PHASE 3: SUBMISSION CHECKLIST/SIGN-OFF SHEET

GROUP #: 1 **GROUP NAME**: Matchmakers

Deliverables:

- Revised and consistent documentation
- Implementation of classes and queries in Python (one per student) using ORM

Assessment:

- Group Status Report
- Projective Presentation

Phase Leader	Josh Honig	
Phase Recorder	Fiona Nicdao	
Phase Checker	Drew Patterson	
Technical Advisor	Hannah Serio	

1. Introduction

The Friend Dating Database (FRIDATING) allows individuals to connect with like-minded people. This platform provides a safe space for fostering new friendships, whether users are new to a city or looking to expand their social circles. In the FRIDATING Database, users can create one account, which is distinct from their user profile. While the user is the individual participating in the platform, the account they create contains their bio and shared interests, enabling them to connect with others. An account matches with other accounts based on shared interests, with communication facilitated through the platform's chat feature. Each account can engage in multiple text chat conversations with those they have matched with. Two unique traits of the FRIDATING Database are the subscription and events features, allowing special privileges to certain users and giving accounts a space to host or join events near them. Accounts can create and host many events that are open and visible for others to join. Others can indicate their interest via an RSVP mechanism, which tells the event organizer how many people to anticipate. Each user manages one subscription at a time, with a choice between base and premium subscriptions. Premium subscriptions offer additional perks, and users can choose to split their payment across multiple different payment methods. Overall, the FRIDATING Database provides an engaging and dynamic platform for individuals to build meaningful connections both online and in person. The inspiration for the design of the database comes from the group's collective experience using dating apps.

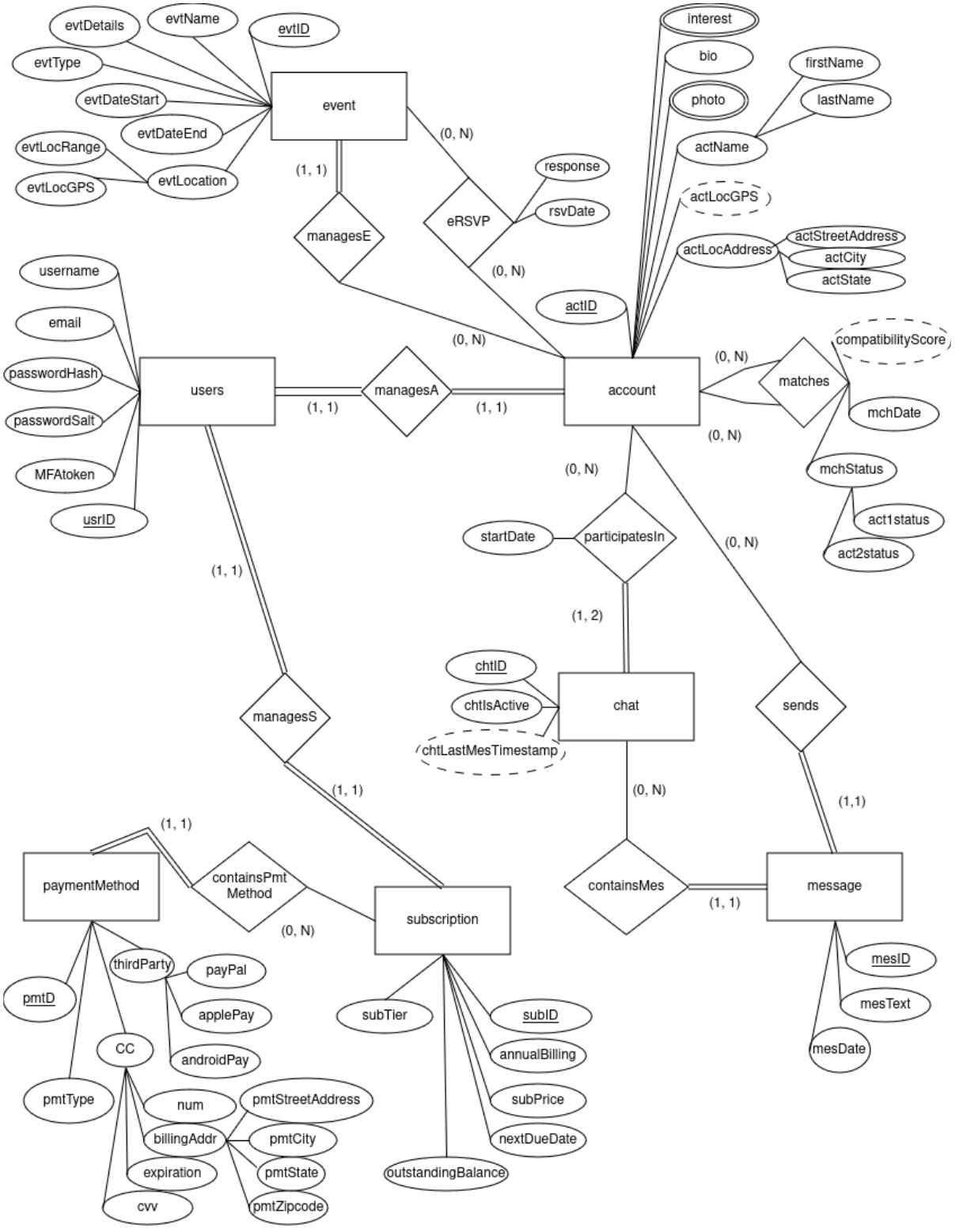
2. Description of the Organization

The Friend Dating Database (FRIDATING) stores relevant information regarding a company's need to track the actions of its users on a friend 'dating' site. The FRIDATING Database contains data about the user's matches, account information, subscription tier with possible payment methods, chats created between two users, messages sent between two users, and events shared amongst users.

- Each user possesses a unique identification number, multi-factor authentication token, username, and password. Only sensitive information is stored in the user; profile information is stored elsewhere for security purposes. Payment information is also tied to the user, since this information is considered sensitive.
- Each user must manage their account. Each account is only managed by one user. The account contains profile information such as account identification number, interests, a biography, account name, address, a user's location, and their photos. The address contains the user's street address, city, and state, from which GPS coordinates may be derived for matching users together. An account can match with zero to many other accounts. The match status is stored depending on the individual status of each user's match with each other. The match date and derived compatibility score are also stored.
- Each user can manage zero to many events containing the unique event identification number, event name, event date, event type, and event location. Event location is stored as the latitude, longitude, and location range. Each event can only be managed by one user. In addition to managing events, users can also RSVP to zero to many events using a response type.
- Each user must manage their subscription. Subscription tier type, balance, date their next payment is due, billing mode, price, and subscription identification number are all stored. Each subscription must be managed by one user.
- Each subscription can contain zero to many payment methods. A payment method must belong to one subscription. The method identification number, method type, credit card information, and third party payment information are stored.
- Accounts can participate in zero to multiple chats with their matches. The start date of the chat, chat identification number, and chat status are stored. When there has been no activity in the chat for 30 days, the chat's status will be set to “expired,” and users will not be able to send messages in the chat.
- Each chat contains zero to many messages which have a message identification number, content of the message, and date of the message. Each message must belong to only one chat. Each account can send zero to many messages. Each message must be sent by one account.

3. ER Diagram

The figure below shows the ER diagram of the Friend Dating Database (FRIDATING).



4. ER Diagram Uncaptured Constraints

The following is a list of constraints that are not captured by the ER diagram of FRIDATING:

Overall

- All payments are in USD.
- All dates are stored as timestamps stored without time zones, and are in UTC.

Subscription Entity

- The attribute **subTier**, subscription tier, has a predetermined set of values: `Free`, `Premium`, and `Bestie for Life`.
- The attribute **subPrice** is always a positive number.
- The attributes **nextDueDate** and **outstandingBalance** will be null when the subscription tier **subTier is** equal to `Free`. Otherwise, **nextDueDate** and **outstandingBalance** should not be null.
- The attribute **annualBilling** notes if the subscription is in annual billing mode or not, and is a boolean value. When true, the subscription is billed annually. When false, the subscription is billed monthly.

Account Entity

- The attribute **actLocAddress** refers to the users' address. This address is used to derive the **actLocGPS**, which is used to calculate the **compatabilityScore** between two potential matches.

Matches Relationship

- The attribute **mchStatus** for **act1status** and **act2status** refers to if the users have seen their matches and accepted or denied, or if **act1status** or **act2status** have not been presented with the match, thus their status is unseen. The predetermined set of values is: `accept`, `deny`, `unseen`.
- ~~The attribute **mchStatus** for **act1status** and **act2status** when references the account ID with the lowest ID number will be **act1status** and **act2status** will be the other account ID.~~ Removed this uncaptured constraint because UUID is used.

eRSVP Relationship

- The attribute **response** refers to if the user has RSVPed to the event so it has a predetermined set of values: `yes`, `no`, `maybe`.

- Users may change their RSVP response (**response**) as many times as they wish, but the **rsvDate** will always track the date and time of the latest response.

Event entity

- The attribute **evtType** is the type of event an account makes. There are no set values because we want to keep it open for new and different events the users can think of. Some examples of events are movie night, sports viewing, yoga, picnic, etc.
- The attribute **evtLocRange** stores the circular distance from the center point **evtLocGPS**, in feet, in which the event is held.

Chat entity

- The attribute **chtIsActive** refers to the state of the chat, depending on whether the users in it are active. It is true if the chat is active, and false if it is not.
- **chtIsActive** will change from true to false when there are no new chats within 30 days of the last message sent.
- **chtLastMesTimestamp** will reflect the timestamp of the most recent message sent in the chat. This is used for **chtIsActive** to determine if the status should be changed from active to expired.
- A **chat** can only be created between two users if both match statuses, **mchStatus**, for **act1status** and **act2status** are `accept`.

Payment Method Entity

- The attribute **pmtType** has a predetermined set of values: `credit`, `debit`, `thirdParty`.
- The attribute **thirdParty** and subattributes thereof (**androidPay**, **applePay**, **payPal**) are null if pmtType is `credit` or `debit`.
- The attribute **CC** and subattributes thereof (**num**, **cvv**, **expiration**, **pmtStreetAddr**, **pmtCity**, **pmtState**, **pmtZipcode**) are null if pmtType is `thirdParty`.
- A user's subscription will have zero payment methods if their subTier is `Free`.

5. Relational Schema

This section provides the relational schema with referential integrity and the relational table details. Foreign keys, which are not shown in §3 or §5.2 are marked in purple. Tables which are created to track one-to-many or many-to-many relationships, or multi-value attributes are marked in blue.

5.1 Relational Schema with Referential Integrity

users(usrID, username, email, passwordHash, passwordSalt, MFAtoken)

account(actID, bio, firstName, lastName, actStreetAddress, actCity, actState, actLocGPS, userID)
foreign key (userID) references users(usrID)

accountInterests(actID, interest)
foreign key (actID) references account(actID)

accountPhotos(actID, photo)
foreign key (actID) references account(actID)

chat(chatID, chtIsActive, chtLastMesTimestamp)

participatesIn(actID, chatID, startDate)
foreign key (actID) references account(actID)
foreign key (chatID) references chat (chatID)

message(mesID, mesText, mesDate, chatID, senderID)
foreign key (chatID) references chat(chatID)
foreign key (senderID) references account(actID)

event(evtID, evtType, evtDateStart, evtDateEnd, evtName, evtDetails, evtLocGPS, evtLocRange, evtManager)
foreign key (evtManager) references account(actID)

eRSVP(actID, evtID, response, rsvDate)
foreign key (actID) references account(actID)
foreign key (evtID) references event(evtID)

subscription(subID, subTier, subPrice, outstandingBalance, nextDueDate, annualBilling, userID)
foreign key (userID) references users(usrID)

paymentMethod(pmtID, subID, pmtType, num, cvv, expiration, pmtStreetAddr, pmtCity, pmtState, pmtZipcode, androidPay, applePay, payPal)
foreign key (subscription) references subscription(subID)

matches(actID1, actID2, mchDate, compatabilityScore, act1status, act2status)
foreign key (actID1) references account(actID)
foreign key (actID2) references account(actID)

5.2 Relational Table Details

The relational schema given in Section 5.1 was mapped into the following tables in the FRIDATING database. Primary keys have been underlined.

Table Name	Attribute	Description
users	usrID	Unique user ID.
	username	Username.
	email	User's email.
	passwordHash	Cryptographic hash of the password.
	passwordSalt	Cryptographic salt of the password.
	MFAtoken	Multi-factor authentication token.
account	actID	Unique account ID.
	bio	Biography for the account.
	firstName	First name of the account.
	lastName	Last name of the account.
	actStreet-Address	Street address of the account.
	actCity	City of the account
	actState	State of the account
	actLocGPS	GPS coordinates (derived from aLocAddress).
	userID	User ID associated with the account, FK
account-Interest	actID	ID of account, FK
	interest	Interest (multi-value attribute).
account-Photos	actID	ID of account, FK
	photo	Photo URL (multi-value attribute).
chat	chatID	Chat ID
	chtIsActive	Whether the chat is active.
	chtLastMes-Timestamp	Date and time of the most recent message sent in the chat (derived from the most recent message).

5.2 Relational Table Details (cont'd)

participatesIn	actID	ID of account in the chat. FK
	chatID	ID of the chat. FK
	startDate	Date and time the chat started.
message	mesID	Message ID.
	mesText	Text content of the message.
	mesDate	Date and time the message was sent.
	chatID	ID of the chat that the message belongs in. FK
	senderID	ID of the chat's sender's account. FK
event	evtID	Event ID.
	evtName	Name of the event.
	evtDetails	Details about the event.
	evtDateStart	Date and time the event starts.
	evtDateEnd	Date and time the event ends.
	evtType	Type of event.
	evtLocGPS	GPS coordinates of the event location.
	evtLocRange	Circular range, in feet, of the event location.
	evtManager	Account ID of the event manager, FK
eRSVP	actID	ID of account who RSVP, FK
	evtID	ID of the event, FK
	response	RSVP response of account (`yes,maybe,no`).
	rsvDate	Date and time of the most recent RSVP selection.

5.2 Relational Table Details (cont'd)

subscription	subID	Subscription ID.
	subTier	Subscription tier.
	subPrice	Price of the subscription, per billing cycle.
	outstanding-Balance	Outstanding balance.
	nextDueDate	Due date of next billing cycle.
	annualBilling	Whether the billing mode is annual (monthly if false).
	userID	ID of the user associated with the subscription, FK
payment Method	pmtID	Payment Method ID.
	subID	Subscription that the payment method belongs to, FK
	pmtType	Which type this payment method is (Credit card or Third Party).
	num	Credit card number.
	cvv	Credit card CVV.
	expiration	Credit card expiration.
	pmtStreet Address	Street address of billing address for credit card.
	pmtCity	City of billing address for credit card.
	pmtState	State of billing address for credit card.
	pmtZipcode	Postal code of billing address for credit card.
	androidPay	Android Pay key.
	applePay	Apple Pay key.
	payPal	PayPal key.

5.2 Relational Table Details (cont'd)

matches	actID1	ID of user 1 that matched with the other user, FK
	actID2	ID of user 2 that matched with the other user, FK
	mchDate	Date of the match.
	compatibility-Score	Numeric score of compatibility for the users (derived attribute using the data from the account entity)
	act1status	User 1's status on the match (“unseen”, “accept”, “reject”).
	act2status	User 2's status on the match.

