Simon Fraser University

CMPT 354 – Database System I

# Project Title: Chemistry Search Engine

By

Project Group 54:

Canh Nhat Minh Le

Chris Liu

Frank Dong

Hong Quang Cung

Xixuan Song

February 2021

Computer Science Department

## Project Objective:

* To apply and enhance knowledge about Database System as well as Database Management
* To develop a Search Engine in Chemistry that supports students in reviewing crucial concepts in Chemistry
* To provide additional services including discussion and practice section to assist students in consolidating knowledge in Chemistry

## Project Description:

Recognition of the difficulty students might encounter while studying chemistry. The team decides to create a search engine that allows students to cover the gap in their knowledge. While building the large-scale database to share a vast knowledge of chemistry is the team’s expectation, the group decide to narrow the scale of the application down to chemistry knowledge at the high school level due to limitations of time and resources.

The team concentrates on developing <web-based application> including three major sections which are Search, Practice, and Discussion sections:

* *Search Section:* this is the heart of the program allowing students to seek fundamental knowledge and concepts in chemistry that store in the database.
* *Practice Section:* students can improve their understanding of questions in the Practice Section.
* *Discussion Section:* this is another additional service that the application provides. Students can ask and seek help from their peers via the forum.

The team selects Python as a primary platform to develop the GUI of the program and use MySQL as an official language to build the database management system (DBMS). The team also uses Flask as a web framework to make the program.

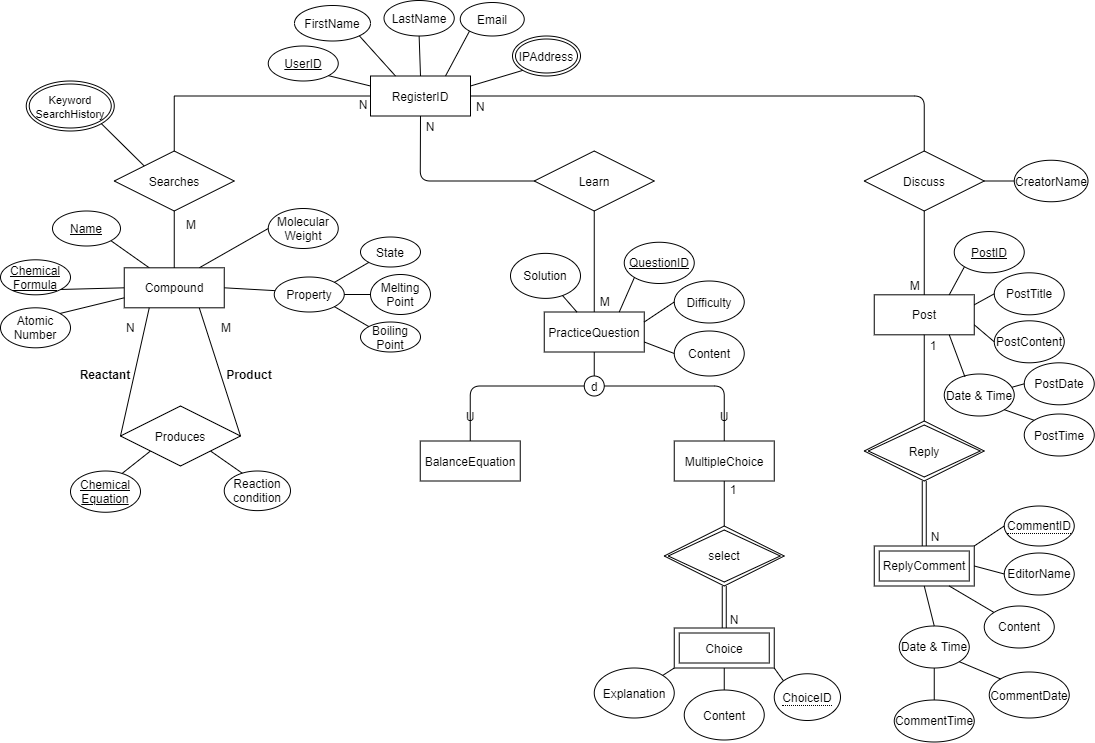
Summary,

* Application: Chemistry Search Engine
* Type of application: Web base application
* Domain: Chemistry knowledge at high school level
* Platform:
  + GUI: Python
  + DBMS: MySQL
  + Other: Flask

## ER Diagram:

Description of ER Diagram:

* Entity: Access, Compound, Reaction, Practice Question, Forum
* Weak Entity: Reply, Choices
* Relation: Searches, Learn, Discuss, Participates in, Prepares
* Generalization/Specialization: Practice Question/[Balance Equation, Problem Solving, Multiple Choice]



The report shows schema and table in the outline of branch, there are 3 branches:

* Searching branch (User entity)
* Learning branch
* Discuss branch

1. ***Searching Branch:***

Compound[CompoundName, ChemicalFormula, AtomicNumber, State, MeltingPoint, BoilingPoint , Appearance, MolecularWeight]

Produces[ReactantFormula, ProductFormula, ChemicalEuqation, ReactionCondition]

Search[CompoundFormula, UserID]

SearchHistory[ChemicalFormula, UserID, KeywordSearchHistory]

