (UserID) REFERENCES User
        ON DELETE CASCASE,

);
```

- Video table

```
CREATE TABLE Video (
    Name VARCHAR (30),
    Upload Time DATE (20),
    RecipeID INTEGER (30),
    Duration INTEGER (10) NOT NULL,
    Views VARCHAR (30) DEFAULT 0,
    PRIMARY KEY (Name, UploadTime, RecipeID),
    FOREIGN KEY (RecipeID) REFERENCES Recipe
          ON DELETE CASCASE
);
```

- Ingredient table

```
CREATE TABLE Ingredient (
    Name VARCHAR (30),
    Allergen Info VARCHAR (30) DEFAULT “None”,
    PRIMARY KEY (Name)
);
```

- Follows table

```
CREATE TABLE Follows (
    UserID1 INTEGER (10),
    UserID2 INTEGER (10),
    PRIMARY KEY (UserID1, UserID2),
    FOREIGN KEY (UserID1) REFERENCES User1
            ON DELETE CASCADE,
    FOREIGN KEY (UserID2) REFERENCES User2
            ON DELETE CASCADE
);
```

- Participate table

```
CREATE TABLE Participates (
    UserID: INTEGER (10),
    EventID:  INTEGER (10),
    PRIMARY KEY (Userid, EventID),
    FOREIGN KEY (Userid) REFERENCES User
            ON DELETE CASCADE,
    FOREIGN KEY (EventID) REFERENCES Event
            ON DELETE CASCADE
);
```

- Ranking 1 table

```
CREATE TABLE Ranking1 (
    Points: INTEGER (20),
    Position: INTEGER (10)),
    PRIMARY KEY (Points)
);
```

- Ranking 2 table

```
CREATE TABLE Ranking2 (
    UserID INTEGER (10),
    Leaderboard Category VARCHAR (30),
    Points INTEGER (20),
    PRIMARY KEY (UserID, Leaderboard Category),
    FOREIGN KEY (UserID) REFERENCES User
            ON DELETE CASCADE,
    FOREIGN KEY (Leaderboard Category) REFERENCES Leaderboard
            ON DELETE CASCADE
);
```

- Utilizes table

```
CREATE TABLE Utilizes (
    RecipeID INTEGER (10),
    Name VARCHAR (10),
    Brand VARCHAR (10),
    PRIMARY KEY (RecipeID, Brand, Name),
    FOREIGN KEY (RecipeID) REFERENCES Recipe
            ON DELETE CASCADE,
```

```
    FOREIGN KEY (Brand, Name) REFERENCES Cooking Equipment
            ON DELETE CASCADE
);

- Post table

CREATE TABLE Posts (
    RecipeID INTEGER (10),
    UserID INTEGER (10),
    PRIMARY KEY (RecipeID, UserID),
    FOREIGN KEY (RecipeID) REFERENCES Recipe
            ON DELETE CASCADE
    FOREIGN KEY (UserID) REFERENCES User
            ON DELETE CASCADE
);
```

Data Insert Statement

Here are the Insert statements:

```
-- Home Cook1
INSERT INTO HomeCook1
VALUES
('CookMaster',
'1000011010000101111000010100101000111100100010000110101010110110000', 500, 100, 28,
'Italian'),
('BakerExtraordinaire',
''1000011010000101111000010100101000111100100010000110101010110110000'', 700, 150,
35, 'French'),
('GrillKing', ''1000011010000101111000010100101000111100100010000110101010110110000'',
300, 50, 40, 'Barbecue'),
('SweetToothChef',
''1000011010000101111000010100101000111100100010000110101010110110000'', 400, 200,
30, 'Desserts'),
('SpiceExplorer',
''1000011010000101111000010100101000111100100010000110101010110110000'', 800, 400,
45, 'Indian');

-- Home Cook2
INSERT INTO HomeCook2
VALUES
```

```
(101, 'GourmetGuru',
''10000110100010111100001010010100011110010001000011010101101100000''),
(102, 'CulinaryQueen',
''10000110100010111100001010010100011110010001000011010101101100000''),
(103, 'EpicureanEater',
''10000110100010111100001010010100011110010001000011010101101100000''),
(104, 'FoodieFrenzy',
''10000110100010111100001010010100011110010001000011010101101100000''),
(105, 'TasteTester',
''10000110100010111100001010010100011110010001000011010101101100000'');