6 Queries

Query Name	Description	Output	Relations Accessed
msgStatsFor EmailDomains (Josh)	Shows sending statistics for messages based on the email domain that is associated with each user's accounts, including the total number of messages sent, the oldest message date, and newest message date. Useful for spam moderation.	emailDomain countDomainMsgs oldestDomainMsg newestDomain-Msg	users account message
bestieForLifeActive ChatCount (Hannah)	For each user with the 'Bestie for Life' subscription tier, print the total number of chats that have an active status.	userID username activeChatsTotal	account participatesIn chat
AttendanceForFutu reEvents (Fiona)	For each event count the number of accounts who RSVP with a response of 'yes' and the state the account is located at and date of the events is in the future	numberOf-Attendance evtName ActState evtDateStart	Event eRSVP account
paidSubscriptionLo cationCount (Drew)	For each user with a paid subscription, get the count per each sub tier by location and also by payment method	actCity Subtier Subscription_count pmtType	subscriptions payment-Method account

7 DDL, DML, SQL

The following in an SQL definition of the tables for the FRIDATING database.

```
/* Josh's CREATE and INSERT statements */
CREATE TABLE users(
usrID uuid DEFAULT gen_random_uuid() NOT NULL UNIQUE,
username varchar(32) NOT NULL UNIQUE,
email varchar(96) NOT NULL UNIQUE,
passwordHash varchar(256) NOT NULL,
passwordSalt varchar(256) NOT NULL UNIQUE,
MFAtoken varchar(64) NOT NULL UNIQUE,
PRIMARY KEY(usrID)
);

INSERT INTO users(usrID, username, email, passwordHash, passwordSalt, MFAtoken) VALUES
('94ab690c-3864-407b-8354-4e606ee0cc70', 'jsmith', 'jsmith2618@gmail.com',
md5('password1'), md5(random()::text), md5(random()::text)),
('90d7a0ae-b451-400f-bc12-784817309921', 'tony1', 'tony2@yahoo.com',
md5('secret-password'), md5(random()::text), md5(random()::text)),
('cf7487c3-9196-4f46-99bd-b337baa2d655', 'ann.music', 'amusic@luc.edu', md5('mu$ic'),
md5(random()::text), md5(random()::text)),
('a799046b-9ef7-4166-9120-7f99ebac91b9', 'kay-smith', 'ks1281@gmail.com',
md5('v3rysecurepassword'), md5(random()::text), md5(random()::text)),
('40520f25-1382-4bc1-acb7-cb33af4be407', 'AzureDiamond', 'adiamond2@gmail.com',
md5('hunter2'), md5(random()::text), md5(random()::text));

CREATE TABLE account(
   actID uuid DEFAULT gen_random_uuid() NOT NULL UNIQUE,
   bio varchar(300),
   firstName varchar(32) NOT NULL,
   lastName varchar(32),
   actStreetAddress varchar(128) NOT NULL,
   actCity varchar(64) NOT NULL,
   actState varchar(64) NOT NULL,
   actLocGPS point NOT NULL,
   userID uuid NOT NULL UNIQUE,
       FOREIGN KEY(userID) REFERENCES users(usrID) ON DELETE cascade ON UPDATE
cascade,
   PRIMARY KEY(actID)
);

INSERT INTO account VALUES
(   '4a7536a1-3782-451b-a889-ecc18e6b7fb3',
   'My name is John.',
   'John', 'Smith', '6400 N Sheridan Rd, Apt 512', 'Chicago', 'IL',
   point(41.9983675,-87.6623145), '94ab690c-3864-407b-8354-4e606ee0cc70'
),
(   '3a32afed-2d4d-41c9-bef4-434ef8945dfa',
   'as featured in Tony Hawk''s Pro Skater AND Tony Hawk''s Pro strcpy',
   'Tony', NULL, '10353 E County Rd', 'Galveston', 'IN',
   point(40.6129651, -86.1789315), '90d7a0ae-b451-400f-bc12-784817309921'
),
(
   '257d8621-2e49-442b-96c2-605a651a996d',
   'hi, i''m Ann!',
   'Ann', 'M', '644 N Magnolia Av', 'Chicago', 'IL',
   point(41.9510346,-87.6630623), 'cf7487c3-9196-4f46-99bd-b337baa2d655'
),
(
   '2493c668-be9f-4485-beb7-48dbf596f983',
   NULL,
```

```
   'kay', NULL, '3688 N Atlantic Ave', 'Cocoa Beach', 'FL',
   point(28.352895,-80.6093542), 'a799046b-9ef7-4166-9120-7f99ebac91b9'
),
(
   '56131135-346e-46f2-84e3-1565178e1164',
   'fun fact: did you know if you type your password it will show as stars?',
   'AzureDiamond', NULL, '1400 Defense Boulevard', 'Arlington', 'VA',
   point(38.8697197,-77.0545081), '40520f25-1382-4bc1-acb7-cb33af4be407'
);

CREATE TABLE accountInterests(
actID uuid NOT NULL,
   FOREIGN KEY(actID) REFERENCES account(actID)
ON DELETE CASCADE ON UPDATE CASCADE,
interest varchar(32) NOT NULL UNIQUE,
PRIMARY KEY(actID, interest)
);

INSERT INTO accountInterests(actID, interest) VALUES
('4a7536a1-3782-451b-a889-ecc18e6b7fb3', 'Hiking'),
('4a7536a1-3782-451b-a889-ecc18e6b7fb3', 'Painting'),
('4a7536a1-3782-451b-a889-ecc18e6b7fb3', 'Running'),
('3a32afed-2d4d-41c9-bef4-434ef8945dfa', 'skating'),
('3a32afed-2d4d-41c9-bef4-434ef8945dfa', 'playing Tony Hawk''s Pro Skater'),
('257d8621-2e49-442b-96c2-605a651a996d', 'music'),
('257d8621-2e49-442b-96c2-605a651a996d', 'singing'),
('56131135-346e-46f2-84e3-1565178e1164', 'IRC'),
('56131135-346e-46f2-84e3-1565178e1164', 'Gaming');

CREATE TABLE accountPhotos(
actID uuid NOT NULL,
   FOREIGN KEY(actID) REFERENCES account(actID)
          ON DELETE CASCADE ON UPDATE CASCADE,
photo varchar(2048) NOT NULL UNIQUE,
PRIMARY KEY(actID, photo)
);

INSERT INTO accountPhotos(actID, photo) VALUES
('4a7536a1-3782-451b-a889-ecc18e6b7fb3',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   a2195bf2-2211-4d8a-bbd7-09a655740ce6.jpg'),
('4a7536a1-3782-451b-a889-ecc18e6b7fb3',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   751e035d-3526-44c5-8427-a12fc34e6a45.jpg'),
('4a7536a1-3782-451b-a889-ecc18e6b7fb3',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   e4e19be7-112d-420f-8d75-93947f133a9f.jpg'),
('4a7536a1-3782-451b-a889-ecc18e6b7fb3',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   81ce9010-b69a-4002-bda9-d6459f6de980.jpg'),
('4a7536a1-3782-451b-a889-ecc18e6b7fb3',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   f9e80b5e-cdc4-42d6-a541-392c6076340f.jpg'),
('3a32afed-2d4d-41c9-bef4-434ef8945dfa',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   eb42d1d6-b4cd-4283-82eb-4e05feef46f3.jpg'),
('3a32afed-2d4d-41c9-bef4-434ef8945dfa',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   53bd7274-b8eb-442c-9c9b-ba83c418a03a.jpg'),
('257d8621-2e49-442b-96c2-605a651a996d',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   ec0bb0a1-b1bb-4381-8bf8-121b90d318a1.jpg'),
('257d8621-2e49-442b-96c2-605a651a996d',
```

```
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   fd03f3ca-eb70-4632-a3b3-92428fc21623.jpg'),
('56131135-346e-46f2-84e3-1565178e1164',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   dc896d53-5e8e-4ea5-9d8e-3b31a8df88b5.jpg'),
('56131135-346e-46f2-84e3-1565178e1164',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   a9e61e2b-9573-4034-a84b-b14e1468d0aa.jpg'),
('56131135-346e-46f2-84e3-1565178e1164',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   63427ccd-2bd7-4d93-92e3-e14b0a4e9a46.jpg');

CREATE TABLE chat(
chatID uuid DEFAULT gen_random_uuid() NOT NULL UNIQUE,
chtIsActive boolean NOT NULL,
chtLastMesTimestamp timestamp WITHOUT time zone, /* UTC */
PRIMARY KEY(chatID)
);

INSERT INTO chat VALUES
('0b3d9d58-85d0-4679-a5a1-86f7b5fd019f', true, current_timestamp),
('9389cf0b-7fee-4368-b276-c5c1b28f3ffc', true, (current_timestamp - interval '3'
day)),
('bad455be-b7a5-49ce-9299-93a514a2b507', false, (current_timestamp - interval '5'
month - interval '15' day)),
('aaefa26b-b842-4d31-950b-9f7a9b8e0ea7', true, NULL)
/* New chat with no messages */;


CREATE TABLE participatesIn(
   actID uuid NOT NULL,
           FOREIGN KEY(actID) REFERENCES account(actID)
ON DELETE CASCADE ON UPDATE CASCADE,
   chatID uuid NOT NULL,
           FOREIGN KEY(chatID) REFERENCES chat(chatID)
ON DELETE CASCADE ON UPDATE CASCADE,
   startDate timestamp WITHOUT time zone,
   PRIMARY KEY(actID, chatID)
);

INSERT INTO participatesIn(actID, chatID, startDate) VALUES
(
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
   '0b3d9d58-85d0-4679-a5a1-86f7b5fd019f',
   (current_timestamp - interval '1' day)
),
(
   '3a32afed-2d4d-41c9-bef4-434ef8945dfa', /* Tony */
   '0b3d9d58-85d0-4679-a5a1-86f7b5fd019f',
   (current_timestamp - interval '1' day)
),
(
   '257d8621-2e49-442b-96c2-605a651a996d', /* Ann */
   '9389cf0b-7fee-4368-b276-c5c1b28f3ffc',
   (current_timestamp - interval '6' day)
),
(
   '56131135-346e-46f2-84e3-1565178e1164', /* AzureDiamond */
   '9389cf0b-7fee-4368-b276-c5c1b28f3ffc',
   (current_timestamp - interval '6' day)
),
(
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
```

```
   'bad455be-b7a5-49ce-9299-93a514a2b507',
   (current_timestamp - interval '5' month - interval '30' day)
),
(
   '56131135-346e-46f2-84e3-1565178e1164', /* AzureDiamond */
   'bad455be-b7a5-49ce-9299-93a514a2b507',
   (current_timestamp - interval '5' month - interval '30' day)
),
(
   '2493c668-be9f-4485-beb7-48dbf596f983', /* kay */
   'aaefa26b-b842-4d31-950b-9f7a9b8e0ea7',
   (current_timestamp - interval '5' minute)
),
(
   '257d8621-2e49-442b-96c2-605a651a996d', /* Ann */
   'aaefa26b-b842-4d31-950b-9f7a9b8e0ea7',
   (current_timestamp - interval '5' minute)
);


CREATE TABLE message(
mesID serial NOT NULL,
mesText varchar(2048) NOT NULL,
mesDate timestamp WITHOUT time zone NOT NULL,
chatID uuid NOT NULL,
   FOREIGN KEY(chatID) REFERENCES chat(chatID)
           ON DELETE CASCADE ON UPDATE CASCADE,
senderID uuid NOT NULL,
   FOREIGN KEY(senderID) REFERENCES account(actID)
           ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(mesID)
);


INSERT INTO message(mesText, mesDate, chatID, senderID) VALUES

(
   'Hi Tony. How are you?',
   current_timestamp - interval '1' day - interval '15' minute,
   '0b3d9d58-85d0-4679-a5a1-86f7b5fd019f',
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3' /* John */
),
(
   'I''m good John, how are you?..',
   current_timestamp - interval '1' day,
   '0b3d9d58-85d0-4679-a5a1-86f7b5fd019f',
   '3a32afed-2d4d-41c9-bef4-434ef8945dfa' /* Tony */
),

(
   'hi Ann! what''s up?',
   current_timestamp - interval '8' day,
   '9389cf0b-7fee-4368-b276-c5c1b28f3ffc',
   '56131135-346e-46f2-84e3-1565178e1164' /* AzureDiamond */
),
(
   'not much. tell me about the fact in your bio! i don''t really get it',
   current_timestamp - interval '7' day - interval '12' hour,
   '9389cf0b-7fee-4368-b276-c5c1b28f3ffc',
   '257d8621-2e49-442b-96c2-605a651a996d' /* Ann */
),
(
   'oh if you type your password it automatically turns to stars! like this: hunter2',
```

```
   current_timestamp - interval '6' day,
   '9389cf0b-7fee-4368-b276-c5c1b28f3ffc',
   '56131135-346e-46f2-84e3-1565178e1164' /* AzureDiamond */
),

(
   'Hey there, neat username. What does it stand for?',
   current_timestamp - interval '5' month - interval '15' day,
   'bad455be-b7a5-49ce-9299-93a514a2b507',
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3' /* John */
);

CREATE TABLE event(
evtID uuid DEFAULT gen_random_uuid() NOT NULL UNIQUE,
evtType varchar(32),
evtDateStart timestamp WITHOUT time zone NOT NULL,
evtDateEnd timestamp WITHOUT time zone NOT NULL,
evtName varchar(64) NOT NULL,
evtDetails varchar(2048) NOT NULL,
evtLocGPS point NOT NULL,
evtLocRange int NOT NULL, /* in feet */
evtManager uuid NOT NULL,
   FOREIGN KEY(evtManager) REFERENCES account(actID)
           ON DELETE CASCADE ON UPDATE CASCADE,
PRIMARY KEY(evtID)
);

INSERT INTO event VALUES
(
   '17170cd3-1bfa-4809-9816-28503c9986fe',
   'Game',
   current_timestamp + interval '5' day,
   current_timestamp + interval '5' day + interval '4' hour,
   'Board Games @ Cuneen''s',
   'Our THIRD board game night for anybody in the Edgewater area!
   We''ll have Azul, Scrabble, Jenga, and Catan. Tonight we''ll be at
   Cuneen''s on Devon.',
   point(41.9983065,-87.6666288), 50,
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3'
),
(
   '85067bd4-11fb-4fcb-b1e7-b3883088f90c',
   'Group Activity',
   current_timestamp + interval '7' day,
   current_timestamp + interval '8' day,
   'Alligator Wrestling',
   'this week''s gator wrestling. everybody wear dark dark clothes,
   we do NOT want the cops called on us again.
   DON''T BE THERE UNTIL AFTER SUNDOWN',
   point(28.288404259380975,-80.61084415798224), 65,
   '2493c668-be9f-4485-beb7-48dbf596f983'
);

CREATE TYPE rsvp_status AS ENUM ('yes', 'maybe', 'no');
CREATE TABLE eRSVP(
   actID uuid NOT NULL,
           FOREIGN KEY(actID) REFERENCES account(actID)
           ON DELETE CASCADE ON UPDATE CASCADE,
   evtID uuid NOT NULL,
           FOREIGN KEY(evtID) REFERENCES event(evtID)
           ON DELETE CASCADE ON UPDATE CASCADE,
   response rsvp_status NOT NULL,
   rsvDate timestamp WITHOUT time zone NOT NULL,
```

```
   PRIMARY KEY(actID, evtID)
);

INSERT INTO eRSVP VALUES
(
   '257d8621-2e49-442b-96c2-605a651a996d', /* Ann */
   '17170cd3-1bfa-4809-9816-28503c9986fe', /* Board Games @ Cuneen's */
   'yes',
   current_timestamp - interval '1' day
),
(
   '3a32afed-2d4d-41c9-bef4-434ef8945dfa', /* Tony */
   '17170cd3-1bfa-4809-9816-28503c9986fe', /* Board Games @ Cuneen's */
   'maybe',
   current_timestamp - interval '20' hour
),
(
   '56131135-346e-46f2-84e3-1565178e1164', /* AzureDiamond */
   '85067bd4-11fb-4fcb-b1e7-b3883088f90c', /* Alligator Wrestling */
   'maybe',
   current_timestamp - interval '2' hour
),
(
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
   '85067bd4-11fb-4fcb-b1e7-b3883088f90c', /* Alligator Wrestling */
   'no',
   current_timestamp - interval '5' minute
);

CREATE TABLE subscription(
   subID uuid DEFAULT gen_random_uuid() NOT NULL UNIQUE,
   subTier varchar(64) NOT NULL,
   subPrice numeric,
   outstandingBalance numeric,
   nextDueDate timestamp WITHOUT time zone,
   annualBilling boolean,
   userID uuid NOT NULL,
           FOREIGN KEY(userID) REFERENCES users(usrID),
   PRIMARY KEY(subID)
);

INSERT INTO subscription VALUES
(
   '4f36fd46-c84c-4d7c-868b-85532e580dab',
   'Bestie for Life', 49.99, 0.00,
   '2024-12-10 16:39:00', true,
   '94ab690c-3864-407b-8354-4e606ee0cc70'
);
INSERT INTO subscription(subTier, userID) VALUES
('Free', '90d7a0ae-b451-400f-bc12-784817309921'),
('Free', 'cf7487c3-9196-4f46-99bd-b337baa2d655'),
('Free', 'a799046b-9ef7-4166-9120-7f99ebac91b9'),
('Free', '40520f25-1382-4bc1-acb7-cb33af4be407');

CREATE TYPE payment_type AS ENUM ('credit', 'debit', 'thirdParty');
CREATE TABLE paymentMethod(
pmtID uuid DEFAULT gen_random_uuid() NOT NULL UNIQUE,
subID uuid NOT NULL,
   FOREIGN KEY(subID) REFERENCES subscription(subID)
           ON UPDATE CASCADE ON DELETE CASCADE,
pmtType payment_type NOT NULL,
num numeric,
cvv numeric,
```

```
expiration varchar(16),
pmtStreetAddr varchar(128),
pmtCity varchar(64),
pmtState varchar(64),
pmtZipcode varchar(64),
androidPay varchar(128),
applePay varchar(128),
payPal varchar(128),
PRIMARY KEY(pmtID)
);

INSERT INTO paymentMethod(pmtID, subID, pmtType, applePay) VALUES
(
   '49ee6acd-08a8-481c-9462-9e10969a97f2', '4f36fd46-c84c-4d7c-868b-85532e580dab',
   'thirdParty', '5G8HzqBxpa73VTqSjfipb0u0NKRgwbTe'
);

CREATE TYPE match_status AS ENUM ('unseen', 'accept', 'reject');
CREATE TABLE matches(
actID1 uuid NOT NULL,
   FOREIGN KEY(actID1) REFERENCES account(actID)
            ON DELETE CASCADE ON UPDATE CASCADE,
actID2 uuid NOT NULL,
   FOREIGN KEY(actID2) REFERENCES account(actID)
            ON DELETE CASCADE ON UPDATE CASCADE,
mchDate timestamp WITHOUT time zone NOT NULL,
compatabilityScore numeric NOT NULL,
act1status match_status NOT NULL DEFAULT 'unseen',
act2status match_status NOT NULL DEFAULT 'unseen',
PRIMARY KEY(actID1, actID2)
);

INSERT INTO matches VALUES
(
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
   '3a32afed-2d4d-41c9-bef4-434ef8945dfa', /* Tony */
   current_timestamp - interval '3' day,
   9.553262, 'accept', 'accept'
),
(
