University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: Jul 21, 2024

Group Number: 27

Name	Student Number	CS Alias	E-mail Address
Jeffrey Zhai	63347439	n3c7e	jeffreygxzhai@gmail.com
Yixian Cheng	94548492	b9k9n	chengyx@student.ubc.ca
Mark Zhu	25611807	j9k7w	markziyuzhu@gmail.com

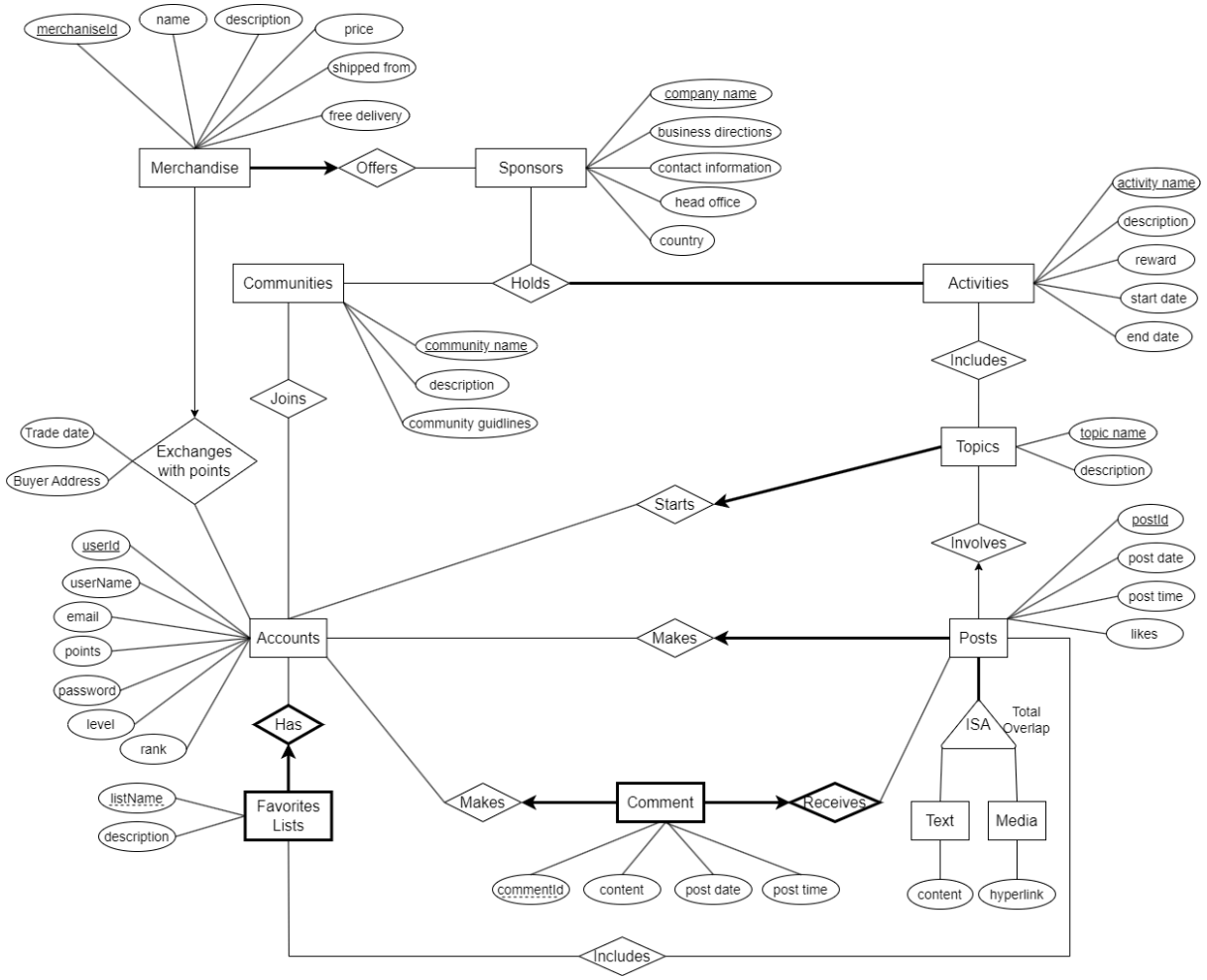
By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above.

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

2. Brief summary of project (~2-3 sentences):

This project portrays a social media platform that supports both community-based communication online and commercial activities. We aim to create an application that allows users to greet others and form communities via posting contents and participate in activities; we also allow sponsor companies to engage in activities and offer merchandise, which users could exchange for with the point credits they earned from the activities.

3. ER Diagram



Modifications since Milestone 1:

- Added attributes “level”, “rank” to entity Accounts;
- Added attributes “shipped from”, “free delivery” to entity Merchandise;
- Added attributes “head office”, “address” to entity Sponsors;
- Changed relationship between Merchandise and Sponsors from “Holds” to “Offers”, in order to distinguish this relationship from the “Holds” that involves Activities;

- Changed participation constraint between “Topics” and “Starts” to total participation, so that a topic must be started by an account;
- Changed participation constraint between “Comment” and “Makes” to total participation, so that a comment must be posted by an account;

4. Schema

Accounts(<u>userId</u>: INTEGER, userName: VARCHAR, email: VARCHAR, points: INTEGER, password: VARCHAR, level: INTEGER, rank: VARCHAR)