-- Home Cook3
INSERT INTO HomeCook3
VALUES
(200, 'Intermediate'),
(100, 'Novice'),
(300, 'Advanced'),
(50, 'Novice'),
(150, 'Intermediate');

-- Professional Chef 1
INSERT INTO ProfessionalChef1
VALUES
('MasterChef',
''10000110100010111100001010010100011110010001000011010101101100000'', 1000, 500,
45, 'FineDiningEats', 'New York'),
('GourmetGuru',
''10000110100010111100001010010100011110010001000011010101101100000'', 800, 400,
40, 'TasteOfItaly', 'Rome'),
('CulinaryMaestro',
''10000110100010111100001010010100011110010001000011010101101100000'', 1200, 600,
50, 'SushiSensation', 'Tokyo'),
('ChefExtraordinaire',
''10000110100010111100001010010100011110010001000011010101101100000'', 1500, 700,
55, 'SpiceParadise', 'Mumbai'),
('CookingProdigy',
''10000110100010111100001010010100011110010001000011010101101100000'', 2000, 1000,
60, 'HauteCuisine', 'Paris');

-- ProfessionalChef2
INSERT INTO ProfessionalChef2
VALUES
```

```
 (201, 'SousChefSupreme',
''10000110100010111100001010010100011110010001000011010101101100000''),
(202, 'ChefDeCuisine',
''10000110100010111100001010010100011110010001000011010101101100000''),
(203, 'ExecutiveChef',
''10000110100010111100001010010100011110010001000011010101101100000''),
(204, 'PastryMaestro',
''10000110100010111100001010010100011110010001000011010101101100000''),
(205, 'SaucierSavant',
''10000110100010111100001010010100011110010001000011010101101100000'');

-- Professional Chef 3
INSERT INTO `ProfessionalChef3
VALUES
('TasteOfChina', 'Beijing', 'MasterChef'),
('MediterraneanMastery', 'Athens', 'CulinaryExpert'),
('NordicNectar', 'Stockholm', 'GastronomyPro'),
('SoulfulSpices', 'New Delhi', 'FlavorWizard'),
('PacificPleasures', 'Sydney', 'SensoryArtist');

-- User 1
INSERT INTO `User 1`
VALUES
('FoodFanatic',
''10000110100010111100001010010100011110010001000011010101101100000'', 500, 200,
28),
('CookbookAddict',
''10000110100010111100001010010100011110010001000011010101101100000'', 300, 150,
35),
('TasteTester',
''10000110100010111100001010010100011110010001000011010101101100000'', 700, 300,
40),
('FlavorFan', ''10000110100010111100001010010100011110010001000011010101101100000'',
200, 100, 30),
('EaterExtraordinaire',
''10000110100010111100001010010100011110010001000011010101101100000'', 1000, 500,
45);

-- User 2
INSERT INTO `User 2`
VALUES
(301, 'CuisineConnoisseur',
''10000110100010111100001010010100011110010001000011010101101100000''),
```

```
(302, 'DishDevotee',
''10000110100010111100001010010100011110010001000011010101101100000''),
(303, 'GastronomyGuru',
''10000110100010111100001010010100011110010001000011010101101100000''),
(304, 'EpicureanExplorer',
''10000110100010111100001010010100011110010001000011010101101100000''),
(305, 'PalatePioneer',
''10000110100010111100001010010100011110010001000011010101101100000'');

-- Leaderboard
INSERT INTO Leaderboard
VALUES
('Top Chefs', 'Exclusive Cooking Class'),
('Best Recipes', 'Cookbook Collection'),
('Most Active Users', 'Gift Card'),
('Highest Ratings', 'Kitchen Gadgets'),
('Ultimate Foodie', 'Fine Dining Experience');

-- Event1
INSERT INTO Event1
VALUES
('Cooking Competition', 50, 'New York'),
('Food Festival', 20, 'Paris'),
('Cooking Workshop', 30, 'Tokyo'),
('Culinary Tour', 100, 'Rome'),
('Tasting Event', 25, 'Sydney');

-- Event2
INSERT INTO Event2
VALUES
(1, '2024-04-15', 'Cooking Class', 50),
(2, '2024-05-20', 'Chef's Table Dinner', 100),
(3, '2024-06-10', 'Wine Pairing Event', 75),
(4, '2024-07-05', 'Cook-off Challenge', 75),
(5, '2024-08-12', 'Food Truck Rally', 0);