Products.ReactantFormula references Compound.ChemicalFormula

Products.ProductFormula references Compound.ChemicalFormula

Search. ChemicalFormula references Compound.ChemicalFormula

Search.UserID references User.UserID

SearchHistory.{CompoundFormula, UserID} references Search.{CompoundFormula, UserID}

#### Table definition:

* “Compound” table:

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| CompoundName | VARCHAR(30) | NONE | NOT NULL, UNIQUE |
| ChemicalFormula | VARCHAR(10) | P | NOT NULL, UNIQUE |
| AtomicNumber | INT | NONE |  |
| State | VARCHAR(30) | NONE |  |
| MeltingPoint | INT | NONE |  |
| BoilingPoint | INT | NONE |  |
| Appearance | VARCHAR(100) | NONE |  |
| MolecularWeight | VARCHAR(30) | NONE |  |

* “Produces” table:

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| ReactantFormula | VARCHAR(30) | P & F | NOT NULL |
| ProductFormula | VARCHAR(10) | P &F | NOT NULL |
| ChemicalEquation | INT | NONE |  |
| ReactionCondition | VARCHAR(30) | NONE |  |

* “Search” table:

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| ChemicalFormula | VARCHAR(30) | P & F | NOT NULL, UNIQUE |
| UserID | INT | P &F | NOT NULL, UNIQUE |

* “SearchHistory” table:

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| ChemicalFormula | VARCHAR(30) | P & F | NOT NULL, UNIQUE |
| UserID | INT | P &F | NOT NULL, UNIQUE |
| KeywordHistroy | VARCHAR(100) | P |  |

#### SQL dump:

“Compound” table:

CREATE TABLE IF NOT EXISTS Compound(

CompoundName VARCHAR(30) NOT NULL UNIQUE,

ChemicalFormula VARCHAR(10) NOT NULL UNIQUE,

AtomicNumber INT,

State VARCHAR(30),

MeltingPoint INT,

BoilingPoint INT,

Appearance VARCHAR(100),

MolecularWeight VARCHAR(30),

PRIMARY KEY(ChemicalFormula)

);

“Produces” table:

CREATE TABLE IF NOT EXISTS Produces(

ReactantFormula VARCHAR(30) NOT NULL,

ProductFormula VARCHAR(30) NOT NULL,

ChemicalEquation VARCHAR(100) NOT NULL,

ReactionCondition VARCHAR(100),

PRIMARY KEY(ReactantFormula, ProductFormula),

FOREIGN KEY(ReactantFormula) REFERENCES Compound(ChemicalFormula) ON DELETE CASCADE,

FOREIGN KEY(ProductFormula) REFERENCES Compound(ChemicalFormula) ON DELETE CASCADE

);

“Search” table:

CREATE TABLE IF NOT EXISTS Search (

CompoundFormula VARCHAR(30) NOT NULL UNIQUE,

UserID INT NOT NULL UNIQUE,

PRIMARY KEY(CompoundFormula, UserID),

FOREIGN KEY(CompoundFormula) REFERENCES Compound(CompoundFormula) ON DELETE CASCADE,

FOREIGN KEY(UserID) REFERENCES RegisterUser(ID) ON DELETE CASCADE

);

“SearchHistory” table:

CREATE TABLE IF NOT EXISTS SearchHistory(

CompoundFormula VARCHAR(30) NOT NULL UNIQUE,

UserID INT NOT NULL UNIQUE,

KeywordHistory VARCHAR (100),

PRIMARY KEY(CompoundFormula, UserID),

FOREIGN KEY(CompoundFormula, UserID) REFERENCES Search(CompoundFormula,UserID) ON DELETE CASCADE

);

#### Screenshot:

1. ***Learning Branch:***

RegisterUser [UserID, FirstName, LastName, Email]

IPAddress [IPAddress, UserID]

PracticeQuestion [QuestionID, Content, Solution, Difficulty]

BalanceEquation [QuestionID]

MultipleChoice [Question ID]

Choice [QuestionID, ChoiceID, Content, Explanation]

Learn [UserID, QuestionID]

Foreign Keys:

IPAddress.UserID references RegisterUser.UserID

BalanceEquation.QuestionID references PracticeQuestion.QuestionID

MultipleChoice.QuestionID references PracticeQuestion.Question ID

Choice.Question ID references MultipleQuestion.Question ID

Learn.User ID references RegisterUser.UserID

Learn.QuestionID references PracticeQuestion.QuestionID

#### Table definition:

* “RegisterUser” table

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| UserID | CHAR(50) | P | NOT NULL, UNIQUE |
| FirstName | CHAR(50) | NONE | NOT NULL |
| LastName | CHAR(50) | NONE | NOT NULL |
| Email | CHAR(100) | NONE | NOT NULL |

* “IPAddress” table

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| IPAddress | CHAR(50) | NONE | NOT NULL, UNIQUE |
| UserID | CHAR(50) | P, F | NOT NULL, UNIQUE |

* “PracticeQuestion” table

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| QuestionID | INT | P | NOT NULL, UNIQUE |
| Content | VARCHAR(500) | NONE | NOT NULL |
| Solution | VARCHAR(500) | NONE | NOT NULL |
| Difficulty | CHAR(50) | NONE | NOT NULL |