   '56131135-346e-46f2-84e3-1565178e1164', /* AzureDiamond */
   '257d8621-2e49-442b-96c2-605a651a996d', /* Ann */
   current_timestamp - interval '10' day,
   4.613523, 'accept', 'accept'
),
(
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
   '56131135-346e-46f2-84e3-1565178e1164', /* AzureDiamond */
   current_timestamp - interval '6' month,
   2.61918804, 'accept', 'accept'
),
(
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
   '2493c668-be9f-4485-beb7-48dbf596f983', /* Kay */
   current_timestamp - interval '1' month,
   3.161310, 'reject', 'accept'
),
(
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
   '257d8621-2e49-442b-96c2-605a651a996d', /* Ann */
   current_timestamp - interval '3' day,
   8.1356971, 'accept', 'unseen'
),
```

```
(
   '3a32afed-2d4d-41c9-bef4-434ef8945dfa', /* Tony */
   '56131135-346e-46f2-84e3-1565178e1164', /* AzureDiamond */
   current_timestamp - interval '1' minute,
   3.1851014, 'unseen', 'unseen'
),
(
   '3a32afed-2d4d-41c9-bef4-434ef8945dfa', /* Tony */
   '2493c668-be9f-4485-beb7-48dbf596f983', /* Kay */
   current_timestamp - interval '1' minute,
   2.1358178, 'unseen', 'unseen'
);

/* Josh's spammer/scammer profile INSERT statements */
INSERT INTO users(usrID, username, email, passwordHash, passwordSalt, MFAToken) VALUES
('a87b4c9b-f164-4e4b-99a4-2d64f2212ca7', 'FreeRobux', 'robux1578@sketchkysite.ru',
md5('jEh812hWH&(Q@qjio'), md5(random()::text), md5(random()::text));
INSERT INTO account VALUES
(
   'b079763a-21fd-409e-89a8-ccd1d570848f',
   'Want to learn how to earn FRE_E <R0BUX>? Go to
http://freemon3y.robux1513125316.pw',
   'Free-Robux_website', NULL,
   '123454 Red Square', 'Tverskoy District', 'Moscow',
   point(55.7536141,37.6227219),
   'a87b4c9b-f164-4e4b-99a4-2d64f2212ca7'
);
INSERT INTO accountPhotos VALUES
('b079763a-21fd-409e-89a8-ccd1d570848f',
'https://s3.amazonaws.com/cdn.fridating.org/account-img/af20a503-e4de-425b-a1af-ed2a3d
989f4a');
INSERT INTO matches VALUES
(
   'b079763a-21fd-409e-89a8-ccd1d570848f',
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3',
   current_timestamp - interval '15' day,
   1.062179, 'accept', 'accept'
),
(
   'b079763a-21fd-409e-89a8-ccd1d570848f',
   '3a32afed-2d4d-41c9-bef4-434ef8945dfa',
   current_timestamp - interval '15' day,
   1.13589, 'accept', 'unseen'
),
(
   'b079763a-21fd-409e-89a8-ccd1d570848f',
   '257d8621-2e49-442b-96c2-605a651a996d',
   current_timestamp - interval '15' day,
   1.36161, 'accept', 'unseen'
),
(
   'b079763a-21fd-409e-89a8-ccd1d570848f',
   '2493c668-be9f-4485-beb7-48dbf596f983',
   current_timestamp - interval '15' day,
   1.91748, 'accept', 'accept'
),
(
   'b079763a-21fd-409e-89a8-ccd1d570848f',
   '56131135-346e-46f2-84e3-1565178e1164',
   current_timestamp - interval '15' day,
   1.222179, 'accept', 'unseen'
);
INSERT INTO chat VALUES
```

```
('c3eca16c-0076-4806-b531-c84dc0727894', true,
current_timestamp - interval '3' day),
('5aeeadef-5122-4f6c-969e-ba5b391da800', true,
current_timestamp - interval '3' day),
('eee3a5e8-e2f5-4400-b4b6-e9c0cab3c0d5', true,
current_timestamp - interval '3' day),
('6482e27a-1a10-493c-86e7-8c95c6649ef3', true,
current_timestamp - interval '3' day);
INSERT INTO participatesIn VALUES
(
   'b079763a-21fd-409e-89a8-ccd1d570848f', /* spammer */
   'c3eca16c-0076-4806-b531-c84dc0727894',
   current_timestamp - interval '3' day
),
(
   '2493c668-be9f-4485-beb7-48dbf596f983',
   'c3eca16c-0076-4806-b531-c84dc0727894',
   current_timestamp - interval '3' day
),

(
   'b079763a-21fd-409e-89a8-ccd1d570848f', /* spammer */
   '5aeeadef-5122-4f6c-969e-ba5b391da800',
   current_timestamp - interval '3' day
),
(
   '257d8621-2e49-442b-96c2-605a651a996d',
   '5aeeadef-5122-4f6c-969e-ba5b391da800',
   current_timestamp - interval '3' day
),

(
   'b079763a-21fd-409e-89a8-ccd1d570848f', /* spammer */
   'eee3a5e8-e2f5-4400-b4b6-e9c0cab3c0d5',
   current_timestamp - interval '3' day
),
(
   '4a7536a1-3782-451b-a889-ecc18e6b7fb3',
   'eee3a5e8-e2f5-4400-b4b6-e9c0cab3c0d5',
   current_timestamp - interval '3' day
),

(
   'b079763a-21fd-409e-89a8-ccd1d570848f', /* spammer */
   '6482e27a-1a10-493c-86e7-8c95c6649ef3',
   current_timestamp - interval '3' day
),
(
   '56131135-346e-46f2-84e3-1565178e1164',
   '6482e27a-1a10-493c-86e7-8c95c6649ef3',
   current_timestamp - interval '3' day
);
INSERT INTO message(mesText, mesDate, chatID, senderID) VALUES
(
   'FREE R0BUCKS AT THIS LINK ---> http://freemoney.robuxscam2351.pw',
   current_timestamp - interval '3' day,
   'c3eca16c-0076-4806-b531-c84dc0727894',
   'b079763a-21fd-409e-89a8-ccd1d570848f'
),
(
   'NEED ROBUX NOW? FREE HERE --> http://robux.legit2353212website.ru',
   current_timestamp - interval '3' day,
   '5aeeadef-5122-4f6c-969e-ba5b391da800',
```

```
   'b079763a-21fd-409e-89a8-ccd1d570848f'
),
(
   '------> http://robux-generator-now-free.roblocks.ua <------',
   current_timestamp - interval '3' day,
   'eee3a5e8-e2f5-4400-b4b6-e9c0cab3c0d5',
   'b079763a-21fd-409e-89a8-ccd1d570848f'
),
(
   '!!! Tap HERE >>> http://robloock-generator-free-noscam.org <<< HERE NOW for FREE
robux !!!',
   current_timestamp - interval '3' day,
   '6482e27a-1a10-493c-86e7-8c95c6649ef3',
   'b079763a-21fd-409e-89a8-ccd1d570848f'
);

/* Hannah's insertions */
INSERT INTO users(usrID, username, email, passwordHash, passwordSalt, MFAtoken) VALUES
('3e5fb29d-ce1b-4f3b-9ced-c35858d389cf', 'katy_pr3ston', 'katyp9570@gmail.com',
md5('katy'), md5(random()::text), md5(random()::text)),
('b7832602-640f-45c9-856a-51496cb97b97', 'madmax', 'mmd0g@yahoo.com',
md5('madmaxd0g'), md5(random()::text), md5(random()::text)),
('bebaaef9-e9a0-40a2-9a65-ab905b164ad7', 'chicagolex', 'lex33420@hotmail.com',
md5('lex1'), md5(random()::text), md5(random()::text)),
('a70baa2f-84c6-4839-9b07-543510a752c0', 'berrybenson', 'b_benson12@gmail.com',
md5('bens0n12'), md5(random()::text), md5(random()::text)),
('7f702396-b720-43d7-a2d9-f96a1b4d3569', 'd0gl0ver213', 'amanda_spears30@gmail.com',
md5('amandasm1th'), md5(random()::text), md5(random()::text));

INSERT INTO account VALUES
('072cb76c-8ca0-43f2-a515-8418d1931193', 'It''s Katy P! <3', 'Katy', 'Perry', '1939
Loomis St', 'Rockford', 'IL', point(42.26220,-89.12451),
'3e5fb29d-ce1b-4f3b-9ced-c35858d389cf'),
('c00605f5-dc5e-41df-83e3-36e4f6427284', 'Looking to play basketball with friends',
'Max', 'Madewell', '301 S 4th Ave', 'Brighton', 'CO', point(39.98200, -104.81848),
'b7832602-640f-45c9-856a-51496cb97b97'),
('f1b21a51-c213-4e83-bc50-b821f317bea4', 'single & ready to.. Make friends!', 'Berry',
'Benson', '512 S 3rd St', 'Champaign', 'IL', point(40.11231, -88.23516),
'bebaaef9-e9a0-40a2-9a65-ab905b164ad7'),
('d7165614-5e9c-4d7c-9003-102dd83545ce', 'If you love reading, walks by the lake, and
movies let''s hang!', 'Amanda', 'Smith', '2634 W Argyle St', 'Chicago', 'IL',
point(41.97254, -87.69531), '7f702396-b720-43d7-a2d9-f96a1b4d3569');

INSERT INTO accountPhotos(actID, photo) VALUES
('072cb76c-8ca0-43f2-a515-8418d1931193',
'https://s3.amazonaws.com/cdn.fridating.org/account-img/
file1.jpg'),
('d7165614-5e9c-4d7c-9003-102dd83545ce',
'https://s3.amazonaws.com/cdn.fridating.org/account-img/
file2.jpg');

INSERT INTO accountInterests(actID, interest) VALUES
('f1b21a51-c213-4e83-bc50-b821f317bea4', 'Fishing'),
('f1b21a51-c213-4e83-bc50-b821f317bea4', 'Photography'),
('d7165614-5e9c-4d7c-9003-102dd83545ce', 'Reading');

INSERT INTO subscription(subID, subTier, userID, outstandingBalance, nextDueDate)
VALUES
('1cb5b930-62ed-47f0-8a42-52c8f9ed0a2b', 'Bestie for Life',
'3e5fb29d-ce1b-4f3b-9ced-c35858d389cf', 0.00, '2024-12-15 16:39:00'),
('56669135-539c-4993-911a-736f9b237916', 'Bestie for Life',
'bebaaef9-e9a0-40a2-9a65-ab905b164ad7', 0.00, '2024-12-30 16:39:00'),
```

```
('e8368d90-94ff-47c4-9f9b-428b1b456d85', 'Premium',
'7f702396-b720-43d7-a2d9-f96a1b4d3569', 0.00, '2024-12-17 16:39:00'),
('d44d4ac4-97eb-4b6a-b888-23571bc3ab41', 'Free',
'b7832602-640f-45c9-856a-51496cb97b97', NULL, NULL);

INSERT INTO paymentMethod(pmtID, subID, pmtType, num, cvv, expiration, pmtStreetAddr,
pmtCity, pmtState, pmtZipcode) VALUES
('5fde54a6-af9a-496e-955b-a7aaf9cce663', '1cb5b930-62ed-47f0-8a42-52c8f9ed0a2b',
'debit', '4532970356782245', '123', '12/25', '123 Elm Street', 'Springfield', 'IL',
'62701'),
('710a95b5-dac9-41f3-ad38-bbf5dde1c38f', '56669135-539c-4993-911a-736f9b237916',
'debit', '6011592898763443', '456', '06/26', '456 Maple Ave', 'Anytown', 'CA',
'90210'),
('a8e3fed5-9411-4b2c-81ec-034b024a77cc', 'e8368d90-94ff-47c4-9f9b-428b1b456d85',
'debit', '371449635398431', '789', '09/27', '512 S 3rd St', 'Champaign', 'IL',
'61820');


INSERT INTO chat VALUES
('0e8c29f9-4176-48dd-b231-19419005451a', true, current_timestamp),
('a26cdfb4-112b-4654-9ef5-80b5bfbbeff2', true, (current_timestamp - interval '1'
day)),
('84ec9373-79f9-4477-b8fe-7f37f9afd83a', false, (current_timestamp - interval '7'
month)),
('2ac5ea0f-bc55-4d0a-8112-63926079cb40', true, (current_timestamp - interval '1'
day)),
('5b93b3a3-f1fe-459a-9451-de9677852d10', true, (current_timestamp - interval '1'
day));

INSERT INTO participatesIn(actID, chatID, startDate) VALUES
('072cb76c-8ca0-43f2-a515-8418d1931193', /* Katy */
'0e8c29f9-4176-48dd-b231-19419005451a', (current_timestamp - interval '10' day)),
('f1b21a51-c213-4e83-bc50-b821f317bea4', /* Berry */
'0e8c29f9-4176-48dd-b231-19419005451a', (current_timestamp - interval '10' day)),
('072cb76c-8ca0-43f2-a515-8418d1931193', /* Katy */
'a26cdfb4-112b-4654-9ef5-80b5bfbbeff2', (current_timestamp - interval '3' day)),
('c00605f5-dc5e-41df-83e3-36e4f6427284', /* Max */
'a26cdfb4-112b-4654-9ef5-80b5bfbbeff2', (current_timestamp - interval '3' day)),
('f1b21a51-c213-4e83-bc50-b821f317bea4', /* Berry */
'84ec9373-79f9-4477-b8fe-7f37f9afd83a', (current_timestamp - interval '5' month -
interval '30' day)),
('c00605f5-dc5e-41df-83e3-36e4f6427284', /* Max */
'84ec9373-79f9-4477-b8fe-7f37f9afd83a', (current_timestamp - interval '5' month -
interval '30' day)),
('072cb76c-8ca0-43f2-a515-8418d1931193', /* Katy */
'2ac5ea0f-bc55-4d0a-8112-63926079cb40', (current_timestamp - interval '3' minute)),
('d7165614-5e9c-4d7c-9003-102dd83545ce', /* Amanda */
'2ac5ea0f-bc55-4d0a-8112-63926079cb40', (current_timestamp - interval '3' minute)),
('c00605f5-dc5e-41df-83e3-36e4f6427284', /* Max */
'5b93b3a3-f1fe-459a-9451-de9677852d10', (current_timestamp - interval '7' month -
interval '30' day)),
('d7165614-5e9c-4d7c-9003-102dd83545ce', /* Amanda */
'5b93b3a3-f1fe-459a-9451-de9677852d10', (current_timestamp - interval '7' month -
interval '30' day));

INSERT INTO matches VALUES
('072cb76c-8ca0-43f2-a515-8418d1931193', /* Katy */
'f1b21a51-c213-4e83-bc50-b821f317bea4', /* Berry */
current_timestamp - interval '10' day,
8.433218, 'accept', 'accept'),
('072cb76c-8ca0-43f2-a515-8418d1931193', /* Katy */
'c00605f5-dc5e-41df-83e3-36e4f6427284', /* Max */
current_timestamp - interval '3' day,
```

```
4.389701, 'accept', 'accept'),
('c00605f5-dc5e-41df-83e3-36e4f6427284', /* Max */
'f1b21a51-c213-4e83-bc50-b821f317bea4', /* Berry */
current_timestamp - interval '5' month - interval '30' day,
7.845243, 'accept', 'accept'),
('072cb76c-8ca0-43f2-a515-8418d1931193', /* Katy */
'd7165614-5e9c-4d7c-9003-102dd83545ce', /* Amanda */
current_timestamp - interval '3' minute,
9.826543, 'accept', 'accept'),
('c00605f5-dc5e-41df-83e3-36e4f6427284', /* Max */
'd7165614-5e9c-4d7c-9003-102dd83545ce', /* Amanda */
current_timestamp - interval '7' month - interval '30' day,
3.213907, 'accept', 'accept'),
('4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
'd7165614-5e9c-4d7c-9003-102dd83545ce', /* Amanda */
current_timestamp - interval '8' day,
2.608476, 'accept', 'reject'),
('4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
'c00605f5-dc5e-41df-83e3-36e4f6427284', /* Max */
current_timestamp - interval '2' month,
3.901824, 'reject', 'unseen');


INSERT INTO event VALUES
('5b13ed06-e18e-4cd3-a29a-5315d1a78147',
'Group Activity',
current_timestamp + interval '14' day,
current_timestamp + interval '14' day + interval '2' hour,
'Twilight Book Club!',
'Come enjoy wine and baked goods as we discuss the Twilight trilogy',
point(41.93776,-87.66006), 100,
'4a7536a1-3782-451b-a889-ecc18e6b7fb3'),
('e91b9519-ab8a-4d00-8f4a-b6086e9c9b7e',
'Game',
current_timestamp + interval '7' day,
current_timestamp + interval '7' day + interval '6' hour,
'Chess Tournament',
'Players of all levels are welcome! $100 grand prize for the 1st place winner.',
point(28.288404259380975,-80.61084415798224), 40,
'f1b21a51-c213-4e83-bc50-b821f317bea4');

INSERT INTO eRSVP VALUES
('c00605f5-dc5e-41df-83e3-36e4f6427284', /* Max */
'e91b9519-ab8a-4d00-8f4a-b6086e9c9b7e', /* Chess Tournament */
'maybe',
current_timestamp - interval '1' day),
('d7165614-5e9c-4d7c-9003-102dd83545ce', /* Amanda */
'e91b9519-ab8a-4d00-8f4a-b6086e9c9b7e', /* Chess Tournament */
'yes',
current_timestamp - interval '20' hour),
('56131135-346e-46f2-84e3-1565178e1164', /* AzureDiamond */
'5b13ed06-e18e-4cd3-a29a-5315d1a78147', /* Book Club */
'yes',
current_timestamp - interval '7' hour),
('4a7536a1-3782-451b-a889-ecc18e6b7fb3', /* John */
'5b13ed06-e18e-4cd3-a29a-5315d1a78147', /* Book Club */
'no',
current_timestamp - interval '5' minute);

INSERT INTO message(mesText, mesDate, chatID, senderID) VALUES
('Hi Berry, are you going to the chess tournament?',
current_timestamp - interval '15' minute,
'0e8c29f9-4176-48dd-b231-19419005451a',
```

```
'072cb76c-8ca0-43f2-a515-8418d1931193' /* Katy */),
('I''m thinking about it. I''ll let you know later.',
current_timestamp - interval '5' minute,
'0e8c29f9-4176-48dd-b231-19419005451a',
'f1b21a51-c213-4e83-bc50-b821f317bea4' /* Berry */);


/* Drew's additions to tables */

INSERT INTO users(usrID, username, email, passwordHash, passwordSalt, MFAtoken)
VALUES
('6ecd2aaa-4d44-4b9e-990d-409773324c5b', 'toddy846', 'todd.hanks@gmail.com',
md5('ThePassword123'), md5(random()::text), md5(random()::text)),
('3aced7f0-c2e7-435a-916d-a0a8904389bb', 'LianaT24','liana.timmons24@gmail.com',
md5('randopass'), md5(random()::text), md5(random()::text)),
('829ebcd5-d57f-434d-931b-aec167cf5c03', 'JayMorrow87', 'jason.morrow87@hotmail.com',
md5('helloworld'), md5(random()::text), md5(random()::text)) ,
('61c389ba-63fb-48b9-861b-fcbd8f77d710', 'Rivas12', 'marisol.rivas12@yahoo.com',
md5('awesomeepic'), md5(random()::text), md5(random()::text)),
('21bb1aa8-6dbf-4e14-ace7-01c4e8df2296', 'TyGuy5', 'tyler.caldwell45@aol.com',
md5('veryCoolsw@g'), md5(random()::text), md5(random()::text));


INSERT INTO account VALUES
(   '40d2f395-88b0-48d7-b291-a71e28f6b16f',
   'Yo its Todd!!!',
   'Todd', 'Hanks', '4320 Vesta Drive', 'Chicago', 'IL',
   point(41.929520,-87.548576), '6ecd2aaa-4d44-4b9e-990d-409773324c5b'
),
(   '3542a953-f5b5-4e4c-9de2-06f67985caf4',
   'Excited to get to meet some new people!',
   'Liana', 'Timmons', '2824 Sand Fork Road', 'Galveston', 'IN',
   point(40.588402, -86.200150), '3aced7f0-c2e7-435a-916d-a0a8904389bb'
),
(
   'a72c26d1-d294-4a69-812c-7db5d319299e',
   'hi, i''m Jay!',
   'Jay', 'M', '4136 Oakmound Drive', 'Chicago', 'IL',
   point(41.896301,-87.642776), '829ebcd5-d57f-434d-931b-aec167cf5c03'
),
(
   '65523ca9-4f42-41fc-a8e1-84863ce8fa43',
   NULL,
   'Marisol', NULL, '342 Rosemont Avenue', 'Cocoa Beach', 'FL',
   point(28.282207,-80.633247), '61c389ba-63fb-48b9-861b-fcbd8f77d710'
),
(
   '33d4d3ac-576c-4d11-85b1-bbb9ebaa38e6',
   'fun fact about me is that I''ve broken every bone in my body',
