University of British Columbia, Vancouver

Department of Computer Science

CPSC 304 Project - Subletter

Milestone #2

Date: July 27, 2023

Group Number: 11

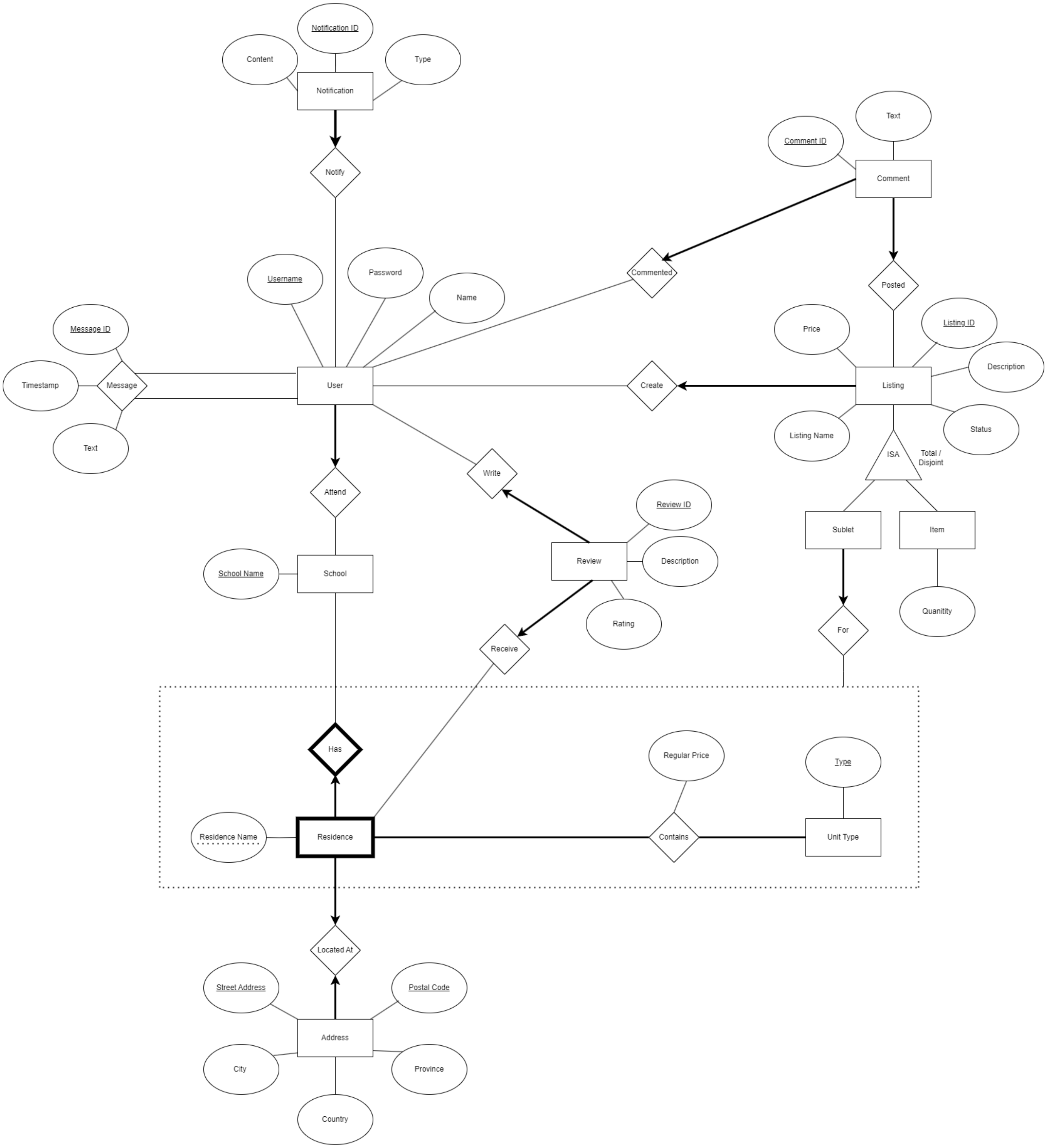
Members

Name	Student Number	CS Alias (User ID)	E-mail Address
Andy Hu	58787557	y3f6z	andy45.hu@gmail.com
Imaad Junaidi	89417463	j0w8b	imaadj10@student.ubc.ca
Yu Cheng Li	24714545	k8p8k	ychengli11065@gmail.com