- **Primary Key:** userId
- **Candidate Keys:** email
- **Foreign Keys:** (None)
- **Other Constraints:**
    - All attributes are not NULL;
    - email is unique;
    - userId is a positive integer;
    - points is a non-negative integer;
    - level is a positive integer, between 1 and 10;
    - rank is a string from {“Newb”, “Experienced”, “Renowned”}, corresponding to level intervals {[1, 3], [4, 8], [9, 10]}

FavoriteLists(**<u>userId</u>**: INTEGER, <u>listName</u>: VARCHAR, description: VARCHAR)

- **Primary Key:** userId + listName; listName is partial key
- **Candidate Keys:** (None)
- **Foreign Keys:** userId
- **Other Constraints:**
    - userId and listName are not NULL by default;
    - userId references Accounts table; it has ON DELETE/UPDATE CASCADE;

Posts(<u>postId</u>: INTEGER, postDate: DATE, postTime: TIME, likes: INTEGER, **postedBy**: INTEGER, **topic**: INTEGER)

- **Primary Key:** postId
- **Candidate Keys:** (None)
- **Foreign Keys:** postedBy, topic
- **Other Constraints:**
    - All attributes are not NULL except for topic and postedBy;
    - postId is a positive integer;
    - likes is a non-negative integer;
    - postedBy references Accounts table; it has ON DELETE SET NULL and ON UPDATE CASCADE;

    - topic references Topics table; it has ON DELETE SET NULL and ON UPDATE CASCADE;

TextPosts(**postId**: INTEGER, content: VARCHAR)

- **Primary Key:** postId
- **Candidate Keys:** (None)
- **Foreign Keys:** postId
- **Other Constraints:**
    - All attributes are not NULL;
    - postId references Posts table; it has ON DELETE/UPDATE CASCADE;

MediaPosts(**postId**: INTEGER, hyperlink: VARCHAR)

- **Primary Key:** postId
- **Candidate Keys:** (None)
- **Foreign Keys:** postId
- **Other Constraints:**
    - All attributes are not NULL;
    - postId references Posts table; it has ON DELETE/UPDATE CASCADE;

Comments(**postId**: INTEGER, commentId: INTEGER, **commentedBy**: INTEGER, content: VARCHAR, postDate: DATE, postTime: TIME)

- **Primary Key:** postId + commentId; commentId is partial key
- **Candidate Keys:** (None)
- **Foreign Keys:** postId, commentedBy
- **Other Constraints:**
    - All attributes are not NULL except commentedBy;
    - postId references Posts table; it has ON DELETE/UPDATE CASCADE;
    - commentedBy references Accounts table; it has ON DELETE SET NULL and ON UPDATE CASCADE;

FavoritedPosts(**userId**: INTEGER, **listName**: VARCHAR, **postId**: INTEGER)

- **Primary Key:** userId + listName + postId
- **Candidate Keys:** (None)
- **Foreign Keys:** userId + listName, postId
- **Other Constraints:**
    - userId references FavoriteLists table (indirectly references Accounts table); it has ON DELETE/UPDATE CASCADE;
    - listName references FavoriteLists table; it has ON DELETE/UPDATE CASCADE;
    - postId references Posts table; it has ON DELETE/UPDATE CASCADE;

Topics(topicName: VARCHAR, description: VARCHAR, **startedBy**: INTEGER)

- **Primary Key:** topicName
- **Candidate Keys:** (None)
- **Foreign Keys:** startedBy
- **Other Constraints:**
    - All attributes are not NULL except for startedBy;
    - startedBy references Accounts table; it has ON DELETE SET NULL and ON UPDATE CASCADE;

Activities(activityName: VARCHAR, description: VARCHAR, reward: INTEGER, startDate: DATE, endDate: DATE)

- **Primary Key:** activityName
- **Candidate Keys:** (None)
- **Foreign Keys:** (None)
- **Other Constraints:**
    - All attributes are not NULL;
    - reward is a non-negative integer;
    - startDate must be earlier than endDate;

ActivityIncludesTopics(**topicName**: VARCHAR, **activityName**: VARCHAR)

- **Primary Key:** topicName + activityName
- **Candidate Keys:** (None)
- **Foreign Keys:** topicName, activityName
- **Other Constraints:**
    - All attributes are not NULL by default;
    - reward is a non-negative integer;
    - topicName references Topics table; it has ON DELETE/ UPDATE CASCADE;
    - activityName references Activities table; it has ON DELETE/ UPDATE CASCADE;

Communities(communityName: VARCHAR, description: VARCHAR, communityGuidelines: VARCHAR)

- **Primary Key:** communityName
- **Candidate Keys:** (None)
- **Foreign Keys:** (None)
- **Other Constraints:**
    - communityName and description are not NULL;

Joins(**communityName**: VARCHAR, **userId**: INTEGER)

- **Primary Key:** communityName + userId
- **Candidate Keys:** (None)
- **Foreign Keys:** communityName, userId

- **Other Constraints:**
    - communityName references Communities table; it has ON DELETE/ UPDATE CASCADE;
    - userId references Accounts table; it has ON DELETE/ UPDATE CASCADE;

Sponsors(companyName: VARCHAR, businessDirections: VARCHAR, contactInformation: VARCHAR, headOffice: VARCHAR, country: VARCHAR)

- **Primary Key:** companyName
- **Candidate Keys:** (None)
- **Foreign Keys:** (None)
- **Other Constraints:**
    - All attributes are not NULL except for businessDirections;

