**Changes to ER Diagram:**

1. Eliminate the description attribute from Manga because description belongs to the AnimatedSeries;
2. We add a relationship In between character and AnimatedSeries because different character in different animated series could share same name so we need to build a new connection, it should be full participation, one to many;
3. We change the name in Character, name in Voice Actor to cname, vname to avoid confusion;
4. Correct the inappropriate types for entity attributes;
5. We add another relationship Favored between User and AnimatedSeries, it’s a many to many relationship;
6. For Subtopic, the relationship Create goes from User to Subtopic instead of Admin to Subtopic;
7. For Comment, add relationship Write between Comment and User, full participation from the Comment side
8. Add rname for User referring the real name of the User
9. Bold the relationship Have for Forum and Subtopic, bold the relationship Contain for Subtopic and Comment and also add dotted underline for title, date, msg as partial keys

**Schema**

// entity

1) *Anime*(title: char,genre: char,rating : int , description: char ,season: int,status: char, start date : char, studio: char, #of episode: int)

* Represents: entity set *Anime*.
* Primary key: Title,season
* Foreign keys: N/A
* Constraints:
  + There is a ISA relationship between AnimatedSeries and Anime
  + Anime must have a title and season

2*) Manga* ( title : char,genre : char,rating:int, description: char,author: char,volume: int, published date :char)

* Represents: entity set *Manga*.
* Rating is a number between 1-5.
* Primary key: title, author
* Foreign keys: N/A
* Constraints:
  + There is a ISA relationship between AnimatedSeries and Manga
  + Manga must have a title and author

*3) Voice Actor* ( vname: char, birthday : char, gender : char, bio : char)

* Represents: entity set Voice Actor.
* Primary key: vname, birthday
* Gender is one of {"male", "female" }
* Foreign keys: NA
* Constraints: *Voice Actor must have name and birthday*

4) *Character* (cname : char, description : char, rating :int)

* Represents: entity set *Character*
* Primary key: cname
* Constraints:
  + Every character must have cname and cname is unique

5) *Admin*(Email : char,admin name: char, password: char, name: char)

* Represents: entity set *Admin*
* Primary key: Email
* Constraints:
  + Every Admin must have an email Address.

6) *Member*(Email : char,user name: char, password: char, name: char)

* Represents: entity set *Member*
* Primary key: Email
* Constraints:
  + Every Member must have an email Address.

// relationship

7) *Voice*(Title : char,Season:int, Birthday : char, Vname : char, Cname : char)

* Represents: the ternary *voice* relationship set.
* Primary key: Title,Season,Birthday,Vname,Cname
* Foreign keys: (Anime)Title,(Anime)Season,(VoiceActor)birthday,(VoiceActor)vname, (Character)cname

8) *Favored* (Title : char,email: char)

* Represents: the *Favored* relationship set.
* Primary key: Title,email
* Foreign keys: (Anime)Title,(Manga)Title,(Admin)email,(Member) email

*9) Created\_Forum\_Forum* ( Forum ID :int, name: char, date\_created: char,Email: char)

* Represents: entity set Forum combined with Forum\_create relationship set
* Primary key: Forum ID
* Foreign key: Email References (Admin) Email
* Constraints: Forum iD , Email should be unique and not null

*10) Subtopic\_Create\_Subtopic\_Have* ( Forum ID :int, date created: char, title: char, Email: char)

* Represents: entity set Subtopic combined with *Have* and *Subtopic\_create* relationship sets
* Primary key: Forum ID, title,
* Foreign keys: (Created\_Forum\_Forum)Forum ID,(Admin)Email,(Member) Email
* Constraints: Subtopic is a weak entity with strong entity Forum, once the Forum is deleted, the subtopic is deleted

*11) Comment\_Write\_Contain* ( Forum ID: int, title : char, msg: char, date : char, email: char)

* Represents: entity set Comment combined with *contain* and  *write* relationship sets.
* Primary key: Forum ID,title, msg, date
* Foreign keys: (Forum)Forum ID, (Subtopic)title, (Admin) email, (Member) email
* Constraints: Comment is a weak entity with strong entity subtopic, once the Forum is deleted, the subtopic is deleted

*12) In* ( Cname : char, Title: char)

* Represents: the *In* relationship set
* Primary key: Cname, Title
* Foreign key: (Character) Cname, (Anime) Title, (Manga) Title
* Constraints: All characters must participate in this relationship

**Functional Dependencies**

Manga

Author, Title -> Publish Date, Volume, Description, Rating, Genre

* The author and title determine which manga we are talking about and thus determines that manga’s appropriate information such as publish date, rating, description, genre and volume.