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

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

ER Diagram


Notification ID
Content
Type
Notification
Notify
Text
Comment ID
Comment
Commented
Posted
Username
Password
Name
Message ID
Price
Listing ID
Description
Timestamp
Message
User
Create
Listing
Listing Name
Status
ISA
Total / Disjoint
Text
Attend
Write
Review ID
Sublet
Item
School Name
School
Review
Description
Rating
Receive
For
Quanitity
Has
Regular Price
Type
Residence Name
Residence
Contains
Unit Type
Located At
Street Address
Postal Code
Address
City
Province
Country

ER Diagram Changes From Milestone 1

Residences to weak entity

- We did this because we realized that residences don’t exist by themselves, and are rather also identified by the school they belong to, such that the school name and the residence name should rather be the primary key for residences, as opposed to the past where we had residences identified by their name and address details.

Units to Unit Types, and not weak anymore

- In the past, we had unit types as a weak entity to residences, but we realized that the unit type itself can exist despite the layouts of whichever residences, and that residences can have certain unit types to them.

Aggregation of sublet relationship for ‘contains’

- We realized that sublets are on a type of room in a residence, hence why we decided to use an aggregation on that relationship between residence and unit type. This more accurately describes our sublet information.

Add address entity to residence, removing those attributes from residence

- We realized that residence was being clustered with a lot of attributes that could easily be broken apart into a new entity on its own, so we made an address entity which encompassed all of the location details of residences and gave a one-to-one relationship between them with total participation.

Add quantity attribute for item

- We realized that items had no virtual difference to a listing, and thought that users may want to specify the number of items they have for an item posting.

Remove ‘useless attributes’ like date and timestamps of dates

- We realized we had a lot of information that wasn’t really necessary for some entities, and decided to remove them for better simplicity.

