University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: 02/28/2024

Group Number: 67

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Zongxi Li	40628778	zli110	zongxiubc@gmail.com
Shiyu Deng	29343480	b1l9z	simondeng.sy@gmail.comn
Marie Samantha Fidelia	64717309	g0j5n	mariesamantha.f@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 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

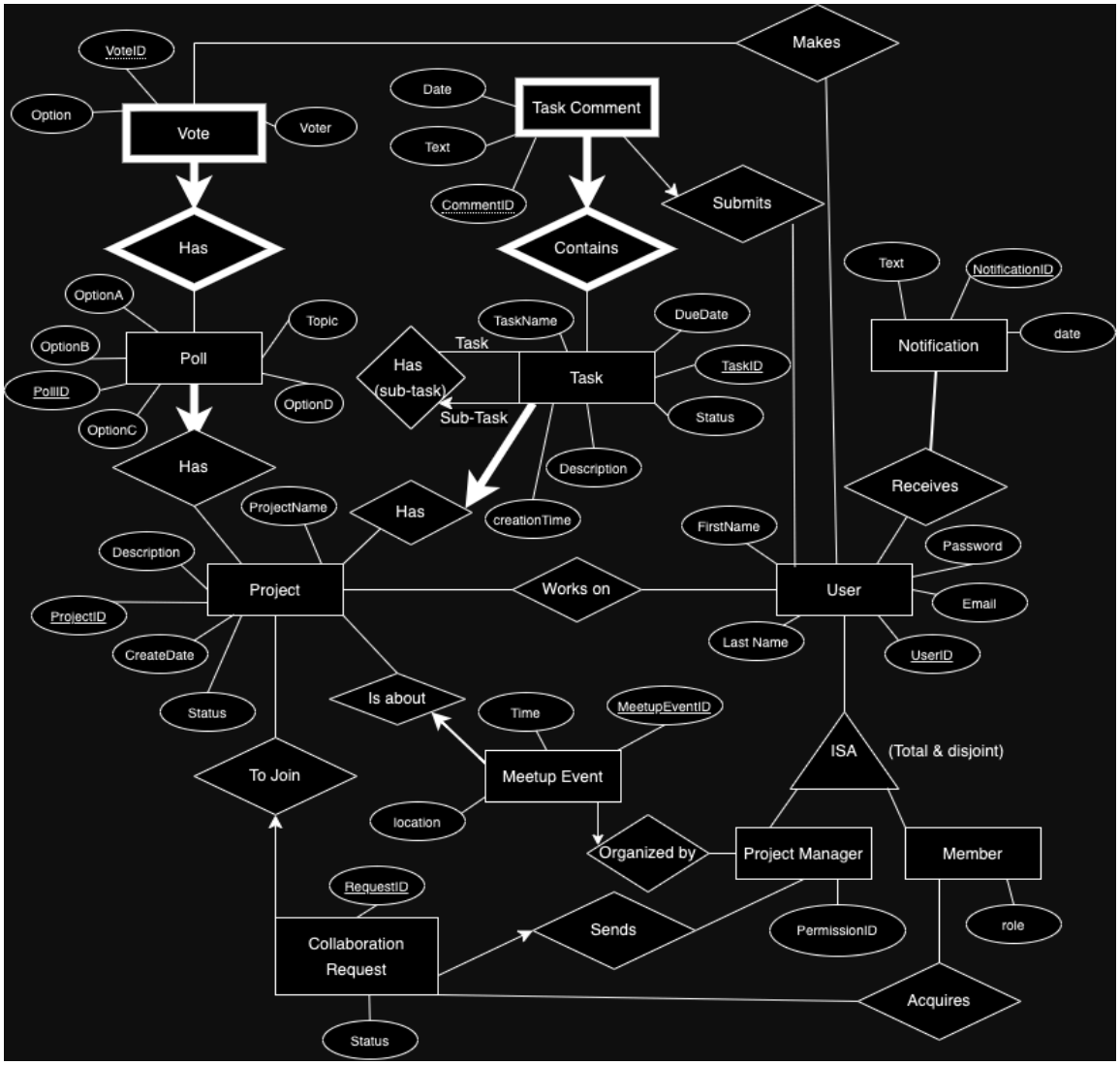
University of British Columbia, Vancouver
Department of Computer Science

Project Collaboration Web App

Brief Summary

Our project is a project collaboration and management platform with features like task assignment, progress tracking, and communication tools. It utilizes a database to efficiently manage user authentication, project details, and real-time collaboration, to enhance productivity and coordination within teams. The database will provide functionality for storing, retrieving, and managing data related to project collaboration and management.