-- Review1
INSERT INTO Review1
VALUES
('2024-01-10', 'Great recipe!', 5),
('2024-02-15', 'Delicious dish!', 4),
('2024-03-20', 'Easy to follow instructions.', 5),
('2024-04-25', 'Could use more seasoning.', 3),
```

```
('2024-05-30', 'Not what I expected.', 2);

-- Review2
INSERT INTO Review2
VALUES
(101, '2024-06-05', 'Best recipe ever!'),
(102, '2024-07-10', 'Disappointing outcome.'),
(103, '2024-08-15', 'Needs improvement.'),
(104, '2024-09-20', 'Absolutely delicious!'),
(105, '2024-10-25', 'Will make again.');

-- Cooking Equipment1
INSERT INTO `Cooking Equipment1`
VALUES
('KitchenAid', 'Mixer', 200, 'High'),
('Cuisinart', 'Blender', 150, 'Medium'),
('Le Creuset', 'Cookware', 300, 'High'),
('Wusthof', 'Knife', 100, 'High'),
('All-Clad', 'Cookware', 250, 'High');

-- Cooking Equipment2
INSERT INTO `Cooking Equipment2`
VALUES
('Instant Pot', 'Pressure Cooker', 120, 'High'),
('Vitamix', 'Blender', 400, 'High'),
('Global', 'Knife', 150, 'High'),
('Pyrex', 'Bakeware', 50, 'Medium'),
('Breville', 'Toaster Oven', 180, 'High');

-- Recipe1
INSERT INTO Recipe1
VALUES
('2023-01-01', 'Spaghetti Carbonara', 'Italian', 'Intermediate', 4),
('2023-02-15', 'Coq au Vin', 'French', 'Advanced', 6),
('2023-03-20', 'Sushi Rolls', 'Japanese', 'Intermediate', 3),
('2023-04-25', 'Tacos al Pastor', 'Mexican', 'Beginner', 4),
('2023-05-30', 'Chicken Tikka Masala', 'Indian', 'Intermediate', 4);

-- Recipe2
INSERT INTO Recipe2
VALUES
(201, 101, 1, '2023-06-05', 60, 'Beef Wellington'),
(202, 102, 2, '2023-07-10', 45, 'Ratatouille'),
```

```
(203, 103, 3, '2023-08-15', 30, 'Pad Thai'),
(204, 104, 4, '2023-09-20', 75, 'Lamb Curry'),
(205, 105, 5, '2023-10-25', 60, 'Tiramisu');

-- Video
INSERT INTO Video
VALUES
(201, 'Beef Wellington Recipe', '2023-06-05', 3600),
(202, 'Ratatouille Cooking Demo', '2023-07-10', 2700),
(203, 'Authentic Pad Thai Tutorial', '2023-08-15', 1800),
(204, 'Spicy Lamb Curry Recipe', '2023-09-20', 4500),
(205, 'Classic Tiramisu Tutorial', '2023-10-25', 3600);

-- Ingredient
INSERT INTO Ingredient
VALUES
('Eggs', 'None'),
('Chicken', 'None'),
('Rice', 'None'),
('Tomatoes', 'None'),
('Flour', 'Gluten');

-- Follows
INSERT INTO Follows
VALUES
(1, 2),
(2, 3),
(3, 4),
(4, 5),
(5, 1);

-- Participates
INSERT INTO Participates
VALUES
(101, 1),
(102, 2),
(103, 3),
(104, 4),
(105, 5);

-- Ranking1
INSERT INTO Ranking1
VALUES
```

```
(1000, 1),
(800, 2),
(1200, 3),
(1500, 4),
(2000, 5);

-- Ranking2
INSERT INTO Ranking2
VALUES
(101, 'Top Chefs', 1000),
(102, 'Top Chefs', 800),
(103, 'Top Chefs', 1200),
(104, 'Top Chefs', 1500),
(105, 'Top Chefs', 2000);

-- Utilizes
INSERT INTO Utilizes
VALUES
(201, 'Beef', 'Angus'),
(202, 'Eggplant', 'Local Farm'),
(203, 'Rice Noodles', 'Thai Kitchen'),
(204, 'Lamb', 'Halal'),
(205, 'Mascarpone', 'BelGioioso');

-- Posts
INSERT INTO Posts
VALUES
(201, 1),
(202, 2),
(203, 3),
(204, 4),
(205, 5);
```