   'Tyler', 'Caldwell', '3454 Perine Street', 'Arlington', 'VA',
   point(38.838612,-77.044670), '21bb1aa8-6dbf-4e14-ace7-01c4e8df2296'
);

INSERT INTO accountInterests(actID, interest) VALUES
('40d2f395-88b0-48d7-b291-a71e28f6b16f', 'Mortal Kombat'),
('3542a953-f5b5-4e4c-9de2-06f67985caf4', 'Snooker'),
('a72c26d1-d294-4a69-812c-7db5d319299e', 'Surfboarding'),
('65523ca9-4f42-41fc-a8e1-84863ce8fa43', 'Roller Skating'),
('33d4d3ac-576c-4d11-85b1-bbb9ebaa38e6', 'Tennis');

INSERT INTO accountPhotos(actID, photo) VALUES
('40d2f395-88b0-48d7-b291-a71e28f6b16f',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
```

```
   827fec18-e3a5-472f-b8d8-f2c42336e91f.jpg'),
('3542a953-f5b5-4e4c-9de2-06f67985caf4',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   28bba5ab-6982-442b-a627-fe23d7c9109c.jpg'),
('a72c26d1-d294-4a69-812c-7db5d319299e',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   394aba0d-0b28-4ffb-a198-8a6cb9e2023f.jpg'),
('65523ca9-4f42-41fc-a8e1-84863ce8fa43',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   4951712d-7521-4e4e-b11a-236c08b6d738.jpg'),
('33d4d3ac-576c-4d11-85b1-bbb9ebaa38e6',
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   1a4bfedc-9b81-450b-a28c-0e2a88e02f4b.jpg');

INSERT INTO chat VALUES
('7753c633-e6e4-4ce0-ac72-5e10a0973c58', true, (current_timestamp - interval '1'
day)),
('3f9cbcd0-be29-421b-bddd-22f9ff9adac3', true, (current_timestamp - interval '7'
day)),
('b4f26983-f368-451e-9f73-9860d858e32d', false, (current_timestamp - interval '6'
month - interval '20' day)),
('3f00a86e-9be1-4904-abb8-036566cc5931', true, NULL);
/* New chat with no messages */

INSERT INTO participatesIn(actID, chatID, startDate) VALUES
(
   '40d2f395-88b0-48d7-b291-a71e28f6b16f', -- Todd
   '7753c633-e6e4-4ce0-ac72-5e10a0973c58', -- w/ Jay
   (current_timestamp - interval '1' day)
),
(
   '3542a953-f5b5-4e4c-9de2-06f67985caf4', -- Liana to Marisol
   '3f9cbcd0-be29-421b-bddd-22f9ff9adac3',
   (current_timestamp - interval '7' day)
),
(
   'a72c26d1-d294-4a69-812c-7db5d319299e', -- Jay
   '7753c633-e6e4-4ce0-ac72-5e10a0973c58', --  w/ Todd
   (current_timestamp - interval '1' day)
),
(
   '65523ca9-4f42-41fc-a8e1-84863ce8fa43', -- Marisol to Liana
   '3f9cbcd0-be29-421b-bddd-22f9ff9adac3',
   (current_timestamp - interval '7' day)
),
(
   '33d4d3ac-576c-4d11-85b1-bbb9ebaa38e6', -- Tyler
   '3f00a86e-9be1-4904-abb8-036566cc5931',
   (NULL)
);

INSERT INTO message(mesText, mesDate, chatID, senderID) VALUES

(
   'Hey Jay! Whats up?',
   current_timestamp - interval '1' day - interval '15' minute,
   '7753c633-e6e4-4ce0-ac72-5e10a0973c58',
   '40d2f395-88b0-48d7-b291-a71e28f6b16f' /* Todd to Jay */
),
(
   'Yo Todd! I''m good how are you bro?..',
   current_timestamp - interval '1' day,
   '7753c633-e6e4-4ce0-ac72-5e10a0973c58',
```

```
   'a72c26d1-d294-4a69-812c-7db5d319299e' /* Jay to Todd*/
),

(
   'Just living it up in Chicago! What are your hobbies?',
   current_timestamp - interval '1' day,
   '7753c633-e6e4-4ce0-ac72-5e10a0973c58',
   '40d2f395-88b0-48d7-b291-a71e28f6b16f' /* Todd to Jay*/
),
(
   'So I see your from Florida! Hows the weather?',
   current_timestamp - interval '7' day - interval '12' hour,
   '3f9cbcd0-be29-421b-bddd-22f9ff9adac3',
   '3542a953-f5b5-4e4c-9de2-06f67985caf4' /* Liana to Marisol */
),
(
   'Super humid so far this season!',
   current_timestamp - interval '6' day,
   '3f9cbcd0-be29-421b-bddd-22f9ff9adac3',
   '65523ca9-4f42-41fc-a8e1-84863ce8fa43' /* Marisol to Liana*/
);

INSERT INTO event VALUES
(
   '41024d3c-bca4-4020-b700-42bcd481f018',
   'Game',
   current_timestamp + interval '5' day,
   current_timestamp + interval '5' day + interval '4' hour,
   'Mortal Kombat Tournament',
   'Join us for an epic Mortal Kombat tournament happening right here at Emporium
Arcade Bar!
   Prepare to unleash your best combos, fatalities, and fighting skills against the
fiercest competitors in town.',
   point(41.906658,-87.671860), 50,
   '40d2f395-88b0-48d7-b291-a71e28f6b16f' /* Todd */
);

INSERT INTO eRSVP VALUES
(
   'a72c26d1-d294-4a69-812c-7db5d319299e', /* Jay */
   '41024d3c-bca4-4020-b700-42bcd481f018', /* MK tourney */
   'yes',
   current_timestamp - interval '1' day
),
(
   '65523ca9-4f42-41fc-a8e1-84863ce8fa43', /* Marisol */
   '85067bd4-11fb-4fcb-b1e7-b3883088f90c', /* Alligator wrestliknq */
   'maybe',
   current_timestamp - interval '20' hour
),
(
   '40d2f395-88b0-48d7-b291-a71e28f6b16f', /* Todd */
   '17170cd3-1bfa-4809-9816-28503c9986fe', /* Cuneen's game night */
   'yes',
   current_timestamp - interval '2' hour
);

INSERT INTO subscription VALUES
(
   '4d39e47b-b878-49f9-b958-493aebbb4a18',
   'Bestie for Life', 49.99, 0.00,
   '2024-11-10 16:39:00', true,
   '6ecd2aaa-4d44-4b9e-990d-409773324c5b' /* Todd */
```

```
),
(
   'b307e71b-bbe4-4055-b488-d01ab711a5bd',
   'Premium', 9.99, 0.00,
   '2025-2-20 16:39:00', true,
   '829ebcd5-d57f-434d-931b-aec167cf5c03' /* Jay */
);

INSERT INTO subscription(subID, subTier, userID) VALUES
('8b9e57d7-2415-4ce6-809e-cdea98b877ac', 'Free',
'61c389ba-63fb-48b9-861b-fcbd8f77d710'), /* Marisol */
('df3d6cc7-fde9-4b10-857f-c7e7d6f87fe3', 'Free',
'3aced7f0-c2e7-435a-916d-a0a8904389bb'), /* Liana */
('711833fc-d69e-42a0-a494-fca38ec97fad', 'Free',
'21bb1aa8-6dbf-4e14-ace7-01c4e8df2296'); /* Tyler */

INSERT INTO paymentMethod(pmtID, subID, pmtType, num, cvv, expiration, pmtStreetAddr,
pmtCity, pmtState, pmtZipcode) VALUES
('5da4e5a7-462d-4ec8-b241-a543ee978cd6', '4d39e47b-b878-49f9-b958-493aebbb4a18',
'debit', 4798315489743516, 456, '11/04', '4320 Vesta Drive', 'Chicago', 'IL',
'60656'), /*Todd */
('1a4b7f9f-9b7d-4699-96cd-8c9e2d6e2d75', 'b307e71b-bbe4-4055-b488-d01ab711a5bd',
'debit', 9874654386546348, 795, '10/10', '4136 Oakmound Drive', 'Chicago', 'IL',
'60651'); /* Jay */


INSERT INTO matches VALUES
(
   '40d2f395-88b0-48d7-b291-a71e28f6b16f', /* Todd */
   'a72c26d1-d294-4a69-812c-7db5d319299e', /* Jay */
   current_timestamp - interval '2' day,
   9.5853262, 'accept', 'accept'
),
(
   '40d2f395-88b0-48d7-b291-a71e28f6b16f', /* Todd  */
   '33d4d3ac-576c-4d11-85b1-bbb9ebaa38e6', /* Tyler */
   current_timestamp - interval '10' day,
   6.617823, 'accept', 'unseen'
),
(
   'a72c26d1-d294-4a69-812c-7db5d319299e', /* Jay */
   '3542a953-f5b5-4e4c-9de2-06f67985caf4', /* Liana */
   current_timestamp - interval '6' month,
   7.61125604, 'accept', 'accept'
),
(
   '40d2f395-88b0-48d7-b291-a71e28f6b16f', /* Todd */
   '257d8621-2e49-442b-96c2-605a651a996d', /* Ann */
   current_timestamp - interval '1' month,
   3.161310, 'reject', 'reject'
),
(
   '65523ca9-4f42-41fc-a8e1-84863ce8fa43', /* Marisol */
   '3542a953-f5b5-4e4c-9de2-06f67985caf4', /* Liana */
   current_timestamp - interval '3' day,
   3.1986971, 'unseen', 'unseen'
),
(
   '33d4d3ac-576c-4d11-85b1-bbb9ebaa38e6', /* Tyler */
   '65523ca9-4f42-41fc-a8e1-84863ce8fa43', /* Marsiol */
   current_timestamp - interval '1' minute,
   3.124879, 'unseen', 'reject'
);
```

```
/* Fiona's additions to tables */
INSERT INTO users(usrID, username, email, passwordHash, passwordSalt, MFAtoken) VALUES
('1c9ab10a-c70f-4209-a64d-c8019a23aac6', 'wmaximoff', 'wmaximoff3@gmail.com',
md5('magic5'), md5(random()::text),md5(random()::text)),
('8f465290-e54b-452b-b981-cbf92a6cf2d1', 'vision', 'vision@yahoo.com',
md5('ihateultron2'), md5(random()::text), md5(random()::text)),
('3db2bb9a-2418-4258-bbc4-c7ea32b39570', 'emmafrost', 'emma1frost@aol.com',
md5('snowflake12'), md5(random()::text), md5(random()::text)),
('3ecd3feb-6076-4ec3-b3ce-87daa077a4b3', 'eriklensherr', 'magneto@gmail.com',
md5('notthevillain3'), md5(random()::text), md5(random()::text)),
('328ee807-7686-4e08-83cc-0b3d9c111e9b', 'steverogers', 'capamerica@aol.com',
md5('1stAvenger'), md5(random()::text), md5(random()::text));

INSERT INTO account VALUES
(
   '8f11b297-0e06-4d8e-ac29-11b3818c6375',
   ' Hi I am Wanda Maximoff and I love magic the gathering',
   'Wanda', 'Maximoff', '1032 W. Sheridan Rd', 'Chicago', 'IL',
point(41.998604, -87.657699), '1c9ab10a-c70f-4209-a64d-c8019a23aac6'
),

(
   '0090798d-60f8-483c-9516-b4ebe4bd55d3',
   ' Hi I am Vision and interested in  advanced technology ',
   'Vision', NULL, '6430 N. Kenmore Ave', 'Chicago', 'IL',
   point(41.999172, -87.657213), '8f465290-e54b-452b-b981-cbf92a6cf2d1'
),
(
   'fe6a275d-03d5-465f-8748-69ad732e3f7b',
   ' Hi I am Emma Frost and winter is my favorite season ',
   'Emma', 'Frost', '202 Sycamore St', 'Galveston', 'IN',
   point(40.577760, -86.187590), '3db2bb9a-2418-4258-bbc4-c7ea32b39570'
),
(
   'e431e4b0-9a77-4ac6-b4ab-0931fedc8a75',
   ' Hi I am Erik Lensherr and I like discussing the marvels of mutation',
   'Erik', 'Lensherr', '429 S California St', 'Galveston', 'IN',
   point(40.573639, -86.186236), '3ecd3feb-6076-4ec3-b3ce-87daa077a4b3'
),
(
   '611eb7a5-1bab-4eea-8107-949d17e7f02e',
   ' Hi I am Steve Rogers and looking make strong friendships',
   'Steve', 'Rogers', '2080 N Atlantic Ave', 'Cocoa Beach', 'FL',
   point(28.343305, -80.607015), '328ee807-7686-4e08-83cc-0b3d9c111e9b'
);

INSERT INTO accountInterests(actID, interest) VALUES
('8f11b297-0e06-4d8e-ac29-11b3818c6375','Games'), /*wanda*/
('0090798d-60f8-483c-9516-b4ebe4bd55d3','Card Games'), /*vision*/
('fe6a275d-03d5-465f-8748-69ad732e3f7b','Going Hiking'), /*emma*/
('fe6a275d-03d5-465f-8748-69ad732e3f7b','listen to music'), /*emma*/
('e431e4b0-9a77-4ac6-b4ab-0931fedc8a75','listening to music'), /*erik*/
('611eb7a5-1bab-4eea-8107-949d17e7f02e','Running Marathons'); /*steve */

INSERT INTO accountPhotos(actID, photo) VALUES
(   '8f11b297-0e06-4d8e-ac29-11b3818c6375', /*wanda*/
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   583322f1-717b-4729-8d7a-e145e2c013c1.jpg'
),
(   '0090798d-60f8-483c-9516-b4ebe4bd55d3', /*vision*/
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   97e5aee6-bb15-472c-a291-78a8c6215334.jpg'
```

```
),
(   'fe6a275d-03d5-465f-8748-69ad732e3f7b', /*emma*/
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   56fe7bd4-363f-4b11-bcaf-d31d86a0a74c.jpg'
),
(   'e431e4b0-9a77-4ac6-b4ab-0931fedc8a75', /*erik*/
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   9369fa02-ea4b-4bc7-a149-d05aee621395.jpg'
),
(   '611eb7a5-1bab-4eea-8107-949d17e7f02e', /*steve*/
   'https://s3.amazonaws.com/cdn.fridating.org/account-img/
   28e084d2-04c2-47a6-9d6e-dd7ed191d0bf.jpg'
);

INSERT INTO chat VALUES
('14fd47ff-2fa2-4d08-8283-600e077af006', true, current_timestamp),
('6c578b7d-73fa-40d8-a171-dc25dddb331d', true, current_timestamp - interval '4' day),
('3528e995-bda2-4eeb-b012-6274f3d3f6b3', true, current_timestamp - interval '2' day);

INSERT INTO participatesIn(actID, chatID, startDate) VALUES
(
'8f11b297-0e06-4d8e-ac29-11b3818c6375',/*wanda*/
'14fd47ff-2fa2-4d08-8283-600e077af006',
(current_timestamp - interval '1' day)
),
(
'0090798d-60f8-483c-9516-b4ebe4bd55d3',/*vision*/
'14fd47ff-2fa2-4d08-8283-600e077af006',
(current_timestamp - interval '1' day)
),
(
'e431e4b0-9a77-4ac6-b4ab-0931fedc8a75',/*erik*/
'6c578b7d-73fa-40d8-a171-dc25dddb331d',
(current_timestamp - interval '4' day)
),
(
'fe6a275d-03d5-465f-8748-69ad732e3f7b',/*emma  */
'6c578b7d-73fa-40d8-a171-dc25dddb331d',
(current_timestamp - interval '4' day)
),
(
'611eb7a5-1bab-4eea-8107-949d17e7f02e', /*steve*/
'3528e995-bda2-4eeb-b012-6274f3d3f6b3',
(current_timestamp - interval '2' day)
),
(
'65523ca9-4f42-41fc-a8e1-84863ce8fa43', /*marisol*/
'3528e995-bda2-4eeb-b012-6274f3d3f6b3',
(current_timestamp - interval '2' day)
);


INSERT INTO message(mesText, mesDate, chatID, senderID) VALUES
(
   'Hi Vision! Do you play Magic the Gathering ?',
   current_timestamp - interval '1' day - interval '5' minute,
   '14fd47ff-2fa2-4d08-8283-600e077af006',
   '8f11b297-0e06-4d8e-ac29-11b3818c6375' /*Wanda */
),
(
   'What is up Wanda! Ya! I just got the new duskmourn deck.',
   current_timestamp - interval '1' day - interval '10' minute,
   '14fd47ff-2fa2-4d08-8283-600e077af006',
```

```
'0090798d-60f8-483c-9516-b4ebe4bd55d3' /*Vision*/
),
(
   'Hi Emma, what is your favorite music genre ? ',
   current_timestamp - interval '4' day - interval '15' minute,
   '6c578b7d-73fa-40d8-a171-dc25dddb331d',
'e431e4b0-9a77-4ac6-b4ab-0931fedc8a75' /*erik*/
),
(
   'Hi Erik, I love alternative pop music? ',
   current_timestamp - interval '4' day - interval '30' minute,
   '6c578b7d-73fa-40d8-a171-dc25dddb331d',
'fe6a275d-03d5-465f-8748-69ad732e3f7b' /*emma*/
),
(
   'Hi Marisol, do you like running ?',
   current_timestamp - interval '2' day - interval '5' minute,
'3528e995-bda2-4eeb-b012-6274f3d3f6b3',
'611eb7a5-1bab-4eea-8107-949d17e7f02e' /*steve*/
),
(
   'Yes! I just finished the Chicago Marathon and hoping to run in New York
next year.',
   current_timestamp - interval '2' day - interval '15' minute,
'3528e995-bda2-4eeb-b012-6274f3d3f6b3',
'65523ca9-4f42-41fc-a8e1-84863ce8fa43' /*marisol*/
);

INSERT INTO eRSVP VALUES
(
   '8f11b297-0e06-4d8e-ac29-11b3818c6375', /*Wanda */
   '17170cd3-1bfa-4809-9816-28503c9986fe', /* Board Games @ Cuneen's */
   'yes',
   current_timestamp - interval '1' day
),
(
   '0090798d-60f8-483c-9516-b4ebe4bd55d3', /*Vision*/
   '17170cd3-1bfa-4809-9816-28503c9986fe', /* Board Games @ Cuneen's */
   'yes',
   current_timestamp - interval '1' day
),
(
   '611eb7a5-1bab-4eea-8107-949d17e7f02e', /*steve*/
   '85067bd4-11fb-4fcb-b1e7-b3883088f90c', /* alligator wrestling */
   'yes',
   current_timestamp - interval '4' day
);

INSERT INTO subscription VALUES
(
   'aa8225c4-48f4-4012-bc78-579c0897d710',
   'Bestie for Life', 49.99, 0.00,
   '2024-11-10 16:00:00', true,
   '328ee807-7686-4e08-83cc-0b3d9c111e9b' /*steve*/
);


INSERT INTO subscription(subTier, userID) VALUES
('Free', '1c9ab10a-c70f-4209-a64d-c8019a23aac6'),
('Free', '8f465290-e54b-452b-b981-cbf92a6cf2d1'),
('Free', '3db2bb9a-2418-4258-bbc4-c7ea32b39570');

INSERT INTO paymentMethod(pmtID, subID, pmtType, PayPal) VALUES
```

```
(
   '4468551d-ffe8-4eca-adcd-1f7392ca82ec', 'aa8225c4-48f4-4012-bc78-579c0897d710',
   'thirdParty', '6L5HzqBxpa73VTqSjfipb8u0NKRgwbTe'
);


INSERT INTO matches VALUES
(
   '8f11b297-0e06-4d8e-ac29-11b3818c6375',/*wanda*/
   '0090798d-60f8-483c-9516-b4ebe4bd55d3',/*vision*/
   current_timestamp - interval '1' day,
   9.300956, 'accept', 'accept'
),
(
   '8f11b297-0e06-4d8e-ac29-11b3818c6375',/*wanda*/
   '611eb7a5-1bab-4eea-8107-949d17e7f02e', /*steve*/
   current_timestamp - interval '2' day,
   4.776183, 'accept', 'reject'
),
(
   'e431e4b0-9a77-4ac6-b4ab-0931fedc8a75',/*erik*/
   'fe6a275d-03d5-465f-8748-69ad732e3f7b',/*emma  */
   current_timestamp - interval '4' day,
   9.708713, 'accept', 'accept'
),
(
   'e431e4b0-9a77-4ac6-b4ab-0931fedc8a75',/*erik*/
   '611eb7a5-1bab-4eea-8107-949d17e7f02e', /*steve*/
   current_timestamp - interval '4' day,
   3.645354, 'unseen', 'unseen'
),
(
   '611eb7a5-1bab-4eea-8107-949d17e7f02e', /*steve*/
   '65523ca9-4f42-41fc-a8e1-84863ce8fa43', /*marisol*/
   current_timestamp - interval '2' day,
   8.502177, 'accept', 'accept'
);
```

```
/* Begin queries */

/* Hannah's query */
/* bestieForLifeActiveChatCount: For each user with the 'Bestie for Life'
subscription tier, print the total number of chats that have an active
status. */
select A.userID, U.username, count(DISTINCT P.chatID) as activeChatCount
from account A, chat C, participatesIn P, subscription S, users U
where A.actID = P.actID and
C.chatID = P.chatID and
A.userID = U.usrID and
S.subTier = 'Bestie for Life'
group by A.userID, U.username
order by activeChatCount desc;
```