ER Diagram (replace below with the revised diagram)



University of British Columbia, Vancouver
Department of Computer Science

Changes we've made from the ER design in Milestone1:

- Changed the PK PermissionID in the entity Project Manager to a normal attribute;
- Added a new many to many relationship Makes between entities Vote and User;
- Added a new one to many relationship Submits between entities User and Task Comment;
- Removed PK of To Join relationship (RequestID and ProjectID);
- Removed PK of To Receives relationship (UserID and NotificationID);
- Removed Partial Keys ProjectID and TaskID in weak entity TaskComment;
- Removed PK ProjectID in entity Task;
- Removed PKs TaskID and SubTaskID from the self-referencing Has(sub task) relationship;
- Removed PK of Work on relationship (UserID and ProjectID);
- Removed PK ProjectID in entity Meetup Event;
- Removed PK SenderPermisionID and Project in entity Collaboration Request;
- Removed Partial Keys ProjectID and PollID in weak entity Vote;
- Removed PK ProjectID in entity Poll;
- Removed attribute ReceiverID from entity Collaboration Request;
- Changed name of the Receives Relationship between Member and Request to "Acquires" to avoid duplicate relationship names;
- Changed MeetingEventID in entity Meetup Event to "MeetupEventID;"
- Removed the UserID attribute in the Task Comment entity;
- Removed attribute assignee from Task entity;

Schema

- Notification(NotificationID: Char(50), Text: Char(255), Date: TimeStamp)
    - CK: NotificationID
    - Date: not null

- User(UserID: Char(50), FirstName: Char(50), LastName: Char(50), Password: Char(50), Email: Char(50))
    - CK: UserID
    - Email: not null and unique
    - Password: not null

- Project Manager(**UserID**: Char(50), PermissionID: Char(50))
    - CK: UserID
    - PermissionID: not null

- Member(**UserID**: Char(50), Role: Char(50))
    - CK: UserID

- Collaboration Request(RequestID: Char(50), **SenderID**: Char(50), **SenderPermissionID**: Char(50), **ProjectID**: Char(50), Status: Boolean)
    - CK: RequestID
    - SenderID: not null
    - SenderPermissionID: not null
    - ProjectID: not null

- Project(ProjectID: Char(50), ProjectName: Char(50), Description: Char(255), CreateDate: TimeStamp, Status: Char(25))
    - CK: ProjectID
    - ProjectName: not null and unique
    - CreateDate: not null

- Task(TaskID: Char(50), TaskName: Char(50), **ProjectID**: Char(50), DueDate: TimeStamp, Statue: Boolean, Description: Char(255), CreateTime: TimeStamp, **SubTaskID**: Char(50))
    - CK: TaskID
    - TaskName: not null
    - ProjectID: not null

- Meetup Event(MeetupEventID: Char(50), **ProjectID**: Char(50), Location: Char(50), Time: TimeStamp, **UserID**: Char(50), **PermissionID**: Char(50))
    - CK: MeetingEventID
    - ProjectID: not null
    - UserID: not null
    - PermissionID: not null

- Poll(PollID: Char(50), **ProjectID**: Char(50), Topic: Char(255), Option A: Char(50), Option B: Char(50), Option C: Char(50), Option D: Char(50))
    - CK: PollID
    - ProjectID: not null
    - Topic: not null

- Vote_Has(VoteID: Char(50), **PollID**: Char(50), Option: Char(50))
    - CK: the combination of VoteID and PollID
    - Option: not null

- Task Comment_Contains(CommentID: Char(50), **TaskID**: Char(50), Date: TimeStamp, Text: Char(255), **UserID**: Char(50))
    - CK: the combination of CommentID and TaskID
    - UserID: not null

- Receives(**NotificationID**: Char(50), **UserID**: Char(50))

- CK: the combination of NotificationID and UserID

- Acquires(**UserID**: Char(50), **RequestID**: Char(50))
  - CK: the combination UserID and RequestID