Anime

Title, Season -> status, start date, studio, # eps, Genre, Rating, Description

* The title and season determine which anime we are talking about and thus will determine its corresponding information

Start Date, # eps -> status

* From the start date and number of episodes we will know if the anime is still airing or not because anime airs on a regular weekly basis

Voice Actor

Vname, Birthday -> gender, biography,

* The Vname and Birthday determine which actor/actress we are referring to and thus determines their gender and biography

Character

Cname -> description, rating

* Cname defines which character we are talking about and thus determines its corresponding description and rating

Member

Email -> username, password, Name

* Email defines which member we are referring to and thus determines its corresponding information like username, password, name

username -> password

* Username should have its corresponding password

Admin

Email -> admin\_name, password, Name

* Email defines which admin we are referring to and thus determines its corresponding information like username, password, name

admin\_name -> password

* Admin\_name should determine its corresponding password for sign in

Created\_Forum\_Forum

forumID -> name, date\_created, email

* forumID determines which forum we are referring to and thus determines its corresponding information like its creator’s email, the name and date that it was created

Subtopic\_Create\_Subtopic\_Have

Title, ForumID -> date\_created, Email

* A subtopic’s title and forumID will determine that specific subtopic and thus will determine when it was created and the creator

Comment­\_create\_contain

ForumID, Title, Msg, Date -> Email

* The ForumID and subtopic title along with the comment’s msg and date determines which person wrote the comment

Voiced

Title, season, cname -> vname

* The title and season of the anime along with the character determines which voice actor voiced the character in that particular season of that anime

Favored

None

In

Cname -> Title

* The character name determines which animated series it belongs to

**Tables after Normalization**

All tables above except anime, member and admin are key dependencies, therefore all tables except these three are already in BCNF.

Normalizing *anime* into BCNF

* aniStatus(startDate, #eps, status)
* anime1(titie, season, startDate, #eps, studio, genre, rating, description)

Normalizing *member* into BCNF

* user(username, password)
* member1(email, name, password)

Normalizing *admin* into BCNF

* adminName(password, adminName)
* admin1(password, Email, name)

**The SQL DDL**

// Tables for Entity

1. CREATE TABLE *Anime*

(

Title char(100),

Genre char(256),

Rating int ,

Description char(10000),

Season int,

Status char(100),

Start\_date char(100),

Studio char(100),

# of episodes int,

Primary key (Title, Season)

)

2. CREATE TABLE *Manga*

(Title char(100),

Genre char(256),

Rating int ,

Description char(10000),

Author char (100),

Volume int,

Published\_date char(100),

Primary key (Title, Author)

)

3. CREATE TABLE *Voice Actor*

( Vname char (100),

Birthday char(100),

Gender char(6),

Bio char(10000),

Primary key (Name, Birthday))

4. CREATE TABLE *Character*

(Cname char(100),

Description char(10000),

Rating int,

Primary key (Cname))

5. CREATE TABLE *Admin*

(Email char (100),

Name char (100),

Admin\_name char (100),

Password char (20),

Primary key (Email))

6. CREATE TABLE *Member*

(Email char (100),

Name char (100),

Username char (100),

Password char (20),

Primary key (Email))

// Tables for Relationship

7. CREATE TABLE Created\_Forum\_Forum

(

Forum ID int,

Email char(100),

Name char (100),

Date\_created char (100),

Primary key (Forum ID),

Foreign key (Email) REFERENCES Admin(Email)

)

8. CREATE TABLE *Subtopic\_Create\_Subtopic\_Have*

(

Forum ID int,

Date created char (100),

Title char (100),

Email char (100),

Primary key (Forum ID, Title),

Foreign key (Forum ID) REFERENCES Create\_Forum\_Forum(Forum ID))

9. CREATE TABLE *Comment\_Write\_Contain*

(

Forum ID int,

Title char (100),

Msg char (10000),

Date char (100),

Email char (100),

Primary key (Forum ID, Title, Msg, Date),

Foreign key (Forum ID, Title) REFERENCES Subtopic\_Create\_Subtopic\_Have (Forum ID, title),

Foreign key (Email) REFERENCES Admin(Email), Member(Email)

)

10. CREATE TABLE *Voice*

(Title char (100),

Season int ,

Birthday char (100),

Vname char (100),

Cname char(100),

Primary key (Title,Season,Birthday,Vname,Cname)),

Foreign key (Title,Season) REFERENCES Anime(Title,Season),

Foreign key (Birthday, Vname) REFERENCES VoiceActor(Birthday,Vname),

Foreign key (Cname) REFERENCES character(Cname))