	userid uuid	username character varying (32)	activechatcount bigint
1	a87b4c9b-f164-4e4b-99a4-2d64f2212ca7	FreeRobux	4
2	40520f25-1382-4bc1-acb7-cb33af4be407	AzureDiamond	3
3	cf7487c3-9196-4f46-99bd-b337baa2d655	ann.music	3
4	b7832602-640f-45c9-856a-51496cb97b97	madmax	3
5	94ab690c-3864-407b-8354-4e606ee0cc70	jsmith	3
6	3e5fb29d-ce1b-4f3b-9ced-c35858d389cf	katy_pr3ston	3
7	7f702396-b720-43d7-a2d9-f96a1b4d3569	d0gl0ver213	2
8	61c389ba-63fb-48b9-861b-fcbd8f77d710	Rivas12	2
9	a799046b-9ef7-4166-9120-7f99ebac91b9	kay-smith	2
10	bebaaef9-e9a0-40a2-9a65-ab905b164ad7	chicagolex	2
11	1c9ab10a-c70f-4209-a64d-c8019a23aac6	wmaximoff	1
12	829ebcd5-d57f-434d-931b-aec167cf5c03	JayMorrow87	1
13	8f465290-e54b-452b-b981-cbf92a6cf2d1	vision	1
14	90d7a0ae-b451-400f-bc12-784817309921	tony1	1
15	3db2bb9a-2418-4258-bbc4-c7ea32b39570	emmafrost	1
16	21bb1aa8-6dbf-4e14-ace7-01c4e8df2296	TyGuy5	1
17	3aced7f0-c2e7-435a-916d-a0a8904389bb	LianaT24	1
18	3ecd3feb-6076-4ec3-b3ce-87daa077a4b3	eriklensherr	1
19	328ee807-7686-4e08-83cc-0b3d9c111e9b	steverogers	1
20	6ecd2aaa-4d44-4b9e-990d-409773324c5b	toddy846	1

```
/* Josh's query */
/* msgStatsForEmailDomains: Shows sending statistics for messages based on
the email domain that is associated with each user's accounts, including
the total number of messages sent, the oldest message date, and newest
message date. Useful for spam moderation. */
SELECT
   split_part(users.email, '@', 2) AS emailDomain,
   count(*) AS countDomainMsgs,
   min(msg.mesDate) AS oldestDomainMsg,
   max(msg.mesDate) AS newestDomainMsg
FROM users, account, message msg
WHERE
   account.userID=users.usrID and
   msg.senderID=account.actID
GROUP BY(emailDomain)
ORDER BY(countDomainMsgs) desc;
```