BEFORE NORMALIZATION

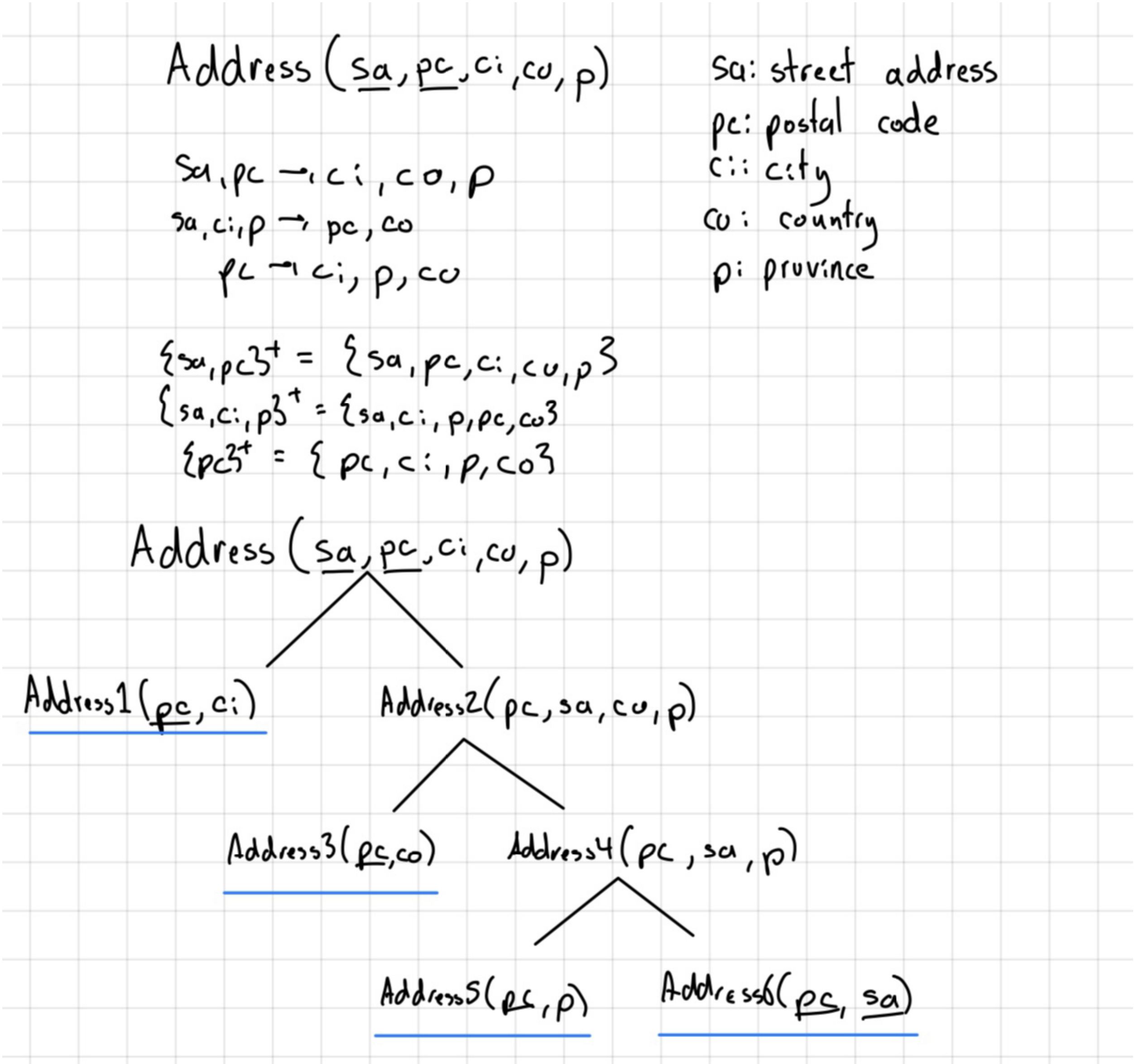
- Notifications(Notification ID: integer, **Username**: char(32), Content: char(128), Type: char(64))
  - Username not null
  - Candidate Keys: Notification ID
  - Functional Dependencies:
    - ID -> Username, content
    - Content -> Type
  - Normalization: 2NF
- Users(Username: char(32), Password: char(32), Name: char(32), **School Name**: char(128))
  - All attributes not null
  - Candidate Keys: Username
  - Functional Dependencies:
    - Username -> Password, Name, School Name
  - Normalization: BCNF
- Messages(**SenderID**: char(32), **ReceiverID**: char(32), MessageID: char(36), Timestamp: date, Text: char(512))
  - Candidate Keys: SenderID/ReceiverID/MessageID
  - Functional Dependencies:
    - MessageID -> SenderID, ReceiverID, Timestamp, Text
    - SenderID, Receiver ID, Timestamp -> Text, MessageID
  - Normalization: BCNF
- Schools(School Name: char(128))
  - Candidate Keys: School Name
  - Normalization: BCNF
- Residences(Residence Name: char(128),**School Name**: char(128), **Street Address**: char(128), **Postal Code**: char(16))
  - Combination of Street Address and Postal Code must be unique
  - Street Address, Postal Code not null
  - Candidate Keys: Residence Name/School Name, Residence Name/Street Address/Postal Code
  - Normalization: BCNF
- Addresses(Street Address: char(128), Postal Code: char(16), City: char(128), Country: char(128), Province: char(128))
  - Candidate Keys: Street Address/Postal Code
  - Functional Dependencies:
    - Street Address/PostalCode -> City, Country, Province
    - Street Address/City/Province -> Postal Code, Country