HoldActivities(**communityName**: VARCHAR, **companyName**: VARCHAR, **activityName**: VARCHAR)

- **Primary Key:** communityName + companyName + activityName
- **Candidate Keys:** (None)
- **Foreign Keys:** communityName, companyName, activityName
- **Other Constraints:**
    - All attributes are not NULL by default;
    - communityName references Communities table; it has ON DELETE/ UPDATE CASCADE;
    - companyName references Sponsors table; it has ON DELETE/ UPDATE CASCADE;
    - activityName references Activities table; it has ON DELETE/ UPDATE CASCADE;

Merchandise(merchaniseId: INTEGER, name: VARCHAR, description: VARCHAR, price: INTEGER, shippedFrom: VARCHAR, freeDelivery: CHAR(1), **offerdBy**: VARCHAR)

- **Primary Key:** merchaniseId
- **Candidate Keys:** (None)
- **Foreign Keys:** offeredBy
- **Other Constraints:**
    - All attributes are not NULL except for description;
    - merchaniseId is a positive integer;
    - price is a non-negative integer;
    - freeDelivery is either ‘Y’ or ‘N’;
    - offeredBy references Sponsors table; it has ON DELETE/UPDATE CASCADE;

Exchange(**merchandiseId**: INTEGER, **buyerId**: INTEGER, tradeDate: DATE, buyerAddress: VARCHAR)

- **Primary Key:** merchaniseId
- **Candidate Keys:** (None)
- **Foreign Keys:** merchaniseId, buyerId
- **Other Constraints:**
    - merchandiseId and tradeDate are not NULL;
    - merchandiseId references Merchandise table; it has ON DELETE/ UPDATE CASCADE;
    - buyerId references Accounts table; it has ON DELETE SET NULL and ON UPDATE CASCADE;

5. Functional Dependencies

Accounts

- userId → userName, email, points, password, level, rank
- email → userId, userName, points, password, level, rank
- level → rank

FavoriteLists

- userId, listName → description

Posts

- postId → postDate, postTime, likes, postedBy, topic

TextPosts

- postId → content

MediaPosts

- postId → hyperlink

Comments

- postId, commentId → commentedBy, content, postDate, postTime

FavoritedPosts

- (No non-trivial FDs)

Topics

- topicName → description, startedBy

Activities

- activityName → description, reward, startDate, endDate

ActivityIncludesTopics

- (No non-trivial FDs)

Communities

- communityName → description, communityGuidelines

Joins

- (No non-trivial FDs)

Sponsors

- companyName → businessDirections, contactInformation, headOffice, country
- headOffice → country

HoldActivities

- (No non-trivial FDs)

Merchandise

- merchaniseId → name, description, price, shippedFrom, freeDelivery, offerdBy

Exchange

- merchandiseId → buyerId, tradeDate, buyerAddress

6. Normalization

Accounts(<u>userId</u>: INTEGER, userName: VARCHAR, email: VARCHAR, points: INTEGER, password: VARCHAR, level: INTEGER, rank: VARCHAR)

- Functional Dependencies:
  - userId → userName, email, points, password, level, rank
  - email → userId, userName, points, password, level, rank
  - level → rank
- Closures:
  - $userId^+$ = {userId, userName, email, points, password, level, rank}
  - $email^+$ = {userId, userName, email, points, password, level, rank}
  - $level^+$ = {level, rank}
- Minimal Cover:
  - Step 1: Put FDs in standard form (have only one attribute on RHS)
    - Break all the FDs with more than one attribute on RHS into FDs with only one attribute on RHS
      - userId → userName
      - userId → email
      - userId → points
      - userId → password
      - userId → level
      - userId → rank
      - email → userId
      - email → userName
      - email → points
      - email → password
      - email → level
      - email → rank
      - level → rank
  - Step 2: Minimize LHS of each FD
    - All FDs' LHS are minimized so no change
      - userId → userName
      - userId → email
      - userId → points
      - userId → password
      - userId → level
      - userId → rank
      - email → userId
      - email → userName
      - email → points
      - email → password
      - email → level
      - email → rank
      - level → rank

        - Step 3: Delete Redundant FDs
            - For email, only keep the FD 'email → userId', and delete FD 'userId → rank'
                - userId → userName
                - userId → email
                - userId → points
                - userId → password
                - userId → level
                - email → userId
                - level → rank
            - Since 'level' is not a superkey of Accounts and 'rank' is not a part of the key, so Accounts is not in BCNF and 3NF.
- Decomposition:
    - Step 1: Decompose Accounts on level → rank:
        - LevelRanks (level, rank)
        - Accounts (userId, userName, email, points, password, level)
        - FDs for both relations all satisfy BCNF.
- Normalization:
    - LevelRanks (level, rank)
    - Accounts (userId, userName, email, points, password, level)

FavoriteLists(**userId**: INTEGER, listName: VARCHAR, description: VARCHAR)

- Functional Dependencies:
    - userId, listName → description
- Closures:
    - userId, listName$^+$ = {userId, listName, description}
- Normalization:
    - Since all the FDs' LHS are super key, this table is in BCNF and 3NF, so we don't need to normalize this table.

Posts(postId: INTEGER, postDate: DATE, postTime: TIME, likes: INTEGER, **postedBy**: INTEGER, **topic**: INTEGER)

- Functional Dependencies:
    - postId → postDate, postTime, likes, postedBy, topic
- Closures:
    - postId$^+$ = {postId, postDate, postTime, likes, postedBy, topic}
- Normalization:
    - Since all the FDs' LHS are super key, this table is in BCNF and 3NF, so we don't need to normalize this table.