- WorksOn(**ProjectID**: Char(50), **UserID**: Char(50))
  - CK: the combination of ProjectID and UserID

- Makes(**UserID**: Char(50), **VoteID**: Char(50))
  - CK: the combination of UserID and VoteID

Functional Dependencies (FDs)

In the relation of the Entity Notification:
- NotificationID -> Text
- NotificationID -> Date

In the relation of the Entity User:
- UserID -> FirstName
- UserID -> LastName
- UserID -> Email
- UserID -> Password

- Email -> FirstName (Because each email only belongs to one person)
- Email -> LastName
- Email -> Password

In the relation of the Entity ProjectManager:
- UserID -> PermissionID

In the relation of the Entity Member:
- UserID -> Role

In the relation of the Entity Collaboration Request:
- RequestID -> SenderID
- RequestID -> SenderPermissionID
- RequestID -> ProjectID
- RequestID -> Status

In the relation of the Entity Project:
- ProjectID -> Description
- ProjectID -> ProjectName

- ProjectID -> CreateDate
- ProjectID -> Status
- Project Name -> Description
- Project Name -> CreateDate

In the relation of the Entity Task:
- TaskID -> TaskName
- TaskID -> ProjectID
- TaskID -> DueDate
- TaskID -> Status
- TaskID -> Description
- TaskID -> CreateTime
- TaskID -> SubTaskID

In the relation of the Entity Meetup Event:
- MeetupEventID -> ProjectID
- MeetupEventID -> Location
- MeetupEventID -> Time
- MeetupEventID -> UserID
- MeetupEventID -> PermissionID

In the relation of the Entity Poll:
- PollID -> ProjectID
- PollID -> Topic
- PollID -> Option A
- PollID -> Option B
- PollID -> Option C
- PollID -> Option D

In the relation of the Weak Entity Vote and its relationship Has:
- VoteID, PollID -> Option

In the relation of the Weak Entity Task and its relationship Contains:
- CommentID, TaskID -> Date
- CommentID, TaskID -> Text
- CommentID, TaskID -> UserID

In the relation of the Relationship Receives:
- No non-trivial FDs

In the relation of the Relationship Acquires:
- No non-trivial FDs

In the relation of the Relationship WorksOn:
- No non-trivial FDs

In the relation of the Relationship Makes:
- No non-trivial FDs

Normalization

There are 2 relational tables that do not follow BCNF, User and Project.

User(UserID: Char(50), FirstName: Char(50), LastName: Char(50), Password: Char(50), Email: Char(50))

Non-PK/CK FDs in this relation:
- Email -> FirstName
- Email -> LastName
- Email -> Password

Email -> FirstName violates 3NF, so the table User is not in 3NF, decompose to BCNF
- R1(Email, FirstName), R2(UserID,, Email, LastName, Password)

Email -> LastName violates 3NF, so the table R2 is not in 3NF, decompose to BCNF
- R3(Email, LastName), R4(UserID, Email, Password)

Email -> Password violates 3NF, so the table R4 is not in 3NF, decompose to BCNF
- R5(Email, Password), R6(UserID, Email)

So the tables of User look like: R1(Email, FirstName), R3(Email, LastName), R5(Email, Password), R6(UserID, Email)

Project(ProjectID: Char(50), ProjectName: Char(50), Description: Char(255), CreateDate: TimeStamp, Status: Char(25))

Non-PK/CK FDs in the Project relation:
- Project Name -> Description
- Project Name -> CreateDate

Project Name -> Description violates 3NF, so the table Project is not in 3NF, decompose to BCNF
- R1(Project Name, Description), R2(ProjectID, ProjectName, CreateDate, Status)

Project Name -> CreateDate violates 3NF, so the table R2 is not in 3NF, decompose to BCNF

- R3(Project Name, CreateDate), R4(ProjectID, ProjectName, Status)

So the tables of Project look like: R1(Project Name, Description), R3(Project Name, CreateDate), R4(ProjectID, ProjectName, Status)