	emaildomain text	countdomainmsgs bigint	oldestdomainmsg timestamp without time zone	newestdomainmsg timestamp without time zone
1	gmail.com	10	2024-04-22 16:02:50.927921	2024-10-28 21:05:25.344074
2	sketchkysite.ru	4	2024-10-19 21:36:49.263736	2024-10-19 21:36:49.263736
3	yahoo.com	4	2024-10-21 16:02:50.927921	2024-10-28 04:56:36.739782
4	hotmail.com	2	2024-10-28 01:30:25.106773	2024-10-28 21:15:25.344074
5	aol.com	2	2024-10-25 04:36:36.739782	2024-10-27 05:01:36.739782
6	luc.edu	1	2024-10-15 04:02:50.927921	2024-10-15 04:02:50.927921

```
/* Drew's query */
/* paidSubscriptionLocationCount: For each user with a paid subscription,
get the count per each sub tier by location and also by payment method. */

select A.actCity, S.subtier, count(S.subID) as subscription_count,
P.pmtType
from paymentMethod P, account A, subscription S
where S.userID = A.userID and
     S.subID = P.subID
group by S.subTier, A.actCity, P.pmtType
order by A.actCity asc, subscription_count desc;
```

	actcity character varying (64)	subtier character varying (64)	subscription_count bigint	pmttype payment_type
1	Champaign	Bestie for Life	1	debit
2	Chicago	Premium	2	debit
3	Chicago	Bestie for Life	1	debit
4	Chicago	Bestie for Life	1	thirdParty
5	Cocoa Beach	Bestie for Life	1	thirdParty
6	Rockford	Bestie for Life	1	debit

```
/* Fiona's query */
/* AttendanceForFutureEvents: For each event count the number of accounts
who RSVP with a response of 'yes'  and the state the account is located at
and date of the events  is in the future. */

SELECT E.evtID, E.evtName, E.evtDateStart, A.actState, count(R.response)
FROM Event E, eRSVP R, Account A
WHERE R.response = 'yes' and E.evtID = R.evtID and A.actID = R.actID
GROUP BY E.evtID, A.actState
ORDER BY evtDateStart asc;
```

	evtid uuid	evtname character varying (64)	evtdatestart timestamp without time zone	actstate character varying (64)	count bigint
1	17170cd3-1bfa-4809-9816-28503c9986fe	Board Games @ Cuneen's	2024-10-27 16:20:17.566907	IL	4
2	85067bd4-11fb-4fcb-b1e7-b3883088f90c	Aligator Wrestling	2024-10-29 16:20:17.566907	FL	1
3	41024d3c-bca4-4020-b700-42bcd481f018	Mortal Kombat Tournament	2024-11-03 01:30:25.106773	IL	1
4	e91b9519-ab8a-4d00-8f4a-b6086e9c9b7e	Chess Tournament	2024-11-04 21:14:05.736744	IL	1
5	5b13ed06-e18e-4cd3-a29a-5315d1a78147	Twilight Book Club!	2024-11-11 21:14:05.736744	VA	1