* “BalanceEquation” table

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| QuestionID | INT | P, F | NOT NULL, UNIQUE |

* “MultipleChoice” table

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| QuestionID | INT | P, F | NOT NULL, UNIQUE |

* “Choice” table

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| QuestionID | INT | P, F | NOT NULL, UNIQUE |
| ChoiceID | CHAR(3) | P | NOT NULL |
| Content | VARCHAR(500) | NONE | NOT NULL |
| Explanation | VARCHAR(500) | NONE | NONE |

* “Learn” table

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| UserID | CHAR(50) | P, F | NOT NULL, UNIQUE |
| QuestionID | INT | P, F | NOT NULL, UNIQUE |

#### SQL dump:

CREATE TABLE RegisterUser (

UserID CHAR(50) NOT NULL,

FirstName CHAR(50) NOT NULL,

LastName CHAR(50) NOT NULL,

Email CHAR(100) NOT NULL,

PRIMARY KEY(UserID),

UNIQUE(UserID)

);

CREATE TABLE IPAddress (

IPAddress CHAR(50) NOT NULL,

UserID CHAR(50) NOT NULL,

PRIMARY KEY(UserID),

UNIQUE(UserID),

Foreign KEY (UserID) REFERENCES RegisterUser(UserID) ON DELETE CASCADE

);

CREATE TABLE Practice Question (

Question ID INT NOT NULL,

Content VARCHAR(500) NOT NULL,

Solution VARCHAR(500) NOT NULL,

Difficulty CHAR(50) NOT NULL,

PRIMARY KEY(Question ID),

UNIQUE(Question ID)

);

CREATE TABLE Balance Equation (

Question ID INT NOT NULL,

PRIMARY KEY(Question ID),

UNIQUE(Question ID),

Foreign KEY(Question ID) REFERENCES Practice Question(Question ID) ON DELETE CASCADE

);

CREATE TABLE Multiple Choice (

Question ID INT NOT NULL,

PRIMARY KEY(Question ID),

UNIQUE(Question ID),

Foreign KEY(Question ID) REFERENCES Practice Question(Question ID) ON DELETE CASCADE

);

CREATE TABLE Choice (

Question ID INT NOT NULL,

Choices ID CHAR(3) NOT NULL,

Content VARCHAR(500) NOT NULL,

Explanation VARCHAR(500),

PRIMARY KEY(Question ID, Choices ID),

Foreign KEY(Question ID) REFERENCES Multiple Choice(Question ID)ON DELETE CASCADE

);

CREATE TABLE Learn (

UserID CHAR(50) NOT NULL,

Question ID INT NOT NULL,

PRIMARY KEY(UserID, Question ID),

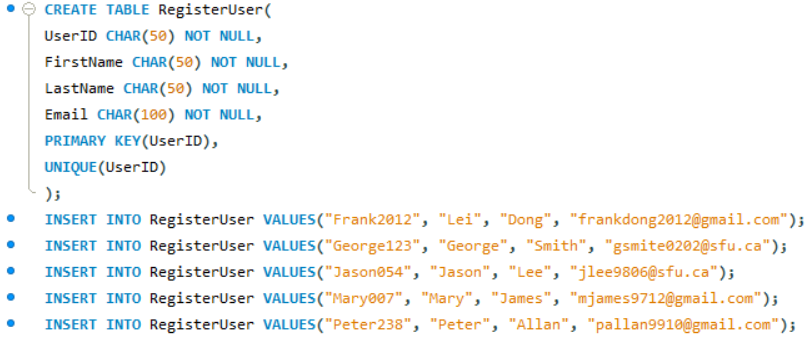
Foreign KEY (UserID) REFERENCES RegisterUser(UserID) ON DELETE CASCADE,

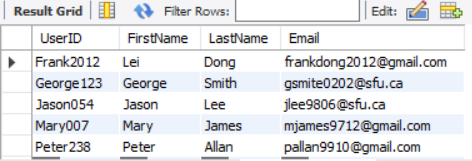
Foreign KEY(Question ID) REFERENCES Practice Question(Question ID) ON DELETE CASCADE

);

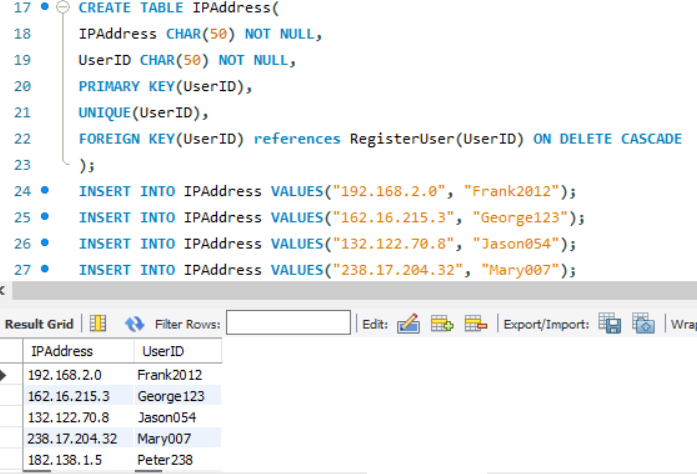
#### Screenshot:

* “RegisterUser” table

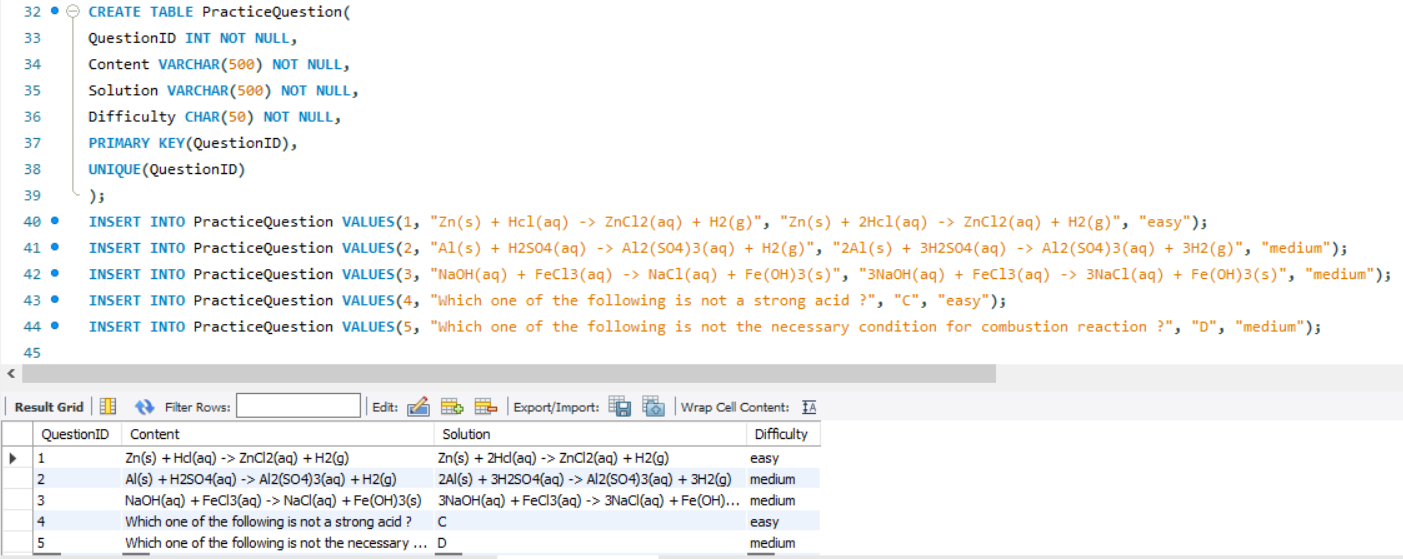




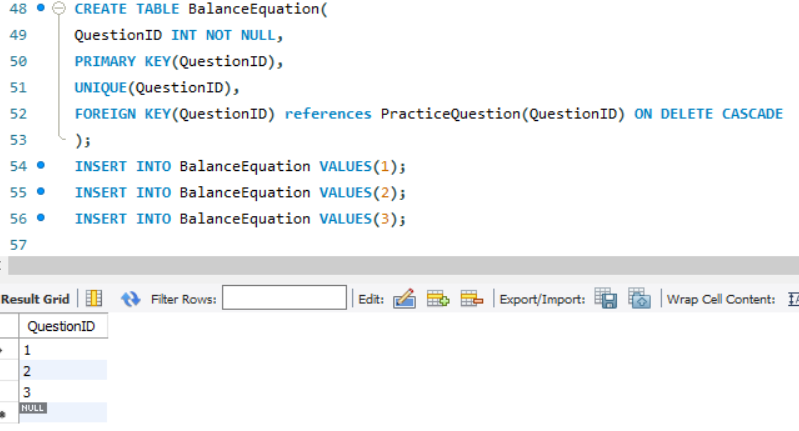
* “IPAddress” table



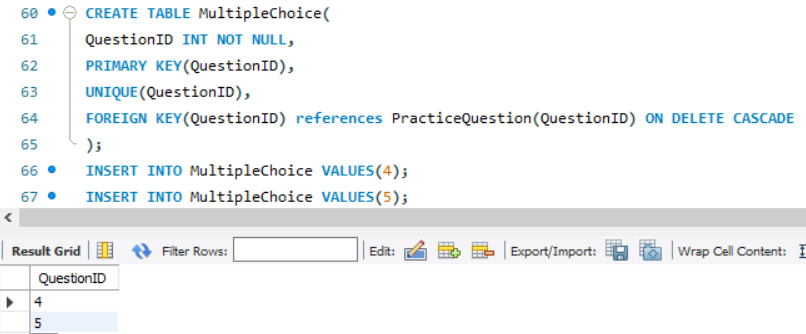
* “PracticeQuestion” table



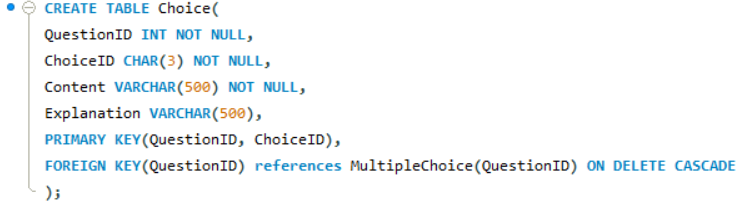
* “BalanceEquation” table

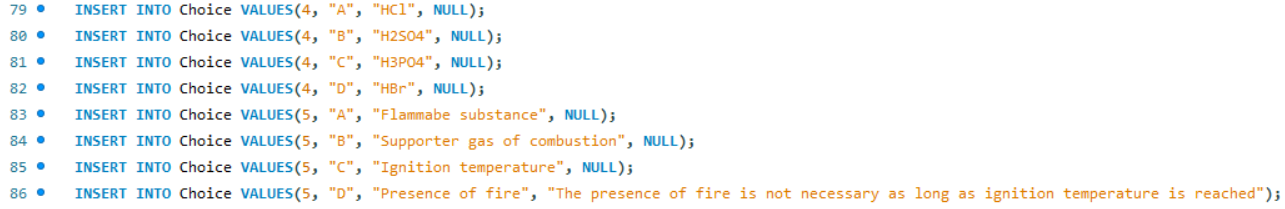


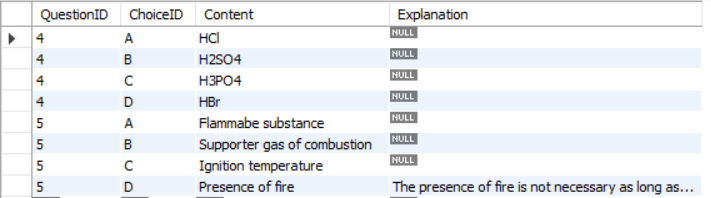
* “MultipleChoice” table



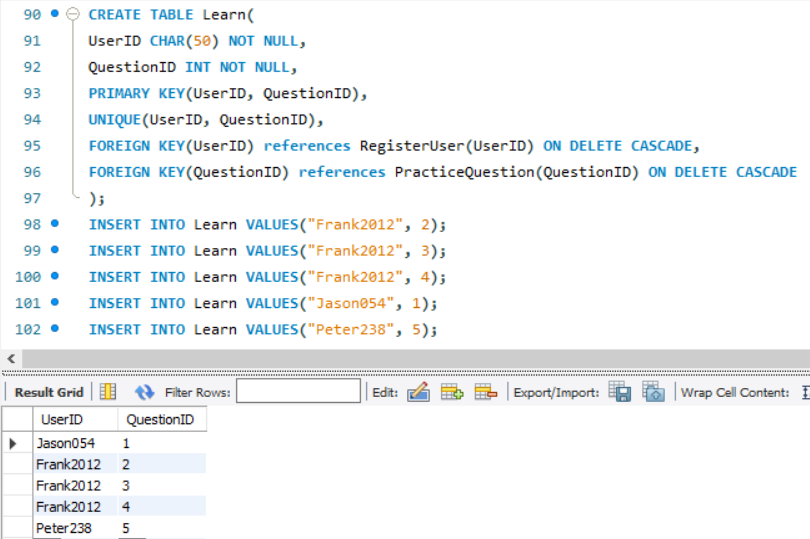
* “Choice” table







* “Learn” table



### Discuss Branch:

Discuss[PostID, ID, CreatorName]

Post[PostID, Post Title, PostContent, PostDate, PostTime]

Comment[CommentID, PostID, EditorName, Content, CommentDate, CommentTime]

Discuss.ID reference to User.ID

Discuss.PostID reference to Post.PostID

Comment.PostID reference to Post.PostID

#### Table definition:

* “Discuss” table:

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Name | Domain | Primary Key/Foreign Key | Other Constraints |
| PostID | INT | P & F | NOT NULL, UNIQUE, CASCADE |
| ID | INT | P & F | NOT NULL, UNIQUE, CASCADE |
| CreatorName | CHAR (200) | NONE | NOT NULL |