SQL DDL Statements & INSERT Statements

```
CREATE TABLE Notification (
        NotificationID CHAR(50),
        Text CHAR(255),
        Date TIMESTAMP NOT NULL,
        PRIMARY KEY (NotificationID)
)
```

```
INSERT
INTO Notification (NotificationID, Text, Date)
VALUES ('n1', 'Created a new poll', CURRENT_TIMESTAMP),
('n2', 'New comment in project', CURRENT_TIMESTAMP),
('n3', 'New user added to project', CURRENT_TIMESTAMP),
('n4', 'Poll Closed', CURRENT_TIMESTAMP),
('n5', 'A user left the project', CURRENT_TIMESTAMP),
('n6', 'New task available', CURRENT_TIMESTAMP)
```

```
CREATE TABLE CollaborationRequest (
        RequestID CHAR(50),
        SenderID CHAR(50) NOT NULL,
        SenderPermissionID CHAR(50) NOT NULL,
        ProjectID CHAR(50) NOT NULL,
        Status BOOLEAN,
        PRIMARY KEY (RequestID),
        FOREIGN KEY (SenderID) REFERENCES ProjectManager(UserID)
        FOREIGN KEY (ProjectID) REFERENCES Project(ProjectID)
        FOREIGN KEY (SenderPermissionID) REFERENCE ProjectManager(PermissionID)
)
```

```
INSERT
INTO CollaborationRequest (RequestID, SenderID, SenderPermissionID, ProjectID, Status)
VALUES ('req1', SELECT UserID from ProjectManager WHERE UserID = 'user1', SELECT
SenderPermissionID from ProjectManager WHERE PermissionID = 'admin1', SELECT
ProjectID from Project WHERE ProjectID = 'pro1', TRUE),
```

('req2', SELECT UserID from ProjectManager WHERE UserID = 'user1', SELECT SenderPermissionID from ProjectManager WHERE PermissionID = 'admin2', SELECT ProjectID from Project WHERE ProjectID = 'pro1', TRUE),
('req3', SELECT UserID from ProjectManager WHERE UserID = 'user2', SELECT SenderPermissionID from ProjectManager WHERE PermissionID = 'admin1', SELECT ProjectID from Project WHERE ProjectID = 'pro1', TRUE),
('req4', SELECT UserID from ProjectManager WHERE UserID = 'user1', SELECT SenderPermissionID from ProjectManager WHERE PermissionID = 'admin1', SELECT ProjectID from Project WHERE ProjectID = 'pro1', FALSE),
('req5', SELECT UserID from ProjectManager WHERE UserID = 'user2', SELECT SenderPermissionID from ProjectManager WHERE PermissionID = 'admin1', SELECT ProjectID from Project WHERE ProjectID = 'pro2', FALSE)

CREATE TABLE Task (
TaskID CHAR(50),
TaskName CHAR(50) NOT NULL,
ProjectID CHAR(50) NOT NULL,
DueDate TIMESTAMP,
Status BOOLEAN,
Description CHAR(255),
CreateTime TIMESTAMP,
SubTaskID CHAR(50),
PRIMARY KEY (TaskID),
FOREIGN KEY (ProjectID) REFERENCES Project(ProjectID)
FOREIGN KEY (SubTaskID) REFERENCES Task(SubTaskID)
)

INSERT
INTO Task(TaskID, TaskName, ProjectID, DueDate, Status, Description, CreateTime, SubTaskID)
VALUES ('task1', 'Vote', SELECT ProjectID from Project WHERE ProjectID = 'pro1', '2024-03-05 12:12:12', TRUE, 'Create a new poll for our theme', CURRENT_TIMESTAMP, SELECT SubTaskID from Task WHERE SubTaskID= 'subtask1'),
('task2', 'Diagram', SELECT ProjectID from Project WHERE ProjectID = 'pro1', '2024-03-05 12:12:12', TRUE, 'Finish project diagram', CURRENT_TIMESTAMP, SELECT SubTaskID from Task WHERE SubTaskID= 'subtask1'),
('task3', 'Finalize', SELECT ProjectID from Project WHERE ProjectID = 'pro2', '2024-03-05 12:12:12', TRUE, 'Finalize project details', CURRENT_TIMESTAMP, SELECT SubTaskID from Task WHERE SubTaskID= 'subtask2'),