TextPosts(**postId**: INTEGER, content: VARCHAR)

- Functional Dependencies:

    - postId → content
- Closure:
    - $postId^+$ = {postId, content}
- Normalization:
    - Since all the FDs' LHS are super key, this table is in BCNF and 3NF, so we don't need to normalize this table.

MediaPosts(**postId**: INTEGER, hyperlink: VARCHAR)

- Functional Dependencies:
    - postId → hyperlink
- Closure:
    - $postId^+$ = {postId, hyperlink}
- Normalization:
    - Since all the FDs' LHS are super key, this table is in BCNF and 3NF, so we don't need to normalize this table.

Comments(**postId**: INTEGER, commentId: INTEGER, **commentedBy**: INTEGER, content: VARCHAR, postDate: DATE, postTime: TIME)

- Functional Dependencies:
    - postId, commentId → commentedBy, content, postDate, postTime
- Closure:
    - postId, $commentId^+$ = {postId, commentId, commentedBy, content, postDate, postTime}
- Normalization:
    - Since all the FDs' LHS are super key, this table is in BCNF and 3NF, so we don't need to normalize this table.

FavoritedPosts(**userId**: INTEGER, **listName**: VARCHAR, **postId**: INTEGER)

- Functional Dependencies:
    - (No non-trivial FDs)
- Closure:
    - (No closure)
- Normalization:
    - (No non-trivial FDs exist so we don't need to normalize)

Topics(topicName: VARCHAR, description: VARCHAR, **startedBy**: INTEGER)

- Functional Dependencies:
    - topicName → description, startedBy
- Closure:
    - $topicName^+$ = {topicName, description, startedBy}
- Normalization:

    - Since all the FDs' LHS are super key, this table is in BCNF and 3NF, so we don't need to normalize this table.

Activities(activityName: VARCHAR, description: VARCHAR, reward: INTEGER, startDate: DATE, endDate: DATE)

- Functional Dependencies:
    - activityName → description, reward, startDate, endDate
- Closure:
    - $activityName^+$ = {activityName, description, reward, startDate, endDate}
- Normalization:
    - Since the FDs are in BCNF and 3NF, so we don't need to normalize this table.

ActivityIncludesTopics(**topicName**: VARCHAR, **activityName**: VARCHAR)

- Functional Dependencies:
    - (No non-trivial FDs)
- Closure:
    - (No closure)
- Normalization:
    - (No non-trivial FDs exist so we don't need to normalize)

Communities(communityName: VARCHAR, description: VARCHAR, communityGuidelines: VARCHAR)

- Functional Dependencies:
    - communityName → description, communityGuidelines
- Closure:
    - $communityName^+$ = {communityName, description, communityGuidelines}
- Normalization:
    - Since all the FDs' LHS are super key, this table is in BCNF and 3NF, so we don't need to normalize this table.

Joins(**communityName**: VARCHAR, **userId**: INTEGER)

- Functional Dependencies:
    - (No non-trivial FDs)
- Closure:
    - (No closure)
- Normalization:
    - (No non-trivial FDs exist so we don't need to normalize)

Sponsors(companyName: VARCHAR, businessDirections: VARCHAR, contactInformation: VARCHAR, headOffice: VARCHAR, country: VARCHAR)

- Functional Dependencies:
    - companyName → businessDirections, contactInformation, headOffice, country
    - headOffice → country
- Closure:
    - companyName$^+$ = {companyName, businessDirections, contactInformation, headOffice, country}
    - headOffice$^+$ = {headOffice, country}
- Minimal Cover:
    - Step 1: Put FDs in standard form (have only one attribute on RHS)
        - Break all the FDs with more than one attribute on RHS into FDs with only one attribute on RHS
            - companyName → businessDirections
            - companyName → contactInformation
            - companyName → headOffice
            - companyName → country
            - headOffice → country
    - Step 2: Minimize LHS of each FD
        - All FDs' LHS are minimized so no change
            - companyName → businessDirections
            - companyName → contactInformation
            - companyName → headOffice
            - companyName → country
            - headOffice → country
    - Step 3: Delete Redundant FDs
        - Delete FD 'companyName → country'
            - companyName → businessDirections
            - companyName → contactInformation
            - companyName → headOffice
            - headOffice → country
        - Since 'headOffice' is not a superkey and 'country' is not a part of the key, so Sponsors is not in BCNF and 3NF.
- Decomposition:
    - Step 1: Decompose Sponsors on headOffice → country:
        - HeadOfficesCountries (headOffice, country)
        - Sponsors (companyName, businessDirections, contactInformation, headOffice)
        - FDs for both relations all satisfy BCNF.
- Normalization:
    - HeadOfficesCountries (<u>headOffice</u>, country)
    - Sponsors (<u>companyName</u>, businessDirections, contactInformation, headOffice)

HoldActivities(**communityName**: VARCHAR, **companyName**: VARCHAR, **activityName**: VARCHAR)

- Functional Dependencies:
  - (No non-trivial FDs)
- Closure:
  - (No closure)
- Normalization:
  - (No non-trivial FDs exist so we don't need to normalize)

Merchandise(merchaniseId: INTEGER, name: VARCHAR, description: VARCHAR, price: INTEGER, shippedFrom: VARCHAR, freeDelivery: CHAR(1), **offerdBy**: VARCHAR)

- Functional Dependencies:
  - merchaniseId → name, description, price, shippedFrom, freeDelivery, offerdBy
- Closure:
  - merchaniseId$^+$ = {merchaniseId, name, description, price, shippedFrom, freeDelivery, offerdBy}
- Normalization:
  - Since all the FDs' LHS are super key, this table is in BCNF and 3NF, so we don't need to normalize this table.