        - Postal Code -> City, Country, Province
    - Normalization: 1NF
- UnitTypes(Type: char(128))
    - Candidate Keys: Type
    - Normalization: BCNF
- Contains(**Residence Name**: char(128), **School Name**: char(128), **Type**: char(128), Price: real)
    - Candidate Keys: Residence Name/School Name/Type
    - Functional Dependencies:
        - Residence Name, School Name, Type -> Price
    - Normalization: BCNF
- Written_Reviews(Review ID: char(36), **Username**: char(32), Description: char(1024), Rating: real)
    - Username not null
    - Candidate Keys: Review ID
    - Functional Dependencies:
        - Review ID -> Username, Residence Name, School Name, Description, rating
    - Normalization: BCNF
- Received_Reviews(**Review ID**: char(36), **Residence Name**: char(128), **School Name**: char(128)
    - Candidate Keys: Review ID/Residence Name/School Name
    - Normalization: BCNF
- Listings(Listing ID: integer, **Username**: char(32), Description: char(1024), Status: bool, Listing Name: char(128), Price: real)
    - Username not null
    - Candidate Keys: Listing ID
    - Functional Dependencies:
        - Listing ID -> Username, Description, Status, Listing name, Price
    - Normalization: BCNF
- Sublets(**Listing ID**: integer, **Type**: char(128), **Residence Name**: char(128), **School Name**: char(128))
    - All attributes not null
    - Candidate Keys: Listing ID
    - Functional Dependencies:
        - Listing ID -> Residence Name, School Name, Type
    - Normalization: BCNF
- Items(**Listing ID**: integer, Quantity: integer)
    - Candidate Keys: Listing ID
    - Functional Dependencies:

        - Listing ID -> Quantity
    - Normalization: BCNF
- Comments(Comment ID: integer, **Username**: char(32), **Listing ID**: integer, Text: char(1024))
    - Username, listing ID not null
    - Candidate Keys: Comment ID
    - Functional Dependencies:
        - Comment ID -> username, listing ID, text
    - Normalization: BCNF

After Normalization:

We had to further decompose on Address and Notification.



Address($\underline{sa}$, $\underline{pc}$, ci, co, p)