('task4', 'Revise', SELECT ProjectID from Project WHERE ProjectID = 'pro2', '2024-03-05 12:12:12', FALSE, 'Revise conclusion statement', CURRENT_TIMESTAMP, SELECT SubTaskID from Task WHERE SubTaskID= 'subtask3'),
('task5', 'Analysis', SELECT ProjectID from Project WHERE ProjectID = 'pro1', '2024-03-05 12:12:12', FALSE, 'Do and finish analysis section', CURRENT_TIMESTAMP, SELECT SubTaskID from Task WHERE SubTaskID= 'subtask3')

CREATE TABLE MeetupEvent (
MeetupEventID CHAR(50),
ProjectID CHAR(50) NOT NULL,
Location CHAR(50),
Time TIMESTAMP,
UserID CHAR(50) NOT NULL,
PermissionID CHAR(50) NOT NULL,
PRIMARY KEY (MeetupEventID),
FOREIGN KEY (ProjectID) REFERENCES Project(ProjectID),
FOREIGN KEY (UserID) REFERENCES ProjectManager(UserID),
FOREIGN KEY (PermissionID) REFERENCES ProjectManager(PermissionID)
)

INSERT
INTO MeetupEvent(MeetupEventID, ProjectID, Location, Time, UserID, PermissionID)
VALUES ('meetup1', SELECT ProjectID from Project WHERE ProjectID = 'pro1', 'BC Place', '2024-03-05 12:12:12', SELECT UserID from ProjectManager WHERE UserID = 'user1', SELECT PermissionID from ProjectManager WHERE PermissionID = 'admin1'),
('meetup2', SELECT ProjectID from Project WHERE ProjectID = 'pro2', 'Surrey', '2024-03-12 12:12:12', SELECT UserID from ProjectManager WHERE UserID = 'user2', SELECT PermissionID from ProjectManager WHERE PermissionID = 'admin1'),
('meetup3', SELECT ProjectID from Project WHERE ProjectID = 'pro2', 'Surrey','2024-03-23 12:12:12', SELECT UserID from ProjectManager WHERE UserID = 'user2', SELECT PermissionID from ProjectManager WHERE PermissionID = 'admin2'),
('meetup4', SELECT ProjectID from Project WHERE ProjectID = 'pro2', 'Richmond','2024-03-30 05:12:12', SELECT UserID from ProjectManager WHERE UserID = 'user2', SELECT PermissionID from ProjectManager WHERE PermissionID = 'admin2'),
('meetup5', SELECT ProjectID from Project WHERE ProjectID = 'pro2', 'UBC','2024-04-01 09:12:12', SELECT UserID from ProjectManager WHERE UserID = 'user2', SELECT PermissionID from ProjectManager WHERE PermissionID = 'admin3'),