Exchange(**merchandiseId**: INTEGER, **buyerId**: INTEGER, tradeDate: DATE, buyerAddress: VARCHAR)

- Functional Dependencies:
  - merchandiseId → buyerId, tradeDate, buyerAddress
- Closure:
  - merchandiseId$^+$ = {merchandiseId, buyerId, tradeDate, buyerAddress}
- Normalization:
  - Since all the FDs' LHS are super key, this table is in BCNF and 3NF, so we don't need to normalize this table.

7. Table Creations

```
CREATE TABLE LevelRanks(
        level           INTEGER     PRIMARY KEY,
        rank            VARCHAR     NOT NULL);


CREATE TABLE Accounts (
        userID          INTEGER     PRIMARY KEY,
```

```
    userName    VARCHAR    NOT NULL,
    email       VARCHAR    NOT NULL UNIQUE,
    points      INTEGER    NOT NULL,
    password    VARCHAR    NOT NULL,
    level       INTEGER    NOT NULL,
    FOREIGN KEY (level) REFERENCES LevelRanks(level));


CREATE TABLE FavoriteLists (
    userID      INTEGER,
    listName    VARCHAR    NOT NULL,
    description VARCHAR,
    PRIMARY KEY (userID, listName),
    FOREIGN KEY (userID) REFERENCES Accounts (userID)
        ON DELETE CASCADE
        ON UPDATE CASCADE);

CREATE TABLE Topics (
    topicName       VARCHAR    PRIMARY KEY,
    description     VARCHAR    NOT NULL,
    startedBy       INTEGER,
    FOREIGN KEY (startedBy) REFERENCES Accounts (userId)
        ON DELETE SET NULL
        ON UPDATE CASCADE);

CREATE TABLE Posts (
    postId      INTEGER    PRIMARY KEY,
    postDate    DATE       NOT NULL,
    postTime    TIME       NOT NULL,
    likes       INTEGER    NOT NULL,
    postedBy    INTEGER,
    topic       VARCHAR,
    FOREIGN KEY (postedBy) REFERENCES Accounts(userId)
        ON DELETE SET NULL
        ON UPDATE CASCADE,
    FOREIGN KEY (topic) REFERENCES Topics(topicName)
        ON DELETE SET NULL
        ON UPDATE CASCADE);


CREATE TABLE  TextPosts (
    postId      INTEGER    PRIMARY KEY,
```

```
    content         VARCHAR     NOT NULL,
    FOREIGN KEY (postId) REFERENCES Posts(postId)
        ON DELETE CASCADE
        ON UPDATE CASCADE);


CREATE TABLE  MediaPosts (
    postId          INTEGER     PRIMARY KEY,
    hyperlink       VARCHAR     NOT NULL,
    FOREIGN KEY (postId) REFERENCES Posts(postId)
        ON DELETE CASCADE
        ON UPDATE CASCADE);


CREATE TABLE Comments (
    postId              INTEGER,
    commentId           INTEGER,
    commentedBy         INTEGER,
    content             VARCHAR     NOT NULL,
    postDate            DATE        NOT NULL,
    postTime            TIME        NOT NULL,
    PRIMARY KEY (postId, commentId),
    FOREIGN KEY (postId) REFERENCES Posts(postId)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    FOREIGN KEY (commentedBy) REFERENCES Accounts (userId)
        ON DELETE SET NULL
        ON UPDATE CASCADE);


CREATE TABLE FavoritedPosts (
    userId          INTEGER,
    listName        VARCHAR,
    postId          INTEGER,
    PRIMARY KEY (userId, listName, postId),
    FOREIGN KEY (postId) REFERENCES Posts(postId)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    FOREIGN KEY (userId, listName) REFERENCES FavoriteLists (userId, listName)
        ON DELETE CASCADE
        ON UPDATE CASCADE);
```

```
CREATE TABLE Activities (
        activityName        VARCHAR     PRIMARY KEY,
        description         VARCHAR     NOT NULL,
        reward              INTEGER     NOT NULL,
        startDate           DATE        NOT NULL,
        endDate             DATE        NOT NULL);


CREATE TABLE ActivitiyIncludesTopics (
        topicName           VARCHAR,
        activityName        VARCHAR,
        PRIMARY KEY (topicName, activityName),
        FOREIGN KEY (topicName) REFERENCES Topics (topicName)
                ON DELETE CASCADE
                ON UPDATE CASCADE,
        FOREIGN KEY (activityName) REFERENCES Activities (activityName)
                ON DELETE CASCADE
                ON UPDATE CASCADE);


CREATE TABLE Communities (
        communityName           VARCHAR     PRIMARY KEY,
        description             VARCHAR     NOT NULL,
        communityGuidelines     VARCHAR);


CREATE TABLE Joins (
        communityName       VARCHAR,
        userId              INTEGER,
        PRIMARY KEY (communityName, userId),
        FOREIGN KEY (communityName) REFERENCES Communities
(communityName)
                ON DELETE CASCADE
                ON UPDATE CASCADE,
        FOREIGN KEY (userId) REFERENCES Accounts (userId)
                ON DELETE CASCADE
                ON UPDATE CASCADE);


CREATE TABLE HeadOfficesCountries (
        headOffice              VARCHAR     PRIMARY KEY,
```

```
    country                 VARCHAR     NOT NULL);


CREATE TABLE Sponsors (
    companyName             VARCHAR     PRIMARY KEY,
    businessDirections      VARCHAR,
    contactInformation      VARCHAR     NOT NULL,
    headOffice              VARCHAR     NOT NULL,
    FOREIGN KEY (headOffice) REFERENCES HeadOfficesCountries(headOffice)
        ON DELETE CASCADE
        ON UPDATE CASCADE);


CREATE TABLE HoldActivites (
    communityName   VARCHAR,
    companyName     VARCHAR,
    activityName    VARCHAR,
    PRIMARY KEY (communityName, companyName, activityName),
    FOREIGN KEY (communityName) REFERENCES Communities
(communityName)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    FOREIGN KEY (companyName) REFERENCES Sponsors (companyName)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    FOREIGN KEY (activityName) REFERENCES Activities (activityName)
        ON DELETE CASCADE
        ON UPDATE CASCADE);


CREATE TABLE Merchandise (
    merchaniseId        INTEGER     PRIMARY KEY,
    name                VARCHAR     NOT NULL,
    description         VARCHAR,
    price               INTEGER     NOT NULL,
    shippedFrom         VARCHAR     NOT NULL,
    freeDelivery        CHAR(1)     NOT NULL,
    offeredBy           VARCHAR     NOT NULL,
    FOREIGN KEY (offeredBy) REFERENCES Sponsors (companyName)
        ON DELETE CASCADE
        ON UPDATE CASCADE);
```

```
CREATE TABLE Exchange (
        merchaniseId        INTEGER     PRIMARY KEY,
        buyerId             INTEGER,
        tradeDate           DATE        NOT NULL,
        buyerAddress        VARCHAR     NOT NULL,
        FOREIGN KEY (merchaniseId) REFERENCES Merchandise (merchaniseId)
                ON DELETE CASCADE
                ON UPDATE CASCADE,
        FOREIGN KEY (buyerId) REFERENCES Accounts (userId)
                ON DELETE SET NULL
                ON UPDATE CASCADE);
```