sa: street address
pc: postal code
ci: city
co: country
p: province

$sa, pc \rightarrow ci, co, p$

$sa, ci, p \rightarrow pc, co$

$pc \rightarrow ci, p, co$

$\{sa, pc\}^+ = \{sa, pc, ci, co, p\}$

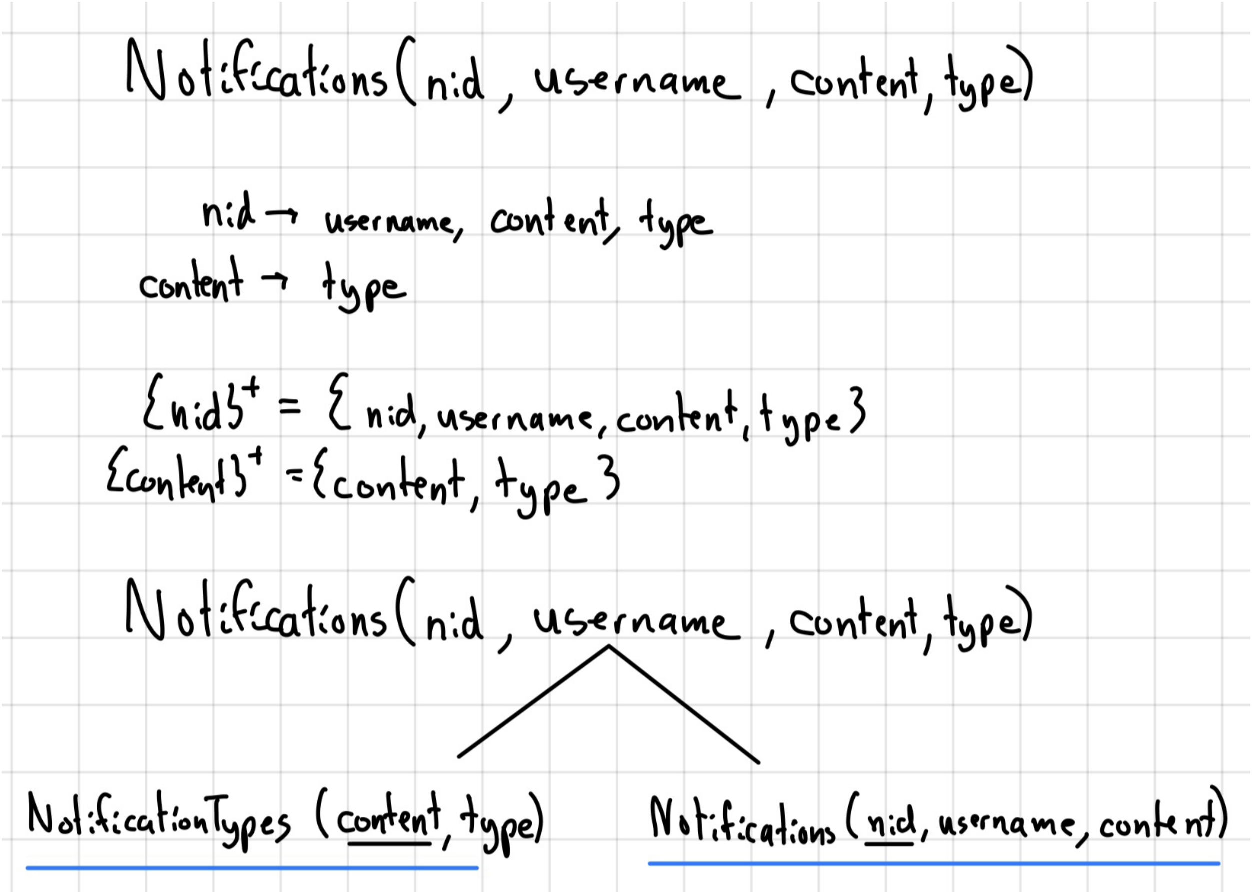
$\{sa, ci, p\}^+ = \{sa, ci, p, pc, co\}$

$\{pc\}^+ = \{pc, ci, p, co\}$

- Addresses1(<u>Postal Code</u>: char(16), City: char(128))
  - Candidate Keys: Postal Code
  - Functional Dependencies:
    - Postal Code -> City
  - Normalization: BCNF
- Addresses2(<u>Postal Code</u>: char(16), Country: char(128))
  - Candidate Keys: Postal Code

- ○ Functional Dependencies:
  - ■ Postal Code -> Country
- ○ Normalization: BCNF

- Addresses3(Postal Code: char(16), Province: char(128))
  - Candidate Keys: Postal Code
  - Functional Dependencies:
    - ■ Postal Code -> Province
  - Normalization: BCNF
- AddressesMain(Postal Code: char(16), Street Address: char(128))
  - Candidate Keys: Postal Code/Street Address
  - Normalization: BCNF



Notifications(nid, username, content, type)

nid → username, content, type

content → type

$\{nid\}^+ = \{nid, username, content, type\}$

$\{content\}^+ = \{content, type\}$

Notifications(nid, username, content, type)

$\text{NotificationTypes}(\underline{\text{content}}, \text{type})$

$\text{Notifications}(\underline{\text{nid}}, \text{username}, \text{content})$

- Notifications(Notification ID: integer, **Username**: char(32), Content: char(128),)
  - Username not null
  - Candidate Keys: Notification ID
  - Functional Dependencies:

        - ID -> Username, content
    - Normalization: BCNF
- Notification_Types(Content: char(128), Type: char(64))
    - Candidate Keys: Content
    - Functional Dependencies:
        - Content -> Type
    - Normalization: BCNF