8 ORM

```
from typing import List
from typing import Optional
from sqlalchemy import ForeignKey
from sqlalchemy import String, Integer, Numeric, DateTime, Enum, Float, func
from sqlalchemy.orm import DeclarativeBase
from sqlalchemy.orm import Mapped
from sqlalchemy.orm import mapped_column
from sqlalchemy.orm import relationship
from sqlalchemy import create_engine
from sqlalchemy.orm import Session
from sqlalchemy import select
from sqlalchemy.dialects.postgresql import UUID
import uuid
import datetime
import enum
import pdb

# IMPORTANT! IMPORTANT! IMPORTANT!
# Change this to TRUE if you want to attempt to write data to the db!
actuallyWriteDataToTheDatabase = False
# IMPORTANT! IMPORTANT! IMPORTANT!

# Database connection
engine =
create_engine("postgresql+psycopg2://jhonig:pfLqVAkQEppPWdo7EaGzUvMJNdHPjX2t@lo
calhost:8080/FRIDATING-dev2")
session = Session(engine)

class Base(DeclarativeBase):
    pass

# Class created by Drew; Defining User class/table
class Users(Base):
    __tablename__ = "users"

    usrid: Mapped[uuid.UUID] = mapped_column(UUID(as_uuid=True),
default=uuid.uuid4, primary_key=True)
    username: Mapped[str] = mapped_column(String(32))
    email: Mapped[str] = mapped_column(String(96))
    passwordhash: Mapped[str] = mapped_column(String(256))
    passwordsalt: Mapped[str] = mapped_column(String(256))
    mfatoken: Mapped[str] = mapped_column(String(64))
    subscription: Mapped[List["Subscription"]] = relationship("Subscription",
back_populates="user")

    def __repr__(self) -> str:
        return f"User(id={self.userID!r}, username={self.username!r},
email={self.email!r})"
```

```
# Class created by Josh; define Subscription table
class Subscription(Base):
    __tablename__ = "subscription"

    subid: Mapped[uuid.UUID] = mapped_column(UUID(as_uuid=True),
default=uuid.uuid4, primary_key=True)
    subtier: Mapped[str] = mapped_column(String(64))
    subprice: Mapped[Numeric] = mapped_column(Numeric, nullable=True)
    outstandingbalance: Mapped[Numeric] = mapped_column(Numeric, nullable=True)
    nextduedate: Mapped[datetime] = mapped_column(DateTime(timezone=False),
nullable=True) # https://stackoverflow.com/a/75364468/11467212
    annualbilling: Mapped[Optional[bool]] # Since bool is a native python type,
sqlalchemy will figure this out # https://stackoverflow.com/a/76499049/11467212
    userid: Mapped[uuid.UUID] = mapped_column(ForeignKey("users.usrid"))

    paymentmethods: Mapped[List["paymentmethod"]] = relationship(
        back_populates="subscription", cascade="all, delete-orphan"
    )

    user: Mapped["Users"] = relationship(back_populates="subscription")

    def __repr__(self) -> str:
        return f"""Subscription(
            id={self.subid!r}, subtier={self.subtier!r},
subprice={self.subprice!r},
            nextduedate={self.nextduedate!r},
annualbilling={self.annualbilling!r},
            userid={self.userid!r})"""

# Class created by Josh; defines enum for use in paymnetmethod
class payment_type(enum.Enum):
    credit = 'credit'
    debit = 'debit'
    thirdParty = 'thirdParty'
# Class created by Josh; define paymentmethod table
class paymentmethod(Base):
    __tablename__ = "paymentmethod"

    pmtid: Mapped[uuid.UUID] = mapped_column(UUID(as_uuid=True),
default=uuid.uuid4, primary_key=True)
    subid: Mapped[uuid.UUID] = mapped_column(ForeignKey("subscription.subid"))
    pmttype = mapped_column(Enum(payment_type))
    num: Mapped[Numeric] = mapped_column(Numeric, nullable=True)
    cvv: Mapped[Numeric] = mapped_column(Numeric, nullable=True)
    expiration: Mapped[str] = mapped_column(String(16), nullable=True)
    pmtstreetaddr: Mapped[str] = mapped_column(String(128), nullable=True)
    pmtcity: Mapped[str] = mapped_column(String(64), nullable=True)
    pmtstate: Mapped[str] = mapped_column(String(64), nullable=True)
    pmtzipcode: Mapped[str] = mapped_column(String(64), nullable=True)
    androidpay: Mapped[str] = mapped_column(String(128), nullable=True)
    applepay: Mapped[str] = mapped_column(String(128), nullable=True)
    paypal: Mapped[str] = mapped_column(String(128), nullable=True)

    subscription: Mapped["Subscription"] =
relationship(back_populates="paymentmethods")
```

```
    def __repr__(self) -> str:
        return f"""paymentMethod(
            pmtid={self.pmtid!r}, subid={self.subid!r},
pmtType={self.pmttype!r},
            num={self.num!r}, cvv={self.cvv!r}, expiration={self.expiration!r},
            pmtstreetaddr={self.pmtstreetaddr!r}, pmtcity={self.pmtcity!r},
            pmtstate={self.pmtstate!r}, pmtzipcode={self.pmtzipcode!r},
            androidpay={self.androidpay!r}, applepay={self.applepay!r},
paypal={self.paypal!r})"""


# Classes created by Hannah
class Account(Base):
    __tablename__ = "account"

    actid: Mapped[uuid.UUID] = mapped_column(UUID(as_uuid=True),
default=uuid.uuid4, primary_key=True)
    bio: Mapped[str] = mapped_column(String(300))
    firstname: Mapped[str] = mapped_column(String(32), nullable=False)
    lastname: Mapped[str] = mapped_column(String(32))
    actstreetaddress: Mapped[str] = mapped_column(String(128), nullable=False)
    actcity: Mapped[str] = mapped_column(String(64), nullable=False)
    actstate: Mapped[str] = mapped_column(String(64), nullable=False)
    #actlocgps_x: Mapped[float] = mapped_column(Float, nullable=False)  #
Longitude or x-coordinate
    #actlocgps_y: Mapped[float] = mapped_column(Float, nullable=False)  #
Latitude or y-coordinate
    userid: Mapped[str] = mapped_column(Integer, ForeignKey("users.usrid"),
nullable=False)
    participates_in: Mapped["ParticipatesIn"] = relationship("ParticipatesIn",
back_populates="account")
    def __repr__(self) -> str:
        return f"Account(id={self.actID!r}, firstName={self.firstname!r},
lastName={self.lastname!r})"

# Class created by Hannah
class ParticipatesIn(Base):
    __tablename__ = "participatesin"

    actid: Mapped[uuid.UUID] = mapped_column(UUID(as_uuid=True),
ForeignKey("account.actid", ondelete="CASCADE", onupdate="CASCADE"),
primary_key=True)
    chatid: Mapped[uuid.UUID] = mapped_column(UUID(as_uuid=True),
ForeignKey("chat.chatid", ondelete="CASCADE", onupdate="CASCADE"),
primary_key=True)
    startdate: Mapped[datetime] = mapped_column(DateTime(timezone=False),
nullable=True)

    # Relationships
    account: Mapped["Account"] = relationship("Account",
back_populates="participates_in")
    chat: Mapped["Chat"] = relationship("Chat", back_populates="participants")
```

```
   def __repr__(self) -> str: return f"ParticipatesIn(actID={self.actid!r},
chatID={self.chatid!r})"

# Classes created by Fiona & Hannah
class Chat(Base):
  __tablename__ = "chat"
   chatid: Mapped[uuid.UUID] = mapped_column(UUID(as_uuid=True),
default=uuid.uuid4, primary_key=True)
  chtisactive: Mapped[bool]
  chtlastmestimestamp: Mapped[datetime] =
mapped_column(DateTime(timezone=False))
  # Relationships
  participants: Mapped["ParticipatesIn"] = relationship("ParticipatesIn",
back_populates="chat")
  # reference to messages
  messages : Mapped[List["Message"]] = relationship(
      back_populates="chat", cascade="all, delete-orphan"
  )
  def __repr__(self) -> str: #represent the object as a string
      return f"Chat (id = {self.id!r}, chatIsActive{self.chtisactive!r},
chtLastMesTimestamp{self.chtlastmestimestamp!r})"

# Class created by Fiona
class Message(Base):
  __tablename__ = "message"
  mesid: Mapped[int] = mapped_column(primary_key=True, autoincrement=True)
  mestext: Mapped[str] = mapped_column(String(200))
  mesdate : Mapped[datetime] = mapped_column(DateTime(timezone=True))
  chatid : Mapped[uuid.UUID] = mapped_column(UUID(as_uuid=True),
ForeignKey("chat.chatid"))
  senderid: Mapped[uuid.UUID] = mapped_column(UUID(as_uuid=True),
ForeignKey("account.actid"))
  chat : Mapped["Chat"] = relationship(back_populates="messages")
   def __repr__(self) -> str :
      return f"Message(id={self.id!r}), mesText={self.mestext!r},
mesDate={self.mesdate!r}"
```

```
# Insert data - Josh
with Session(engine) as session:
    # Insert subscriptions
    subscription1 = Subscription(
        subid = uuid.UUID("ba2a5760-0ead-4988-b9cc-b2cc77cfcd34"),
        subtier = "Premium",
        subprice = 9.99,
        outstandingbalance = 0.00,
        nextduedate = datetime.datetime.now() + datetime.timedelta(days=6),
        annualbilling = False,
        userid = uuid.UUID("3db2bb9a-2418-4258-bbc4-c7ea32b39570")
    )
    subscription2 = Subscription(
        subid = uuid.UUID("ef7b5717-96f2-42b4-8bb2-23ae9172ff65"),
        subtier = "Bestie for Life",
        subprice = 49.99,
        outstandingbalance = 0.00,
        nextduedate = datetime.datetime.now() + datetime.timedelta(days=3) +
datetime.timedelta(hours=5),
        annualbilling = False,
        userid = uuid.UUID("a87b4c9b-f164-4e4b-99a4-2d64f2212ca7")
    )
    if actuallyWriteDataToTheDatabase:
        session.add_all([subscription1, subscription2])
        session.commit()

    # Insert payment methods
    pmtmethod1 = paymentmethod(
        subid = uuid.UUID("ba2a5760-0ead-4988-b9cc-b2cc77cfcd34"),
        pmttype = payment_type.thirdParty,
        paypal = "somePaypalPaymentAPIKeyGoesHere"
    )
    pmtmethod2 = paymentmethod(
        subid = uuid.UUID("ef7b5717-96f2-42b4-8bb2-23ae9172ff65"),
        pmttype = payment_type.credit,
        num = 4381946282957154,
        cvv = 123,
        expiration = "05/25",
        pmtstreetaddr = "1 N Oak St",
        pmtcity = "Anytown",
        pmtstate = "IL",
        pmtzipcode = "60601"
    )
    pmtmethod3 = paymentmethod(
        subid = uuid.UUID("ef7b5717-96f2-42b4-8bb2-23ae9172ff65"),
        pmttype = payment_type.credit,
        num = 6195016491745164,
        cvv = 321,
        expiration = "01/26",
        pmtstreetaddr = "405 S Railroad Ave",
        pmtcity = "Cornfieldia",
        pmtstate = "IN",
        pmtzipcode = "43052"
    )
```

```
    pmtmethod4 = paymentmethod(
        subid = uuid.UUID("ef7b5717-96f2-42b4-8bb2-23ae9172ff65"),
        pmttype = payment_type.credit,
        num = 1750161067181961,
        cvv = 159,
        expiration = "12/24",
        pmtstreetaddr = "8000 University Rd",
        pmtcity = "Lafayette",
        pmtstate = "IN",
        pmtzipcode = "47904"
    )
    if actuallyWriteDataToTheDatabase:
        session.add_all([pmtmethod1, pmtmethod2, pmtmethod3, pmtmethod4])
        session.commit()
```

```
# Insert data - Drew
with Session(engine) as session:
   user_1 = Users(
       usrid=uuid.UUID("4fc51a78-8d6c-4c64-9c68-a005558a0444"),
       username="TrueB00L",
       email="B00leanFreak45@aol.com",
       passwordhash="kYf8npoLQB@",
       passwordsalt="o@a#&%JE@E1",
       mfatoken="MX%V6-Q3pQg",
       subscription=[Subscription(subtier="Free")]
   )
   user_2 = Users(
       usrid=uuid.UUID("2bca8c6d-a738-4ad6-880d-29cb3af07909"),
       username="JaeMason1234",
       email="masonry7@hotmail.com",
       passwordhash="lQEBgAdu1D5",
       passwordsalt="AJ0%LwrL$z",
       mfatoken="FA!vI_t5mcW",
       subscription=[Subscription(subtier="Premium"),
Subscription(subtier='Free')]
   )
   user_3 = Users(
      usrid=uuid.UUID("586c6e9f-2d83-4a09-be8f-03e352eaf0ef"),
      username="mortimersmither",
      email="r&m618@gmail.com",
      passwordhash="B!6H4cBp%9h",
      passwordsalt="ez-F9pk41SP",
      mfatoken="Tm&*2wbt7PY",
      subscription=[Subscription(subtier="Bestie for Life"),
                    Subscription(subtier='Free')]
  )
   user_4 = Users(
      usrid=uuid.UUID("0e607745-f99a-4cbc-aa5c-bc7f2e0c6af3"),
      username="tony_stank88",
      email="male.robo@yahoo.com",
      passwordhash="h!azV^24Y*d",
      passwordsalt="HQipskybvWw",
      mfatoken="8E!A4plaamU",
      subscription=[Subscription(subtier="Bestie for Life"),
                    Subscription(subtier='Free')]
  )
   user_5 = Users(
```

```
    usrid=uuid.UUID("f769d227-5e38-4313-80f5-532b81c32185"),
    username="algal",
    email="alex22@gmail.com",
    passwordhash="ddlnv1atwbu2vyfy3g2kf",
    passwordsalt="m4726nasx9gtdqi4mkrj4",
    mfatoken="tddq69utokdat7ydf263h",
    subscription=[Subscription(subtier="Premium"),
                  Subscription(subtier='Free')]
)
 user_6 = Users(
    usrid=uuid.UUID("ee84defc-0fef-45de-8418-c1cc02a1f90f"),
    username="epic_dude55$",
    email="guy_smith23@aol.com",
    passwordhash="7tfa7ls5upsqwdofntgvs",
    passwordsalt="0xelmdsc413xtk4x0xngz",
    mfatoken="6to1crby5zb7lew5oppge",
    subscription=[Subscription(subtier="Bestie for Life"),
                  Subscription(subtier='Free')]
)
 user_7 = Users(
    usrid=uuid.UUID("55ff8590-2c7d-4270-8ac6-b8bba37f00a1"),
    username="W0WEnjoyer",
    email="worldofworlds@yahoo.com",
    passwordhash="zk0qv0fmxq6yh85bbw9dg",
    passwordsalt="6g36w54d0m6oklnx19fzv",
    mfatoken="g8b8daj9f9jp967g7wdle",
    subscription=[Subscription(subtier='Free')]
)
 user_8 = Users(
    usrid=uuid.UUID("40e4de55-3596-4ef7-9ae9-9f763667e531"),
    username="avidTurkey",
    email="turksny2@outlook.com",
    passwordhash="1pl43bwk6e8debo9a2zyj",
    passwordsalt="hufy9borev9lmcfx8lv5j",
    mfatoken="53v1se4qk2obzqobtlo2n",
    subscription=[Subscription(subtier='Free')]
)
 if actuallyWriteDataToTheDatabase:
     session.add_all([user_1, user_2, user_3, user_4,
                      user_5, user_6, user_7, user_8])
     session.commit()
 # Inserting subs
```

```
subscription1 = Subscription(
    subid = uuid.UUID("05a28cd3-05ab-453e-9ee9-c65d6792ab91"),
    subtier = "Free",
    userid = uuid.UUID("4fc51a78-8d6c-4c64-9c68-a005558a0444")
)
subscription2 = Subscription(
    subid = uuid.UUID("ffd349b2-6c60-4c78-b948-0f5c33ecf3e9"),
    subtier = "Premium",
    subprice = 9.99,
    outstandingbalance = 0.00,
    nextduedate = datetime.datetime.now() + datetime.timedelta(days=45),
    annualbilling = True,
    userid = uuid.UUID("2bca8c6d-a738-4ad6-880d-29cb3af07909")
)
subscription3 = Subscription(
    subid = uuid.UUID("cb9aabca-ceb1-40e3-93d7-6b47b27a9b9d"),
    subtier = "Bestie for Life",
    subprice = 49.99,
    outstandingbalance = 0.00,
    nextduedate = datetime.datetime.now() + datetime.timedelta(days=3),
    annualbilling = True,
    userid = uuid.UUID("586c6e9f-2d83-4a09-be8f-03e352eaf0ef")
)
subscription4 = Subscription(
    subid = uuid.UUID("3ac10e2e-d2ad-4f3d-90d9-487a1500470a"),
    subtier = "Bestie for Life",
    subprice = 49.99,
    outstandingbalance = 0.00,
    nextduedate = datetime.datetime.now() + datetime.timedelta(days=8),
    annualbilling = True,
    userid = uuid.UUID("0e607745-f99a-4cbc-aa5c-bc7f2e0c6af3")
)
subscription5 = Subscription(
    subid = uuid.UUID("886a3851-01e7-4dbf-bcac-c3496f4139d3"),
    subtier = "Premium",
    subprice = 9.99,
    outstandingbalance = 0.00,
    nextduedate = datetime.datetime.now() + datetime.timedelta(days=14),
    annualbilling = True,
    userid = uuid.UUID("f769d227-5e38-4313-80f5-532b81c32185")
)
subscription6 = Subscription(
```

```
        subid = uuid.UUID("b2442bcd-9bcf-4e21-8ca7-c5bed5fd6854"),
        subtier = "Bestie for Life",
        subprice = 49.99,
        outstandingbalance = 0.00,
        nextduedate = datetime.datetime.now() + datetime.timedelta(days=10),
        annualbilling = True,
        userid = uuid.UUID("ee84defc-0fef-45de-8418-c1cc02a1f90f")
    )
    subscription7 = Subscription(
        subid = uuid.UUID("855ba433-95b4-46d3-b9b5-ba2d46bc4691"),
        subtier = "Free",
        userid = uuid.UUID("55ff8590-2c7d-4270-8ac6-b8bba37f00a1")
    )
    subscription8 = Subscription(
        subid = uuid.UUID("9a8f28cc-001a-4b31-b86a-bf6d96369e57"),
        subtier = "Free",
        userid = uuid.UUID("40e4de55-3596-4ef7-9ae9-9f763667e531")
    )
    if actuallyWriteDataToTheDatabase:
        session.add_all([subscription1, subscription2, subscription3,
subscription4,
                         subscription5,subscription6, subscription7,
subscription8])
        session.commit()
```

```
# Insert data - Fiona
with Session(engine) as session :
   chat_1 = Chat(
       chatid = uuid.UUID("294441f3-87d3-4989-96bd-9dfb7c9d5325"),
       chtisactive=True,
       chtlastmestimestamp=datetime.datetime.now(),
       messages=[
           Message(mestext="Hi Vision! Do you play Magic the Gathering ?",
                   mesdate=datetime.datetime.now()- datetime.timedelta(days=1),
                   chatid=uuid.UUID("294441f3-87d3-4989-96bd-9dfb7c9d5325"),
                   senderid=uuid.UUID("8f11b297-0e06-4d8e-ac29-11b3818c6375")),
           Message(mestext="What is up Wanda! Ya! I just got the new duskmourn
deck.",
                   mesdate=datetime.datetime.now(),
                   chatid=uuid.UUID("294441f3-87d3-4989-96bd-9dfb7c9d5325"),
                   senderid=uuid.UUID("0090798d-60f8-483c-9516-b4ebe4bd55d3")),
           Message(mestext="Thats awesome! Do you want to get together and play
MTG commandard ?",
                   mesdate=datetime.datetime.now(),
                   chatid=uuid.UUID("294441f3-87d3-4989-96bd-9dfb7c9d5325"),
                   senderid=uuid.UUID("8f11b297-0e06-4d8e-ac29-11b3818c6375")),
           Message(mestext="Yes! That sounds like fun! ",
                   mesdate=datetime.datetime.now(),
                   chatid=uuid.UUID("294441f3-87d3-4989-96bd-9dfb7c9d5325"),
                   senderid=uuid.UUID("0090798d-60f8-483c-9516-b4ebe4bd55d3")),
           Message(mestext="Great! Is it okay for me to invite two of my
friends for the game? ",
                   mesdate=datetime.datetime.now(),
                   chatid=uuid.UUID("294441f3-87d3-4989-96bd-9dfb7c9d5325"),
                   senderid=uuid.UUID("8f11b297-0e06-4d8e-ac29-11b3818c6375")),
           Message(mestext="Ya! Commander is fun with a group of 4 people  ",
                   mesdate=datetime.datetime.now(),
                   chatid=uuid.UUID("294441f3-87d3-4989-96bd-9dfb7c9d5325"),
                   senderid=uuid.UUID("0090798d-60f8-483c-9516-b4ebe4bd55d3"))
       ]
   )
   chat_2 = Chat(
       chatid = uuid.UUID("53e4898d-e1b6-4a38-9721-1982280cdfb6"),
       chtisactive=True,
       chtlastmestimestamp=datetime.datetime.now() -
datetime.timedelta(days=4),
       messages=[
           Message(mestext="Hi Emma, what is your favorite music genre ?",
                   mesdate=datetime.datetime.now()- datetime.timedelta(days=3),
                   chatid=uuid.UUID("53e4898d-e1b6-4a38-9721-1982280cdfb6"),
                   senderid=uuid.UUID("e431e4b0-9a77-4ac6-b4ab-0931fedc8a75")),
           Message(mestext="Hi Erik, I love alternative pop music?",
                   mesdate=datetime.datetime.now()- datetime.timedelta(days=4),
                   chatid=uuid.UUID("53e4898d-e1b6-4a38-9721-1982280cdfb6"),
                   senderid=uuid.UUID("fe6a275d-03d5-465f-8748-69ad732e3f7b")),
       ]
   )
   chat_3 = Chat(
       chatid = uuid.UUID("cf63c98f-97a6-491c-be7c-1b905dd46c9b"),
```

```
        chtisactive=False,
        chtlastmestimestamp=datetime.datetime.now() -
datetime.timedelta(days=40),
        messages=[
            Message(mestext="Hi Marisol, do you like running ?",
                    mesdate=datetime.datetime.now()-
datetime.timedelta(days=50),
                    chatid=uuid.UUID("cf63c98f-97a6-491c-be7c-1b905dd46c9b"),
                    senderid=uuid.UUID("33d4d3ac-576c-4d11-85b1-bbb9ebaa38e6")),
            Message(mestext="Yes! I just finished the Chicago Marathon" +
                    " and hoping to run in New York next year.",
                    mesdate=datetime.datetime.now()-
datetime.timedelta(days=40),
                    chatid=uuid.UUID("cf63c98f-97a6-491c-be7c-1b905dd46c9b"),
                    senderid=uuid.UUID("65523ca9-4f42-41fc-a8e1-84863ce8fa43")),
        ]
    )
    if actuallyWriteDataToTheDatabase:
        session.add_all([chat_1, chat_2, chat_3])
        session.commit()
```

```
# Hannah's Insert Data
with Session(engine) as session:
    # Insert account data
    account1 = Account(
        actid = uuid.UUID("6a6a9838-0ead-4988-b9cc-b2cc77cfcd34"),
        bio = "looking for a square dancing partner!",
        firstname = "Stephanie",
        lastname = "Gomez",
        actstreetaddress = "3670 Charlemaine Dr",
        actcity = "Aurora",
        actstate = "IL",
        #actlocgps_x = 41.73340246308718,
        #actlocgps_y = -88.22654074602464,
        userid = uuid.UUID("4fc51a78-8d6c-4c64-9c68-a005558a0444")
    )
    account2 = Account(
        actid = uuid.UUID("56d321bd-b33e-4b8a-b8b7-941d3fea8dc7"),
        bio = "Ready to make new friends in Chicago!",
        firstname = "Joseph",
        lastname = "Salvator",
        actstreetaddress = "403 W 2nd St",
        actcity = "North Platte",
        actstate = "NE",
        #actlocgps_x = 41.13538421572084,
        #actlocgps_y = -100.76768205411464,
        userid = uuid.UUID("586c6e9f-2d83-4a09-be8f-03e352eaf0ef")
    )
    account3 = Account(
        actid = uuid.UUID("6ed8a920-0c4c-4ba9-be1d-0fc3fa215338"),
        bio = "Looking for company at sunday night bingo!",
        firstname = "Sammi",
        lastname = "Slante",
        actstreetaddress = "624 Wyandotte St",
        actcity = "Kansas City",
        actstate = "MO",
        #actlocgps_x = 39.10595603996265,
        #actlocgps_y = -94.58570393663828,
        userid = uuid.UUID("0e607745-f99a-4cbc-aa5c-bc7f2e0c6af3")
    )
    account4 = Account(
        actid = uuid.UUID("16bf9d27-ddcc-4b51-8909-7e381ddb2543"),
        bio = "Love to read, practice yoga, and bike!",
        firstname = "Susan",
        lastname = "Clapper",
        actstreetaddress = "12980 Silver Fox Dr",
        actcity = "Lemont",
        actstate = "IL",
        #actlocgps_x = 41.653121331513844,
        #actlocgps_y = -87.94838069975592,
        userid = uuid.UUID("f769d227-5e38-4313-80f5-532b81c32185")
    )
    account5 = Account(
        actid = uuid.UUID("4842d59b-dfdb-4b71-92b7-3647a1967157"),
        bio = "Ready to meet new friends in Chicago",
```

```
        firstname = "Mark",
        lastname = "Sanders",
        actstreetaddress = "821 N Mohawk St",
        actcity = "Chicago",
        actstate = "IL",
        #actlocgps_x = 41.89728457057297,
        #actlocgps_y = -87.641059003511052,
        userid = uuid.UUID("ee84defc-0fef-45de-8418-c1cc02a1f90f")
    )
    account6 = Account(
        actid = uuid.UUID("9e0d1b78-1782-4311-b4de-ace18e339ac4"),
        bio = "Football 4 eva ! Bear down.",
        firstname = "Jens",
        lastname = "Peters",
        actstreetaddress = "821 N Mohawk St",
        actcity = "Chicago",
        actstate = "IL",
        #actlocgps_x = 41.89728457057297,
        #actlocgps_y = -87.641059003511052,
        userid = uuid.UUID("55ff8590-2c7d-4270-8ac6-b8bba37f00a1")
    )
    if actuallyWriteDataToTheDatabase:
        session.add_all([account1, account2, account3, account4, account5,
account6])
        session.commit()

    # Insert participates in
    participatesin1 = ParticipatesIn(
        actid = uuid.UUID("16bf9d27-ddcc-4b51-8909-7e381ddb2543"),
        chatid = uuid.UUID("1f697b04-3fed-47b9-87b6-02f68d04ac21"),
        startdate = datetime.datetime.now() - datetime.timedelta(days=42)
    )
    participatesin2 = ParticipatesIn(
        actid = uuid.UUID("4842d59b-dfdb-4b71-92b7-3647a1967157"),
        chatid = uuid.UUID("1f697b04-3fed-47b9-87b6-02f68d04ac21"),
        startdate = datetime.datetime.now() - datetime.timedelta(days=42)
    )
    participatesin3 = ParticipatesIn(
        actid = uuid.UUID("9e0d1b78-1782-4311-b4de-ace18e339ac4"),
        chatid = uuid.UUID("6312dd54-47b0-4d8d-b21c-5cdda64739e6"),
        startdate = datetime.datetime.now() - datetime.timedelta(days=7)
    )
    participatesin4 = ParticipatesIn(
        actid = uuid.UUID("6a6a9838-0ead-4988-b9cc-b2cc77cfcd34"),
        chatid = uuid.UUID("6312dd54-47b0-4d8d-b21c-5cdda64739e6"),
        startdate = datetime.datetime.now() - datetime.timedelta(days=7)
    )
    participatesin5 = ParticipatesIn(
        actid = uuid.UUID("9e0d1b78-1782-4311-b4de-ace18e339ac4"),
        chatid = uuid.UUID("aceec0e9-bf69-4e19-8c4b-5b9a7d3957cd"),
        startdate = datetime.datetime.now() - datetime.timedelta(days=7)
    )
    participatesin6 = ParticipatesIn(
        actid = uuid.UUID("56d321bd-b33e-4b8a-b8b7-941d3fea8dc7"),
```

```
        chatid = uuid.UUID("aceec0e9-bf69-4e19-8c4b-5b9a7d3957cd"),
        startdate = datetime.datetime.now() - datetime.timedelta(days=7)
    )
    if actuallyWriteDataToTheDatabase:
        session.add_all([participatesin1, participatesin2, participatesin3,
participatesin4, participatesin5, participatesin6])
        session.commit()

    # Insert chat data
    chat1 = Chat(
        chatid = uuid.UUID("1f697b04-3fed-47b9-87b6-02f68d04ac21"),
        chtisactive=False,
        chtlastmestimestamp=datetime.datetime.now() -
datetime.timedelta(days=40)
    )
    chat2 = Chat(
        chatid = uuid.UUID("6312dd54-47b0-4d8d-b21c-5cdda64739e6"),
        chtisactive=True,
        chtlastmestimestamp=datetime.datetime.now() -
datetime.timedelta(hours=2),
        messages=[
            Message(mestext="Hi it's steph! So sorry about that field goal last
week.",
                    mesdate=datetime.datetime.now()-
datetime.timedelta(days=3)),
            Message(mestext="Stephanie, lets go to the next game together!!1",
                    mesdate=datetime.datetime.now()-
datetime.timedelta(hours=2)),
        ]
    )
    chat3 = Chat(
        chatid = uuid.UUID("aceec0e9-bf69-4e19-8c4b-5b9a7d3957cd"),
        chtisactive=True,
        chtlastmestimestamp=datetime.datetime.now() -
datetime.timedelta(days=2),
        messages=[
            Message(mestext="I'm visiting Chicago next week, lets hang!",
                    mesdate=datetime.datetime.now()-
datetime.timedelta(days=3)),
            Message(mestext="sick i'll be your tour guide.",
                    mesdate=datetime.datetime.now()-
datetime.timedelta(days=2)),
        ]
    )
    if actuallyWriteDataToTheDatabase:
        session.add_all([chat1, chat2, chat3])
        session.commit()
```

```
# Join Query by Hannah

# This join query displays the account information for all accounts
who have participated in more than 1 chat.

print("\n## Join Query - Hannah ##")
stmt = (
    select(Account)
    .join(ParticipatesIn, Account.actid == ParticipatesIn.actid)
    .group_by(Account.actid)
    .having(func.count(ParticipatesIn.chatid) > 1)
    .order_by(Account.actid)
)

for record in session.scalars(stmt):
    print(f"""[Hannah] Account ID={record.actid}
              First Name={record.firstname}
              Last Name={record.lastname}""")
```

```
## Join Query - Hannah ##
[Hannah] Account ID=072cb76c-8ca0-43f2-a515-8418d1931193
              First Name=Katy
              Last Name=Perry
[Hannah] Account ID=2493c668-be9f-4485-beb7-48dbf596f983
              First Name=kay
              Last Name=None
[Hannah] Account ID=257d8621-2e49-442b-96c2-605a651a996d
              First Name=Ann
              Last Name=M
[Hannah] Account ID=4a7536a1-3782-451b-a889-ecc18e6b7fb3
              First Name=John
              Last Name=Smith
[Hannah] Account ID=56131135-346e-46f2-84e3-1565178e1164
              First Name=AzureDiamond
              Last Name=None
[Hannah] Account ID=65523ca9-4f42-41fc-a8e1-84863ce8fa43
              First Name=Marisol
              Last Name=None
[Hannah] Account ID=b079763a-21fd-409e-89a8-ccd1d570848f
              First Name=Free-Robux_website
              Last Name=None
[Hannah] Account ID=c00605f5-dc5e-41df-83e3-36e4f6427284
              First Name=Max
              Last Name=Madewell
[Hannah] Account ID=d7165614-5e9c-4d7c-9003-102dd83545ce
              First Name=Amanda
              Last Name=Smith
[Hannah] Account ID=f1b21a51-c213-4e83-bc50-b821f317bea4
              First Name=Berry
              Last Name=Benson
```

```
# Join Query by Josh

# This query shows which credit cards are being used to cover which
non-free-tier subscriptions.
statement = (
    select(paymentmethod)
    .join(paymentmethod.subscription)
    .where(
        Subscription.subtier != "Free" and
        paymentmethod.pmttype != payment_type.thirdParty
    )
    .order_by(Subscription.nextduedate)
)
for record in session.scalars(statement):
    print(f"""[Josh] Card {record.num} is being used to cover
subscription for tier {record.subscription.subtier}
            (id={record.subid}), next due
{record.subscription.nextduedate.strftime("%B %d, %Y")}""")
```

```
[Josh] Card 4798315489743516 is being used to cover subscription for tier Bestie for Life
          (id=4d39e47b-b878-49f9-b958-493aebbb4a18), next due November 10, 2024
[Josh] Card 1750161067181961 is being used to cover subscription for tier Bestie for Life
          (id=ef7b5717-96f2-42b4-8bb2-23ae9172ff65), next due November 23, 2024
[Josh] Card 4381946282957154 is being used to cover subscription for tier Bestie for Life
          (id=ef7b5717-96f2-42b4-8bb2-23ae9172ff65), next due November 23, 2024
[Josh] Card 6195016491745164 is being used to cover subscription for tier Bestie for Life
          (id=ef7b5717-96f2-42b4-8bb2-23ae9172ff65), next due November 23, 2024
[Josh] Card 4532970356782245 is being used to cover subscription for tier Bestie for Life
          (id=1cb5b930-62ed-47f0-8a42-52c8f9ed0a2b), next due December 15, 2024
[Josh] Card 371449635398431 is being used to cover subscription for tier Premium
          (id=e8368d90-94ff-47c4-9f9b-428b1b456d85), next due December 17, 2024
[Josh] Card 6011592898763443 is being used to cover subscription for tier Bestie for Life
          (id=56669135-539c-4993-911a-736f9b237916), next due December 30, 2024
[Josh] Card 9874654386546348 is being used to cover subscription for tier Premium
          (id=b307e71b-bbe4-4055-b488-d01ab711a5bd), next due February 20, 2025
```

```
# Join Query by Fiona

# This join query looks at a specific chat that IS active and
# is using the chat id for a specific chat.  It will print out
# the conversation (messages) are between the user_accounts.

print("\n## Join Query -  Fiona  ##")
stmt = (
    select(Message)
    .join(Message.chat)
    .where(Chat.chtisactive == True)
    .where(Chat.chatid ==
uuid.UUID("294441f3-87d3-4989-96bd-9dfb7c9d5325"))
    .order_by(Message.chatid)
    .order_by(Message.mesid)
)

for record in session.scalars(stmt) :
    print(f"""[Fiona] message id={record.mesid} chat_id={record.chatid}
             message date={record.mesdate}
             message Text={record.mestext} """)
```

```
## Join Query -  Fiona  ##
message id=03a9c586-a4d5-4089-aa34-1198ea8fa3bc chat_id=294441f3-87d3-4989-96bd-9dfb7c9d5325
  message date=2024-11-18 15:04:25.461850-06:00
  message Text=Hi Vision! Do you play Magic the Gathering ?


message id=081195c2-9058-4e10-8316-ed2a5cd69856 chat_id=294441f3-87d3-4989-96bd-9dfb7c9d5325
  message date=2024-11-19 15:04:25.463839-06:00
  message Text=Thats awesome! Do you want to get together and play MTG commandard ?


message id=0c6a58a0-5fc0-44a6-8c39-54c14a8d553f chat_id=294441f3-87d3-4989-96bd-9dfb7c9d5325
  message date=2024-11-19 15:04:25.463824-06:00
  message Text=What is up Wanda! Ya! I just got the new duskmourn deck.


message id=549e76df-c525-4153-a450-51d81142bb5d chat_id=294441f3-87d3-4989-96bd-9dfb7c9d5325
  message date=2024-11-19 15:04:25.463858-06:00
  message Text=Great! Is it okay for me to invite two of my friends for the game?


message id=8fdcd137-5917-4507-9c46-d139503e76ea chat_id=294441f3-87d3-4989-96bd-9dfb7c9d5325
  message date=2024-11-19 15:04:25.463850-06:00
  message Text=Yes! That sounds like fun!


message id=f4bde6c9-4999-45b7-b9b2-577574421334 chat_id=294441f3-87d3-4989-96bd-9dfb7c9d5325
  message date=2024-11-19 15:04:25.463867-06:00
  message Text=Ya! Commander is fun with a group of 4 people
```

```python
# Join Query by Drew
# This query shows which username has a paid subscription, and what
# type.
statement = (
    select(Users)
    .join(Users.subscription)
    .where(
        Subscription.subtier != "Free"
    )
    .order_by(Subscription.subprice)
)
for record in session.scalars(statement):
    print(f"[Drew] User {record.username} currently has
{record.subscription}")
```

```
[Drew] User JayMorrow87 currently has [Subscription(
            id=UUID('b307e71b-bbe4-4055-b488-d01ab711a5bd'), subtier='Premium', subprice=Decimal('9.99'),
            nextduedate=datetime.datetime(2025, 2, 20, 16, 39), annualbilling=True,
            userid=UUID('829ebcd5-d57f-434d-931b-aec167cf5c03'))]
[Drew] User emmafrost currently has [Subscription(
            id=UUID('c55e92df-b7af-4502-a1dd-37438668a0fe'), subtier='Free', subprice=None,
            nextduedate=None, annualbilling=None,
            userid=UUID('3db2bb9a-2418-4258-bbc4-c7ea32b39570')), Subscription(
            id=UUID('ba2a5760-0ead-4988-b9cc-b2cc77cfcd34'), subtier='Premium', subprice=Decimal('9.99'),
            nextduedate=datetime.datetime(2024, 11, 26, 13, 45, 24, 604107), annualbilling=False,
            userid=UUID('3db2bb9a-2418-4258-bbc4-c7ea32b39570'))]
[Drew] User FreeRobux currently has [Subscription(
            id=UUID('ef7b5717-96f2-42b4-8bb2-23ae9172ff65'), subtier='Bestie for Life', subprice=Decimal('49.99'),
            nextduedate=datetime.datetime(2024, 11, 23, 18, 45, 24, 615302), annualbilling=False,
            userid=UUID('a87b4c9b-f164-4e4b-99a4-2d64f2212ca7'))]
[Drew] User toddy846 currently has [Subscription(
            id=UUID('4d39e47b-b878-49f9-b958-493aebbb4a18'), subtier='Bestie for Life', subprice=Decimal('49.99'),
            nextduedate=datetime.datetime(2024, 11, 10, 16, 39), annualbilling=True,
            userid=UUID('6ecd2aaa-4d44-4b9e-990d-409773324c5b'))]
[Drew] User steverogers currently has [Subscription(
            id=UUID('aa8225c4-48f4-4012-bc78-579c0897d710'), subtier='Bestie for Life', subprice=Decimal('49.99'),
            nextduedate=datetime.datetime(2024, 11, 10, 16, 0), annualbilling=True,
            userid=UUID('328ee807-7686-4e08-83cc-0b3d9c111e9b'))]
[Drew] User jsmith currently has [Subscription(
            id=UUID('4f36fd46-c84c-4d7c-868b-85532e580dab'), subtier='Bestie for Life', subprice=Decimal('49.99'),
            nextduedate=datetime.datetime(2024, 12, 10, 16, 39), annualbilling=True,
            userid=UUID('94ab690c-3864-407b-8354-4e606ee0cc70'))]
[Drew] User d0gl0ver213 currently has [Subscription(
            id=UUID('e8368d90-94ff-47c4-9f9b-428b1b456d85'), subtier='Premium', subprice=None,
            nextduedate=datetime.datetime(2024, 12, 17, 16, 39), annualbilling=None,
            userid=UUID('7f702396-b720-43d7-a2d9-f96a1b4d3569'))]
[Drew] User katy_pr3ston currently has [Subscription(
            id=UUID('1cb5b930-62ed-47f0-8a42-52c8f9ed0a2b'), subtier='Bestie for Life', subprice=None,
            nextduedate=datetime.datetime(2024, 12, 15, 16, 39), annualbilling=None,
            userid=UUID('3e5fb29d-ce1b-4f3b-9ced-c35858d389cf'))]
[Drew] User chicagolex currently has [Subscription(
            id=UUID('56669135-539c-4993-911a-736f9b237916'), subtier='Bestie for Life', subprice=None,
            nextduedate=datetime.datetime(2024, 12, 30, 16, 39), annualbilling=None,
            userid=UUID('bebaaef9-e9a0-40a2-9a65-ab905b164ad7'))]
```