8. Insertions

```
--LevelRanks Table

INSERT
INTO        LevelRanks (level, rank)
VALUES      (1, 'Newb');

INSERT
INTO        LevelRanks(level, rank)
VALUES      (2, 'Newb');

INSERT
INTO        LevelRanks (level, rank)
VALUES      (3, 'Newb');

INSERT
INTO        LevelRanks (level, rank)
VALUES      (4, 'Experienced');

INSERT
INTO        LevelRanks (level, rank)
VALUES      (9, 'Renowned');


--Accounts Table

INSERT
```

```
INTO        Accounts (userId, userName, email, points, password, level)
VALUES      (1, 'John', 'John@gmail.com', 10, '1234', 1);

INSERT
INTO        Accounts (userId, userName, email, points, password, level)
VALUES      (2, 'Jack', 'Jack@gmail.com', 11, '1543', 1);

INSERT
INTO        Accounts (userId, userName, email, points, password, level)
VALUES      (3, 'Josh', 'Josh@gmail.com', 1000, '2342', 4);

INSERT
INTO        Accounts (userId, userName, email, points, password, level)
VALUES      (4, 'Joe', 'Joe@gmail.com', 10000, '3934', 9);

INSERT
INTO        Accounts (userId, userName, email, points, password, level, rank)
VALUES      (5, 'James', 'James@gmail.com', 1343, '9999', 3);


--FavoriteLists Table

INSERT
INTO        FavoriteLists(userId, listName, description)
VALUES      (1, 'liked', 'favorited items');

INSERT
INTO        FavoriteLists(userId, listName, description)
VALUES      (2, 'art', 'interesting artworks');

INSERT
INTO        FavoriteLists(userId, listName, description)
VALUES      (3, 'CPSC', 'project ideas');

INSERT
INTO        FavoriteLists(userId, listName, description)
VALUES      (4, 'movies', favorite movies');

INSERT
INTO        FavoriteLists(userId, listName, description)
VALUES      (5, 'school', 'class topics');
```

--Topics Table

```
INSERT
INTO        Topics(topicName, description, startedBy)
VALUES      ('CPSC310', 'questions on CPSC310', 1);

INSERT
INTO        Topics(topicName, description, startedBy)
VALUES      ('CPSC320', 'questions on CPSC320', 2);

INSERT
INTO        Topics(topicName, description, startedBy)
VALUES      ('CPSC221', 'questions on CPSC221', 3);

INSERT
INTO        Topics(topicName, description, startedBy)
VALUES      ('CPSC210', 'questions on CPSC210', 4);

INSERT
INTO        Topics(topicName, description, startedBy)
VALUES      ('CPSC110', 'questions on CPSC110', 5);
```