11. CREATE TABLE *Favored*

(Title char (100),

Email char (100),

Primary key (Title, Email)),

Foreign key (Email) REFERENCES Admin(Email),Member(Email),

Foreign key (Title) REFERENCES Anime(Title), Manga(Title))

12. CREATE TABLE *In*

(Cname char(100),

Title char(100),

Primary key (Cname, Title),

Foreign key (Cname) REFERENCES Character(Cname),

Foreign key (Title) REFERENCES Anime(Title), Manga(Title))

**Specifications**

**Platform Used:**

We will be using the CS Ugrad Oracle installation and provided PHP/Apache along with HTML5 and css.

**Functionality of the final application:**

* + **Which classes of users are you going to provide different interfaces to the data**
  + There is a single interface type that will be used by the admin,member and guest. There will however be different permission levels, with the admin having max control, and the users having restricted functionality. Admins will be able to add new animes and manga as well as create new forums. Members can create subtopics under each forum and comment. Guests will only be able to browse the site and search for anime but will not be able to comment or add anything.
  + **class users access restriction:**
  + User: have partial functionality
    - Search for an anime or manga or voice actor or character
    - Remove comment or subtopic
    - Search by comment or subtopic
    - Update comment or subtopic
    - Add anime or manga to the list of favorite
    - Read a forum
    - Read,create or delete a subtopic
    - Read,create or delete a comment
    - Create an account on the website
    - Search for author that has created more than 5 manga
    - Search for manga that has been adapted from anime

Admin: have maximal functionality

* + - Search for an anime or manga or voice actor or character
    - Remove comment or subtopic or forum
    - Add comment or subtopic or forum
    - Search by comment or subtopic
    - Update comment or subtopic
    - Add anime or manga to the list of favorite
    - Read a forum
    - Read,create or delete a subtopic
    - Read,create or delete a comment
    - Create an account on the website
    - Search for author that has created more than 5 manga
    - Search for manga that has been adapted from anime
    - Add anime or manga or character or voice actor
    - Delete anime or manga or characters or voice actor
    - Update or remove user

Guest: Limited functionality

* + - Search for an anime or manga or voice actor or character
    - Read a forum
    - Create an account on the website
    - Search for author that has created more than 5 manga
    - Search for manga that has been adapted from anime
  + **Update some part of the database from the application, *and* have some queries that depends on the user's input** 
    - There are several instances of this in our database.
    - Ability to add and remove anime/manga/voice actor/character is restricted to Admin user.
    - Users can create and remove comments with inappropriate words
    - Users can search by keyword for an anime or manga they are looking for, we can provide filters as well to refine their search.
  + **interesting queries - beyond a simple select, project, join.**
  + we have several complex queries such examples include:
    - We plan to add a vote button under AnimatedSeries for users and we collect the rating and find the most favorited animated series according to this voting system
    - Most recently created subtopic by users
    - We collect the comments from all the users and we find the most commented subtopic by users
    - We will implement an anime or manga recommendation feature where the user will input a couple genres that he/she is interested in. We will query series of that genre as well as their ratings and then query the user’s fav list of anime if he/she has any to make the final suggestion
  + **triggers or assertions:**
  + Assert that anime status is not null
  + Trigger on delete anime a confirmation button.

**What data will be in the final application (It's fine to just reiterate what you said in the schema-checkpoint). Our goal here is to make sure that the application will still work well with the data that you have. For the same reason, including your schema with your report would also be helpful, though not required.**

We are including data about each animated series with its corresponding anime and manga. We are keeping track of voice actors and characters from anime. We are keeping track of two types of users: member and admin and their corresponding information. We are storing the forum and its subtopics and comments. We are storing each user’s favorite animated series as a favorite list.

**The division of labour:**

* All team members learn how to use git, and how to code in SQL, html, php;
* For updating the ER diagram and translating ER diagram to schema and create SQL DDL table: Weining Hu and Shehryar Tariq;
* For populating each table with 5 tuples: Frank Rui (provide examples) and Weining Hu for populating it;
* For figuring out the functional dependency: Frank Rui
* For normalization of the FD’s and final editing of formal specification: Yu Ju Liang.

Designing the layout the appearance of the web app (skills: html and css)

* Connecting the data from our dataset with our web app using PHP (Frank Rui)
* Adding interesting functionality such as linking with Facebook login in and make the website more pretty (Weining & Shehryar)
* testing the flow of database app (Yu Ju Liang)
* SQL populating data and create tables: together as a team