* “Post” table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Domain** | **Primary Key/Foreign Key** | **Constraint** |
| PostID | INT | P | NOT NULL, UNIQUE, DEFAULT |
| PostTitle | CHAR (200) | NONE | NOT NULL |
| PostContent | VARCHAR (600) | NONE | NOT NULL |
| PostDate | DATE | NONE | NOT NULL, CHECK (Current post Date) |
| PostTime | TIME | NONE | NOT NULL, CHECK (Current Post Time) |

* “ReplyComment” table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Domain** | **Primary Key/Foreign Key** | **Constraint** |
| PostID | INT | P & F | NOT NULL, UNIQUE, CASCADE |
| CommentID | INT | P & F | NOT NULL, UNIQUE, DEFAULT |
| EditorName | CHAR (200) | NONE | NOT NULL, DEFAULT (“Anonymous”) |
| Content | VARCHAR (500) | NONE | NOT NULL |
| CommentDate | DATE | NONE | NOT NULL, CHECK (Current comment date) |
| CommentTime | TIME | NONE | NOT NULL, CHECK (Current comment time) |

#### SQL dump:

“Post” table:

CREATE TABLE IF NOT EXISTS Post(

PostID INT NOT NULL UNIQUE,

PostTitle CHAR(200),

PostContent VARCHAR(600) NOT NULL,

PostDate DATE NOT NULL,

PostTime TIME NOT NULL,

PRIMARY KEY (PostID));

“Discuss” table:

CREATE TABLE IF NOT EXISTS Discuss(

PostID INT NOT NULL UNIQUE,

ID INT NOT NULL,

CreatorName CHAR(200) NOT NULL,

PRIMARY KEY (PostID, ID),

FOREIGN KEY (PostID) references Post(PostID) ON DELETE CASCADE,

FOREIGN KEY (ID) references RegisterUser(ID) ON DELETE CASCADE);

“ReplyCommnet” table:

CREATE TABLE IF NOT EXISTS ReplyComment(

CommentID INT NOT NULL,

PostID INT NOT NULL,

EditorName CHAR(200) DEFAULT “Anonymous”,

Content VARCHAR(500),

CommentDate DATE NOT NULL,

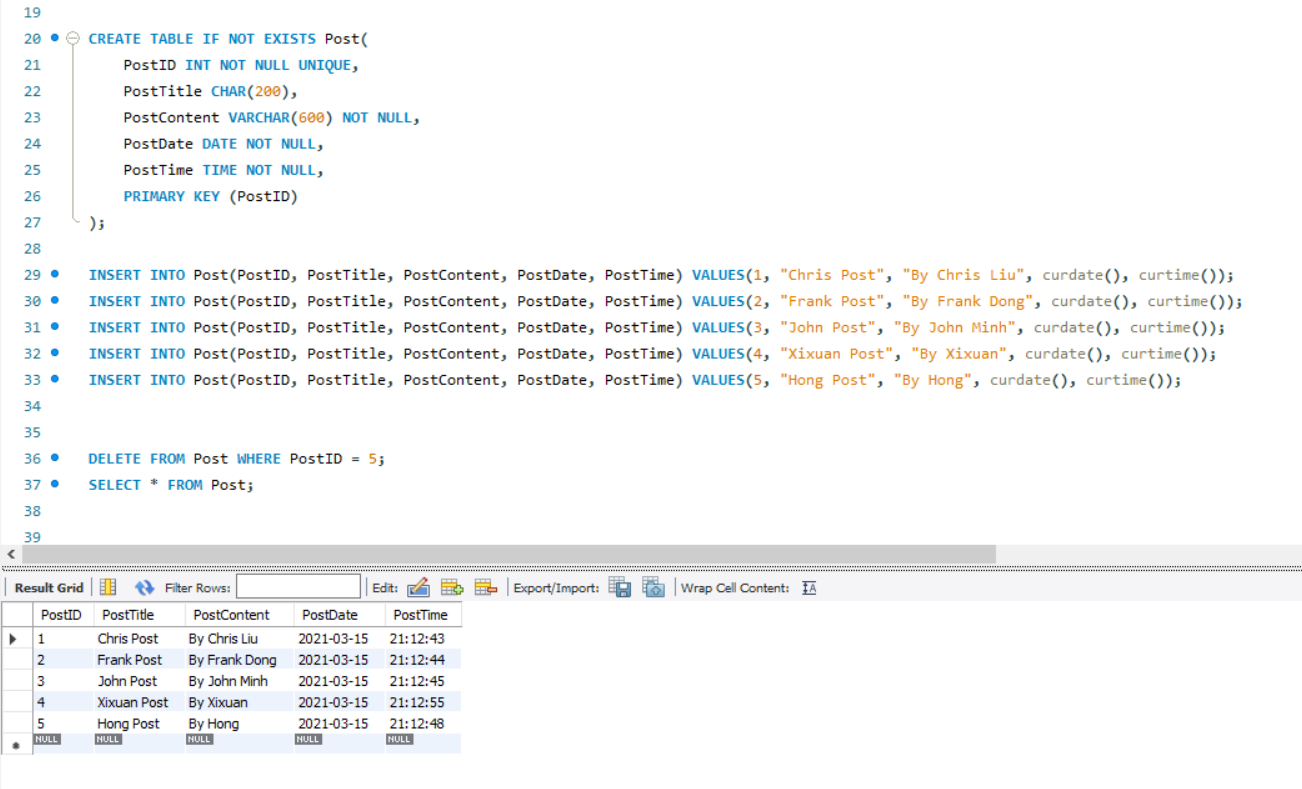
CommentTime TIME NOT NULL,

PRIMARY KEY(CommentID, PostID),

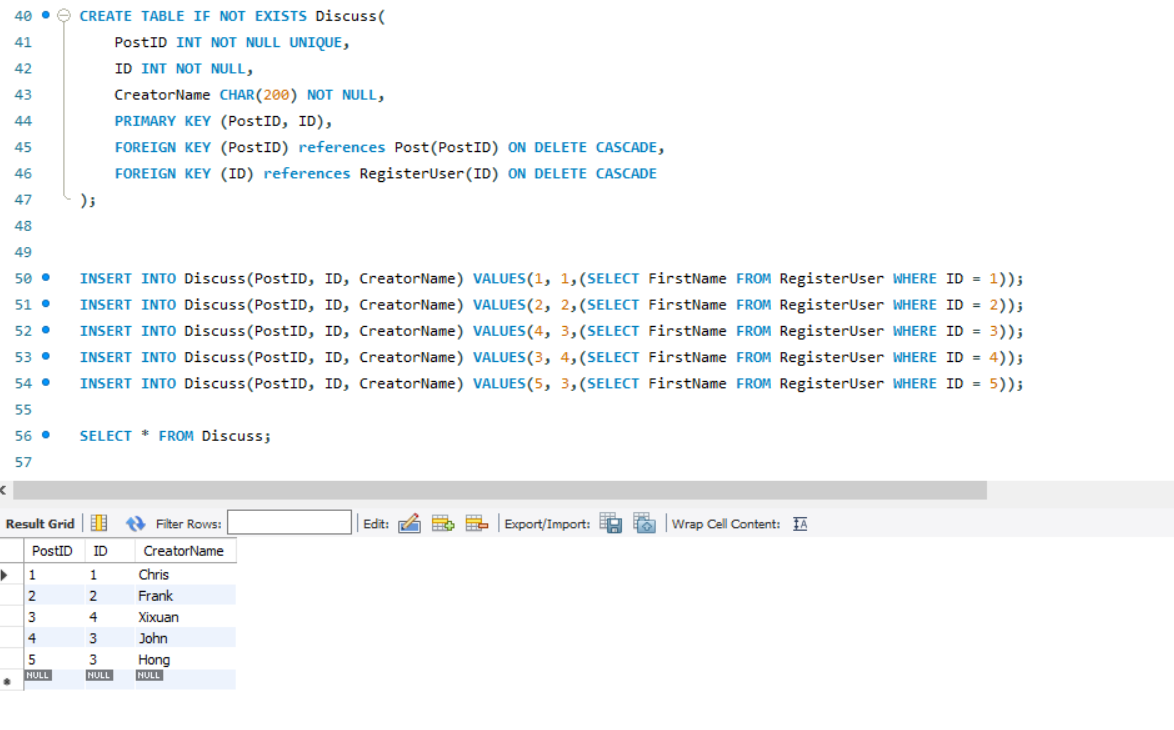
FOREIGN KEY Comment(PostID) references Post(PostID) ON DELETE CASCADE);

#### Screenshot:

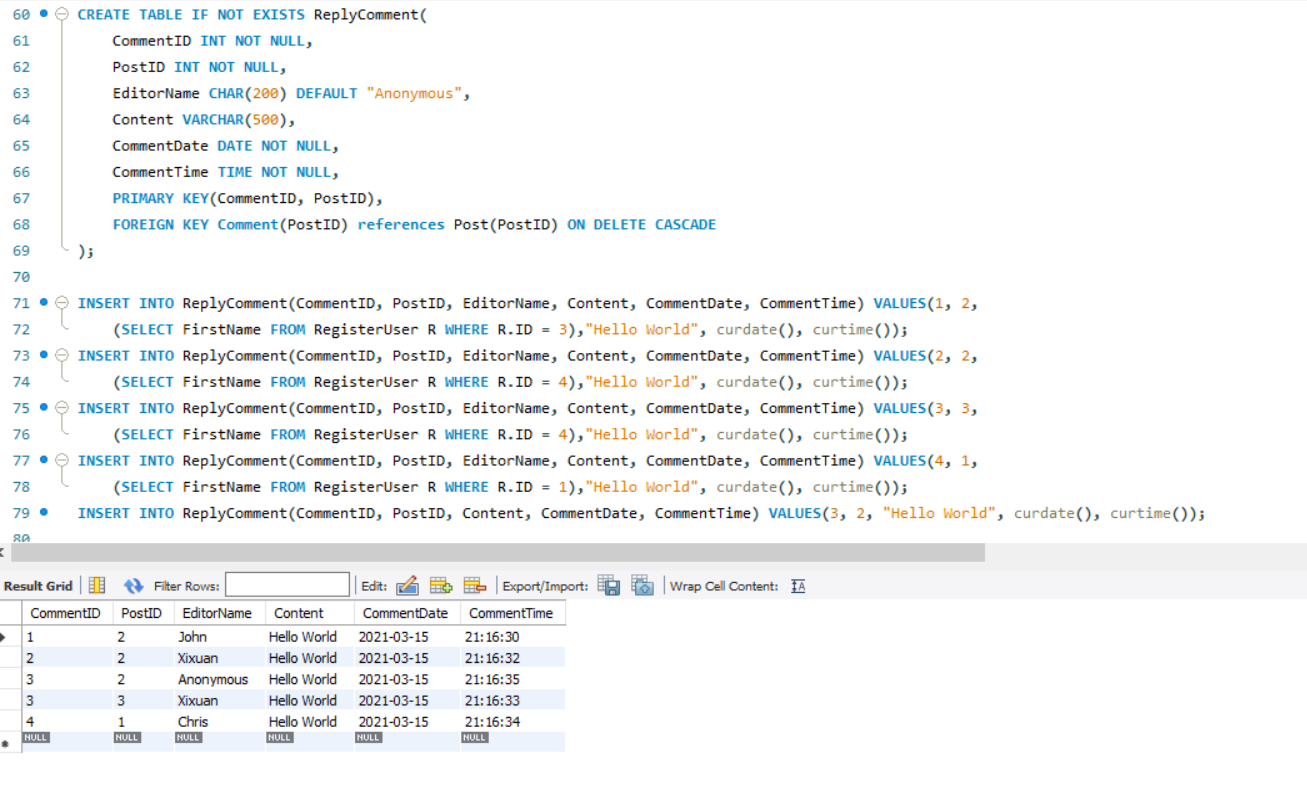
“Post” Table:



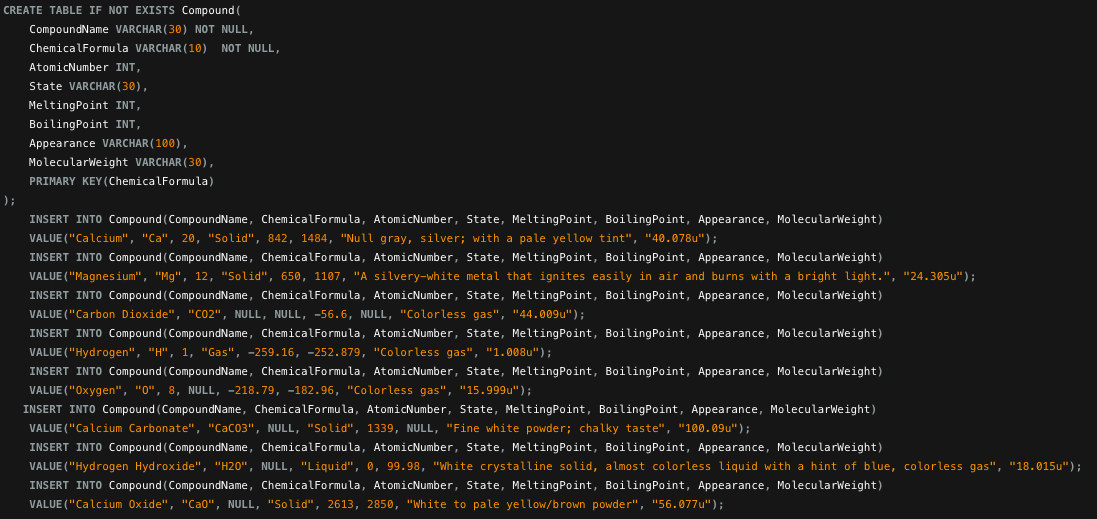
“Discuss” Table:

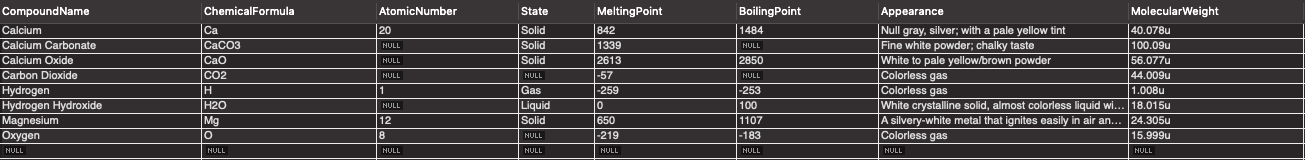
**

*“ReplyComment” Table:*

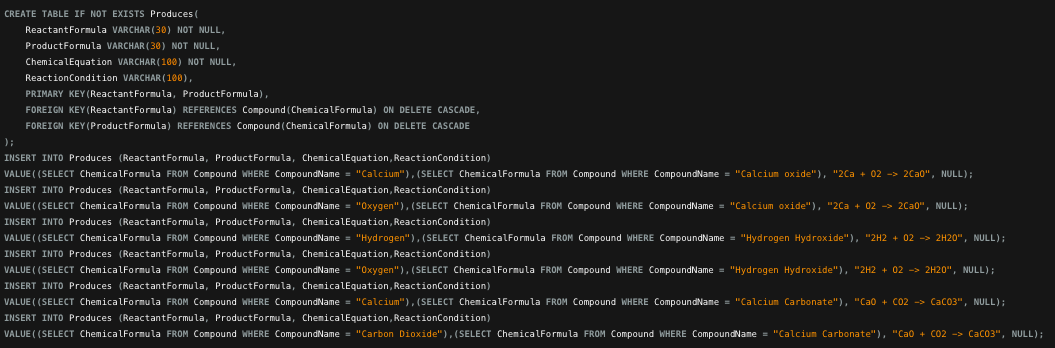
**

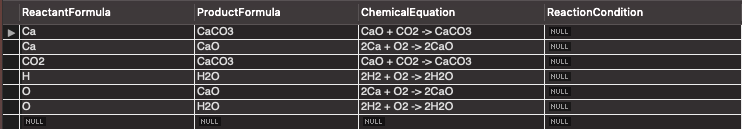
***“Compound” Table:***

**

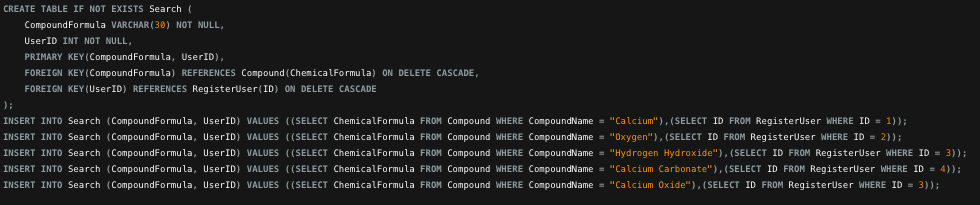
**

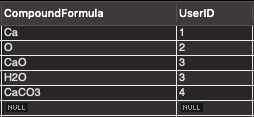
***“Produce” Table:***

**

**

***“Search” Table:***

**



***“SearchHistory” Table:***