--Posts Table

```
INSERT
INTO        Posts (postId, postDate, postTime, likes, postedBy, topic)
VALUES      (1, '07/12/2024', '10:23', 51, 1, 1);

INSERT
INTO        Posts (postId, postDate, postTime, likes, postedBy, topic)
VALUES      (2, '08/15/2024', '09:25', 3, 2, 2);

INSERT
INTO        Posts (postId, postDate, postTime, likes, postedBy, topic)
VALUES      (3, '01/21/2024', '12:34', 12, 3, 3);

INSERT
INTO        Posts (postId, postDate, postTime, likes, postedBy, topic)
VALUES      (4, '12/12/2024', '01:59', 11, 4, 4);
```

```
INSERT
INTO        Posts (postId, postDate, postTime, likes, postedBy, topic)
VALUES      (5, '05/12/2025', '00:01', 512, 5, 5);


--TextPosts Table

INSERT
INTO        TextPosts (postId, content)
VALUES      (1, 'hello, I am stuck on question 4 on the midterm practice');

INSERT
INTO        TextPosts (postId, content)
VALUES      (2, 'Top vacation locations in Canada');

INSERT
INTO        TextPosts (postId, content)
VALUES      (3, 'How to land  internship  fall 2024');

INSERT
INTO        TextPosts (postId, content)
VALUES      (4, 'Best ramen place near UBC ');

INSERT
INTO        TextPosts (postId, content)
VALUES      (5, 'Hello world');


--MediaPosts Table

INSERT
INTO        MediaPosts(postId, hyperlink)
VALUES      (5, 'https://www.CPSC320.com/milestone2.jpg');

INSERT
INTO        MediaPosts(postId, hyperlink)
VALUES      (4, 'https://www.CPSC320.com/milestone2.jpg');

INSERT
INTO        MediaPosts(postId, hyperlink)
VALUES      (3, 'https://www.CPSC320.com/assignment3.jpg');
```

```
INSERT
INTO        MediaPosts(postId, hyperlink)
VALUES      (2, 'https://www.ubcea.com/team/players.jpg');

INSERT
INTO        MediaPosts(postId, hyperlink)
VALUES      (1, 'https://www.canvas.ca/profile.jpg');


--Comments Table

INSERT
INTO        Comments(postId, commentId, commentedBy, content, postDate,
postTime)
VALUES      (1, 1, 1, 'cool artwork', '06/13/2024', '12:30');

INSERT
INTO        Comments(postId, commentId, commentedBy, content, postDate,
postTime)
VALUES      (2, 2, 2, 'good job', '12/10/2024', '01:24');

INSERT
INTO        Comments(postId, commentId, commentedBy, content, postDate,
postTime)
VALUES      (3, 3, 3, 'that is amazing', '05/17/2023', '11:31');

INSERT
INTO        Comments(postId, commentId, commentedBy, content, postDate,
postTime)
VALUES      (4, 4, 4, 'I can't solve this question too', '03/23/2024', '07:56');

INSERT
INTO        Comments(postId, commentId, commentedBy, content, postDate,
postTime)
VALUES      (5, 5, 5, 'very interesting idea', '08/11/2024', '08:30');


--FavoritedPosts Table

INSERT
INTO        FavoritedPosts(userId, listName, postId)
VALUES      (1, 'liked', 1);
```

```
INSERT
INTO        FavoritedPosts(userId, listName, postId)
VALUES      (1, 'liked', 2);

INSERT
INTO        FavoritedPosts(userId, listName, postId)
VALUES      (3, 'CPSC', 3);

INSERT
INTO        FavoritedPosts(userId, listName, postId)
VALUES      (3, 'CPSC', 1);

INSERT
INTO        FavoritedPosts(userId, listName, postId)
VALUES      (4, 'movies', 5);



--Activities Table

INSERT
INTO        Activities(activityName, description, reward, startDate, endDate)
VALUES      ('Midterm Practice', ' Mock exam', '100 points', '07/10/2024',
'08/20/2024');

INSERT
INTO        Activities(activityName, description, reward, startDate, endDate)
VALUES      ('Final Practice', 'Mini game on final topics', '500  points', '08/14/2024',
'08/30/2024');

INSERT
INTO        Activities(activityName, description, reward, startDate, endDate)
VALUES      ('Mario Party', ' Last day tournament', '200 points', '08/30/2024',
'08/31/2024');

INSERT
INTO        Activities(activityName, description, reward, startDate, endDate)
VALUES      ('Orientation', ' First day orientation', '10  points', '09/07/2024',
'09/10/2024');

INSERT
```

```
INTO        Activities(activityName, description, reward, startDate, endDate)
VALUES      ('Team building', 'Team building exercise', '150 points', '09/11/2024',
'09/17/2024');


--ActivityIncludesTopics Table

INSERT
INTO        ActivityIncludesTopics(topicName, activityName)
VALUES      ('CPSC310', 'Midterm Practice');

INSERT
INTO        ActivityIncludesTopics(topicName, activityName)
VALUES      ('CPSC320", 'Final Practice');

INSERT
INTO        ActivityIncludesTopics(topicName, activityName)
VALUES      ('CPSC221', 'Mario Party');

INSERT
INTO        ActivityIncludesTopics(topicName, activityName)
VALUES      ('CPSC210', 'Orientation');

INSERT
INTO        ActivityIncludesTopics(topicName, activityName)
VALUES      ('CPSC110', 'Team building');


--Communities Table

INSERT
INTO        Communities(communityName, description, communityGuidelines)
VALUES      ('Student Center', 'Student discussion group', 'Adhere to academic
integrity');

INSERT
INTO        Communities(communityName, description, communityGuidelines)
VALUES      ('BC Technicians', 'Technical discussion forum', 'Beware of confidential
info');

INSERT
INTO        Communities(communityName, description, communityGuidelines)
```

```
VALUES ('Vancouver Hackathons', 'Hackathon activities in Vancouver', 'No cheating');

INSERT
INTO Communities(communityName, description, communityGuidelines)
VALUES ('Random Posts', 'Just random stuff', 'Be friendly :)');

INSERT
INTO Communities(communityName, description, communityGuidelines)
VALUES ('Formal announcements BC', 'Announcements to BC citizens', 'No violence');


--Joins Table

INSERT
INTO Joins(communityName, userId)
VALUES ('Formal announcements BC', 1);

INSERT
INTO Joins(communityName, userId)
VALUES ('Formal announcements BC', 2);

INSERT
INTO Joins(communityName, userId)
VALUES ('Vancouver Hackathons', 1);

INSERT
INTO Joins(communityName, userId)
VALUES ('Random Posts', 4);

INSERT
INTO Joins(communityName, userId)
VALUES ('BC Technicians', 2);


--HeadOfficesCountries

INSERT
INTO HeadOfficesCountries(headOffice,country)
VALUES ('Vancouver', 'Canada');
```

```
INSERT
INTO        HeadOfficesCountries(headOffice,country)
VALUES      ('Toronto', 'Canada');

INSERT
INTO        HeadOfficesCountries(headOffice,country)
VALUES      ('Delhi', 'India');

INSERT
INTO        HeadOfficesCountries(headOffice,country)
VALUES      ('Shanghai', 'China');

INSERT
INTO        HeadOfficesCountries(headOffice,country)
VALUES      ('New York', 'USA');


--Sponsors

INSERT
INTO        Sponsors (companyName, businessDirections, contactInformation,
headOffice)
VALUES      ('Company A', 'Electronics', '123-456-7890', 'Vancouver');

INSERT
INTO        Sponsors (companyName, businessDirections, contactInformation,
headOffice)
VALUES      ('Company B', 'Biomechanics', '456-789-0123', 'Toronto');

INSERT
INTO        Sponsors (companyName, businessDirections, contactInformation,
headOffice)
VALUES      ('Company C', 'IT', '135-790-2468', 'Delhi');

INTO        Sponsors (companyName, businessDirections, contactInformation,
headOffice)
VALUES      ('Company D', 'Communication', '123-456-78901', 'Shanghai');

INSERT
INTO        Sponsors (companyName, businessDirections, contactInformation,
headOffice)
VALUES      ('Company E', NULL, '246-801-3579', 'New York');
```