CREATE TABLE Poll (

```
        PollID CHAR(50),
        ProjectID CHAR(50) NOT NULL,
        Topic CHAR(255) NOT NULL,
        OptionA CHAR(50),
        OptionB CHAR(50),
        OptionC CHAR(50),
        OptionD CHAR(50),
        PRIMARY KEY (PollID),
        FOREIGN KEY(ProjectID) REFERENCES Project(ProjectID)
)
```

```
INSERT
INTO Poll(PollID, ProjectID, Topic, OptionA, OptionB, OptionC, OptionD)
VALUES ('poll1', SELECT ProjectID from Project WHERE ProjectID = 'pro1', 'Theme', '10', '10',
'50', '30'),
('poll2', SELECT ProjectID from Project WHERE ProjectID = 'pro1', 'Theme', '100', '0', '0', '0'),
('poll3', SELECT ProjectID from Project WHERE ProjectID = 'pro1', 'Theme', '0', '0', '0', '0'),
('poll4', SELECT ProjectID from Project WHERE ProjectID = 'pro2', 'Theme', '25', '25', '25', '25'),
('poll5', SELECT ProjectID from Project WHERE ProjectID = 'pro3', 'Theme', '10', '10', '50', '30')
```

```
CREATE TABLE Vote_Has (
        VoteID CHAR(50),
        PollID CHAR(50),
        Option CHAR(50) NOT NULL,
        PRIMARY KEY (VoteID, PollID),
        FOREIGN KEY (PollID) REFERENCES Poll(PollID)
)
```

```
INSERT
INTO Vote_Has(VoteID, PollID, Option)
VALUES ('vote1', SELECT PollID from Poll WHERE PollID='poll1', 'OptionA'),
('vote2', SELECT PollID from Poll WHERE PollID='poll2', 'OptionA'),
('vote3', SELECT PollID from Poll WHERE PollID='poll1', 'OptionB'),
('vote4', SELECT PollID from Poll WHERE PollID='poll1', 'OptionC'),
('vote5', SELECT PollID from Poll WHERE PollID='poll1', 'OptionD')
```

```
CREATE TABLE Task_Comment_Contains (
        CommentID CHAR(50),
        TaskID CHAR(50),
        Date TIMESTAMP,
```

```
        Text CHAR(255),
        UserID CHAR(50) NOT NULL,
        PRIMARY KEY (CommentID, TaskID),
        FOREIGN KEY (TaskID) REFERENCES Task(TaskID),
        FOREIGN KEY (UserID) REFERENCES User(UserID)
)

INSERT
INTO Task_Comment_Contains(CommentID, TaskID, Date, Text, UserID)
VALUES ('comment1', SELECT TaskID from Task WHERE TaskID = 'task1',
CURRENT_TIMESTAMP, 'Needs revision', SELECT UserID from User WHERE
UserID='user1'),
('comment2', SELECT TaskID from Task WHERE TaskID = 'task1', CURRENT_TIMESTAMP,
'Great job!', SELECT UserID from User WHERE UserID='user2'),
('comment3', SELECT TaskID from Task WHERE TaskID = 'task2', CURRENT_TIMESTAMP,
'Looks good', SELECT UserID from User WHERE UserID='user1'),
('comment4', SELECT TaskID from Task WHERE TaskID = 'task2', CURRENT_TIMESTAMP,
'Please change', SELECT UserID from User WHERE UserID='user2'),
('comment5', SELECT TaskID from Task WHERE TaskID = 'task3', CURRENT_TIMESTAMP,
'Remove this', SELECT UserID from User WHERE UserID='user3')


CREATE TABLE Receives (
        NotificationID CHAR(50),
        UserID CHAR(50),
        PRIMARY KEY (NotificationID, UserID),
        FOREIGN KEY (NotificationID) REFERENCES Notification(NotificationID),
        FOREIGN KEY (UserID) REFERENCES User(UserID)
)

INSERT
INTO Receives(NotificationID, UserID)
VALUES (SELECT NotificationID from Notification WHERE NotificationID='n1', SELECT UserID
from Makes WHERE UserID='user1'),
(SELECT NotificationID from Notification WHERE NotificationID='n1', SELECT UserID from
Makes WHERE UserID='user2'),
(SELECT NotificationID from Notification WHERE NotificationID='n2', SELECT UserID from
Makes WHERE UserID='user3'),
(SELECT NotificationID from Notification WHERE NotificationID='n2', SELECT UserID from
Makes WHERE UserID='user4'),
(SELECT NotificationID from Notification WHERE NotificationID='n3', SELECT UserID from
Makes WHERE UserID='user5')
```

```
CREATE TABLE Acquires (
        UserID CHAR(50),
        RequestID CHAR(50),
        PRIMARY KEY (UserID, RequestID),
        FOREIGN KEY (UserID) REFERENCES Member(UserID),
        FOREIGN KEY (RequestID) REFERENCES CollaborationRequest(RequestID)
)

INSERT
INTO Acquires(UserID, RequestID)
VALUES (SELECT UserID from Member WHERE UserID='user1', SELECT RequestID from
CollaborationRequest WHERE RequestID='request1'),
(SELECT UserID from Member WHERE UserID='user1', SELECT RequestID from
CollaborationRequest WHERE RequestID='request2'),
(SELECT UserID from Member WHERE UserID='user2', SELECT RequestID from
CollaborationRequest WHERE RequestID='request3'),
(SELECT UserID from Member WHERE UserID='user3', SELECT RequestID from
CollaborationRequest WHERE RequestID='request4'),
(SELECT UserID from Member WHERE UserID='user4', SELECT RequestID from
CollaborationRequest WHERE RequestID='request5')
```

```
CREATE TABLE WorksOn (
        ProjectID CHAR(50),
        UserID CHAR(50),
        PRIMARY KEY (ProjectID, UserID),
        FOREIGN KEY (ProjectID) REFERENCES Project(ProjectID),
        FOREIGN KEY (UserID) REFERENCES User(UserID)
)

INSERT
INTO WorksOn(ProjectID, UserID)
VALUES (SELECT ProjectID from Project WHERE ProjectID='pro1', SELECT UserID from
Member WHERE UserID='user1'),
(SELECT ProjectID from Project WHERE ProjectID='pro2', SELECT UserID from Member
WHERE UserID='user2'),
(SELECT ProjectID from Project WHERE ProjectID='pro1', SELECT UserID from Member
WHERE UserID='user3'),
(SELECT ProjectID from Project WHERE ProjectID='pro2', SELECT UserID from Member
WHERE UserID='user4'),
```