GROUP STATUS REPORT

GROUP #: 1 **GROUP NAME:** Matchmakers **PHASE #:** 3

Dates & attendance at group meetings in this phase:

Wednesday	Nov 6	3:30 PM - 4:00 PM	All group members
Tuesday	Nov 12	3:45 PM - 4:30 PM	All group members
Monday	Nov 18	5:30 PM - 6:30PM	Drew and Josh
Tuesday	Nov 19	12:00PM - 2:15PM	All group members

Overview of progress on project as of DATE:

Python code for the ORM was added, along with the corresponding classes which are User, Subscription, Payment_type, Paymentmethod, Account, ParticipatesIn, Chat, and Messages. Each group member created insert data to use in a join query they also created. A screenshot of the output of the join query as added to show the results.
NOTE: GPS was not able to be added to sqlAlchemy so x and y float points were added. Also, the cardinality between Users → Subscription was changed to 1-N, from 1-1, for the sake of the Phase 3 ORM.

Contributions of Group Members

Leader: Drew Patterson
- Created User Class
- Created insert statements for users and subscriptions
- Created query which checks to see which users have paid subscriptions
- Collaborated with Josh to create Subscriptions class

Recorder: Hannah Serio
- Created ORM class for Account and ParticipatesIn
- Created ORM class for chat with Fiona
- Inserted data for Account, ParticipatesIn, and Chat
- Created a Join Query for Account and Chat. The query gets the accounts that participate in more than one chat.

Phase Checker: Josh Honig
- Created Subscription, payment_type, paymentmethod classes
- Created insert statements for subscriptions and payment methods
- Contributed query which checks to see which cards are covering which subscriptions
- Helped to collate all group members' code

Technical Advisor: Fiona
- Created ORM class for Chat with Hannah
- Created ORM class for Message
- Insert data for Chat and Message
- Created a ORM Join Query for Chat and Message, that get a specific chat with all the messages
- Check everyone's ORM to see if the code runs