--HoldActivities Table

```
INSERT
INTO        HoldActivities(communityName, companyName, activityName)
VALUES      ('Student Center', 'Company E', 'Mario Party);

INSERT
INTO        HoldActivities(communityName, companyName, activityName)
VALUES      ('BC Technicians', 'Company B', 'Orientation');

INSERT
INTO        HoldActivities(communityName, companyName, activityName)
VALUES      ('BC Technicians', 'Company C', 'Midterm Practice');

INSERT
INTO        HoldActivities(communityName, companyName, activityName)
VALUES      ('Student Center', 'Company A', 'Final Practice');

INSERT
INTO        HoldActivities(communityName, companyName, activityName)
VALUES      ('Student Center', 'Company A', 'Team building');
```

--Merchandise Table

```
INSERT
INTO        Merchandise (merchaniseId, name, description, price, shippedFrom,
freeDelivery, offerdBy)
VALUES      (1, 'Pen', 'Red pen', 15, 'Vancouver', 'Y', 'Company A');

INSERT
INTO        Merchandise (merchaniseId, name, description, price, shippedFrom,
freeDelivery, offerdBy)
VALUES      (2, 'Pencil', 'Charcoal pencil', 10, 'Richmond', 'N', 'Company A');

INSERT
INTO        Merchandise (merchaniseId, name, description, price, shippedFrom,
freeDelivery, offerdBy)
VALUES      (3, 'Notebook', 'For sketching', 20, 'Burnaby', 'N', 'Company B');
```

```
INSERT
INTO        Merchandise (merchaniseId, name, description, price, shippedFrom,
freeDelivery, offerdBy)
VALUES      (4, 'Eraser', NULL, 5, 'Victoria', 'Y', 'Company C');

INSERT
INTO        Merchandise (merchaniseId, name, description, price, shippedFrom,
freeDelivery, offerdBy)
VALUES      (5, 'Correction Tape', '50m * 10mm', 15, 'Richmond', 'N', 'Company D');


--Exchange Table

INSERT
INTO        Exchange(merchandiseId, buyerId, tradeDate, buyerAddress)
VALUES      (1, 1, '07/21/2024', '1234 AB Road, Vancouver BC');

INSERT
INTO        Exchange(merchandiseId, buyerId, tradeDate, buyerAddress)
VALUES      (2, 3, '07/20/2024', '5678 CD Road, Vancouver BC');

INSERT
INTO        Exchange(merchandiseId, buyerId, tradeDate, buyerAddress)
VALUES      (3, 4, '07/19/2024', '9012 EF Road, Richmond BC');

INSERT
INTO        Exchange(merchandiseId, buyerId, tradeDate, buyerAddress)
VALUES      (4, 5, '07/18/2024', '1415 PI Road, Toronto ON');

INSERT
INTO        Exchange(merchandiseId, buyerId, tradeDate, buyerAddress)
VALUES      (5, 1, '07/17/2024', '9999 GH Road, Burnaby BC');
```