(SELECT ProjectID from Project WHERE ProjectID='pro1', SELECT UserID from Member WHERE UserID='user4')

```
CREATE TABLE Makes (
        UserID CHAR(50),
        VoteID CHAR(50),
        PRIMARY KEY (UserID, VoteID),
        FOREIGN KEY (UserID) REFERENCES User(UserID),
        FOREIGN KEY (VoteID) REFERENCES Vote_Has(VoteID)
)
```

```
INSERT
INTO Makes (UserID, VoteID)
VALUES (SELECT UserID from Member WHERE UserID='user1', SELECT VoteID from
Vote_Has WHERE VoteID='vote1'),
(SELECT UserID from Member WHERE UserID='user1', SELECT VoteID from Vote_Has
WHERE VoteID='vote2'),
(SELECT UserID from Member WHERE UserID='user2', SELECT VoteID from Vote_Has
WHERE VoteID='vote1'),
(SELECT UserID from Member WHERE UserID='user2', SELECT VoteID from Vote_Has
WHERE VoteID='vote2'),
(SELECT UserID from Member WHERE UserID='user3', SELECT VoteID from Vote_Has
WHERE VoteID='vote1')
```

```
CREATE TABLE Project (
        ProjectID CHAR(50),
        ProjectName CHAR(50) NOT NULL,
        Status CHAR(25),
        PRIMARY KEY (ProjectID),0
        UNIQUE (ProjectName)
)
```

```
INSERT
INTO Project(ProjectID, ProjectName, Status)
VALUES ('pro1', 'Statistical Inference', 'Complete'),
 ('pro1', 'Statistical Inference, 'Incomplete'),
 ('pro2', 'Relational Database, 'Complete'),
 ('pro3', 'Crypto Analysis, 'Complete'),
 ('pro4', 'Cryptic Analysis, 'Complete'),
 ('pro5', 'Cryptography Analysis, 'Incomplete'),
```

```
CREATE TABLE Project1 (
        ProjectName CHAR(50),
        Description CHAR(255),
        PRIMARY KEY (ProjectName),
        FOREIGN KEY(ProjectName) REFERENCES Project(ProjectName)
)
```

```
INSERT
INTO Project1 (ProjectName, Description)
VALUES (SELECT ProjectName from Project WHERE ProjectName='Crypto Analysis',
'In-depth analysis of cryptocurrency trends'),
(SELECT ProjectName from Project WHERE ProjectName='Statistical Inference, 'STAT 201
Project'),
(SELECT ProjectName from Project WHERE ProjectName='Relational Database', 'CPSC 304
Milestone 2'),
(SELECT ProjectName from Project WHERE ProjectName='Cryptic Analysis, 'Analysis of
cryptic information'),
(SELECT ProjectName from Project WHERE ProjectName='Cryptography Analysis',
'Cryptographic tools')
```

```
CREATE TABLE Project2 (
        ProjectName CHAR(50),
        Time TIMESTAMP,
        PRIMARY KEY (ProjectName)
        FOREIGN KEY(ProjectName) REFERENCES Project(ProjectName)
)
```

```
INSERT
INTO Project2(ProjectName, Time)
VALUES (SELECT ProjectName from Project WHERE ProjectName='Statistical Inference',
'19:35'),
(SELECT ProjectName from Project WHERE ProjectName='Relational Database',
CURRENT_TIMESTAMP),
(SELECT ProjectName from Project WHERE ProjectName='Crypto Analysis',
CURRENT_TIMESTAMP),
(SELECT ProjectName from Project WHERE ProjectName='Cryptic Analysis,
CURRENT_TIMESTAMP),
(SELECT ProjectName from Project WHERE ProjectName='Cryptography Analysis,
CURRENT_TIMESTAMP)
```

```
CREATE TABLE User (
   UserID CHAR(50) PRIMARY KEY,
   Email CHAR(50) NOT NULL UNIQUE
```

)

INSERT
INTO User(UserID, Email)
VALUES (‘userone’, ‘emailone@gmail.com’),
(‘usertwo’, ‘emailtwo@gmail.com’),
(‘userthree’, ‘emailthree@gmail.com’),
(‘userfour’, ‘emailfour@gmail.com’),
(‘userfive’, ‘emailfive@gmail.com’)

CREATE TABLE User1 (
Email CHAR(50)
Password CHAR(50) NOT NULL
PRIMARY KEY Email,
FOREIGN KEY(Email) REFERENCES User(Email)
)

INSERT
INTO User1(Email, Password)
VALUES (SELECT Email from User WHERE Email=’emailone@gmail.com’, ‘passwordone’),
(SELECT Email from User WHERE Email=’emailtwo@gmail.com’, ‘passwordtwo’),
(SELECT Email from User WHERE Email=’emailthree@gmail.com’, ‘passwordthree’),
(SELECT Email from User WHERE Email=’emailfour@gmail.com’, ‘passwordfour’),
(SELECT Email from User WHERE Email=’emailfive@gmail.com’, ‘passwordfive’),

CREATE TABLE User2 (
Email CHAR(50)
FirstName CHAR(50) NOT NULL
PRIMARY KEY Email,
FOREIGN KEY(Email) REFERENCES User(Email)
)

INSERT
INTO USER2(Email, FirstName)
VALUES (SELECT Email from User WHERE Email=’emailone@gmail.com’, ‘john’),
(SELECT Email from User WHERE Email=’emailtwo@gmail.com’, ‘adam’),
(SELECT Email from User WHERE Email=’emailthree@gmail.com’, ‘doe’),
(SELECT Email from User WHERE Email=’emailfour@gmail.com’, ‘smith’),
(SELECT Email from User WHERE Email=’emailfive@gmail.com’, ‘jason’)

CREATE TABLE User3 (

```
        Email CHAR(50)
        LastName CHAR(50) NOT NULL
        PRIMARY KEY Email,
        FOREIGN KEY(Email) REFERENCES User(Email)
)

INSERT
INTO USER3(Email, LastName)
VALUES (SELECT Email from User WHERE Email='emailone@gmail.com', 'johnny'),
(SELECT Email from User WHERE Email='emailtwo@gmail.com', 'levine'),
(SELECT Email from User WHERE Email='emailthree@gmail.com', 'deer'),
(SELECT Email from User WHERE Email='emailfour@gmail.com', 'sam'),
(SELECT Email from User WHERE Email='emailfive@gmail.com', 'momoa')

CREATE TABLE ProjectManager (
        UserID CHAR(50),
        PermissionID CHAR(50) NOT NULL,
        PRIMARY KEY UserID,
        FOREIGN KEY UserID REFERENCES User(UserID),
)

INSERT
INTO ProjectManager(UserID, PermissionID)
VALUES (SELECT UserID from User WHERE UserID ='user1', 'p1'),
(SELECT UserID from User WHERE UserID ='user2, 'p2'),
(SELECT UserID from User WHERE UserID ='user3, 'p3'),
(SELECT UserID from User WHERE UserID ='user4', 'p4'),
(SELECT UserID from User WHERE UserID ='user5', 'p5')

CREATE TABLE Member (
        UserIDCHAR(50),
        Role CHAR(50),
        PRIMARY KEY UserID,
        FOREIGN KEY UserID REFERENCES User(UserID)
)

INSERT
INTO Member(UserID, Role)
VALUES (SELECT UserID from User WHERE UserID ='user1', 'admin1'),
(SELECT UserID from User WHERE UserID ='user2, 'admin2'),
(SELECT UserID from User WHERE UserID ='user3, 'admin3'),
(SELECT UserID from User WHERE UserID ='user4', 'member1'),
(SELECT UserID from User WHERE UserID ='user5', 'member2')
```