Table Creation - SQL DDL

```
CREATE TABLE Notification_Types (
  content VARCHAR(128),
  type VARCHAR(64),
  PRIMARY KEY (content)
  );

CREATE TABLE Notifications (
  nid INT AUTO_INCREMENT,
  username VARCHAR(32) NOT NULL,
  content VARCHAR(128),
  PRIMARY KEY (nid)
  );

CREATE TABLE Schools (
  school_name VARCHAR(128) PRIMARY KEY
  );

CREATE TABLE Users (
  username VARCHAR(32) PRIMARY KEY,
  `password` VARCHAR(32) NOT NULL,
  name VARCHAR(32) NOT NULL,
  school_name VARCHAR(128) NOT NULL,
  FOREIGN KEY (school_name) REFERENCES Schools(school_name)
          ON UPDATE CASCADE
  );

CREATE TABLE Messages (
  sid VARCHAR(32),
  rid VARCHAR(32),
  mid VARCHAR(36) DEFAULT (UUID()),
  time_sent TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  content VARCHAR(512),
  FOREIGN KEY (sid) references Users(username)
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  FOREIGN KEY (rid) references Users(username)
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  PRIMARY KEY(sid, rid, mid)
  );

CREATE TABLE Units (
  type VARCHAR(128) PRIMARY KEY
  );

CREATE TABLE `Contains` (
  res_name VARCHAR(128),
  school_name VARCHAR(128),
  type VARCHAR(128),
  price DOUBLE,
  FOREIGN KEY (res_name, school_name) REFERENCES Residences(res_name,school_name)
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  FOREIGN KEY (type) REFERENCES Unit_Types(type)
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  PRIMARY KEY(res_name, school_name, type)
  );

CREATE TABLE Listings (
  lid INT AUTO_INCREMENT PRIMARY KEY,
  username VARCHAR(32) NOT NULL,
  description VARCHAR(1024),
  `status` BOOL,
  name VARCHAR(128),
  price DOUBLE,
  FOREIGN KEY(username) REFERENCES Users(username)
          ON UPDATE CASCADE
          ON DELETE CASCADE
  );

CREATE TABLE Sublets (
  lid INT PRIMARY KEY,
  type VARCHAR(128) NOT NULL,
  res_name VARCHAR(128) NOT NULL,
  school_name VARCHAR(128) NOT NULL,
  FOREIGN KEY (lid) REFERENCES listings(lid)
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  FOREIGN KEY (res_name, school_name, type) REFERENCES `Contains`(res_name,
school_name, type)
          ON UPDATE CASCADE
          ON DELETE CASCADE
  );

CREATE TABLE Items (
  lid INT PRIMARY KEY,
  quantity INT,
  FOREIGN KEY (lid) REFERENCES listings(lid)
  );

CREATE TABLE Comments (
  cid INT AUTO_INCREMENT PRIMARY KEY,
  username VARCHAR(32) NOT NULL,
  lid INT NOT NULL,
  content VARCHAR(1024),
  FOREIGN KEY (username) REFERENCES Users(username)
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  FOREIGN KEY (lid) REFERENCES Listings(lid)
          ON UPDATE CASCADE
          ON DELETE CASCADE
 );

CREATE TABLE Written_Reviews (
  rid VARCHAR(36) DEFAULT (UUID()) PRIMARY KEY,
  username VARCHAR(32) NOT NULL,
  description VARCHAR(1024),
  rating DOUBLE,
  FOREIGN KEY (username) references Users(username)
          ON UPDATE CASCADE
          ON DELETE CASCADE
  );

CREATE TABLE Received_Reviews (
  rid VARCHAR(36) DEFAULT (UUID()),
  res_name VARCHAR(128),
  school_name VARCHAR(128),
  FOREIGN KEY (rid) REFERENCES Written_Reviews(rid)
```

```
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  FOREIGN KEY (res_name, school_name) REFERENCES Residences(res_name, school_name)
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  PRIMARY KEY (rid, res_name, school_name)
  );

CREATE TABLE Residences (
  res_name VARCHAR(128),
  school_name VARCHAR(128),
  street_address VARCHAR(128) UNIQUE NOT NULL,
  postal_code VARCHAR(16) NOT NULL,
  FOREIGN KEY (school_name) REFERENCES Schools(school_name)
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  FOREIGN KEY (street_address, postal_code) REFERENCES Addresses_Main(street_address, postal_code)
          ON UPDATE CASCADE
          ON DELETE CASCADE,
  UNIQUE KEY address_constraint (street_address, postal_code);
  PRIMARY KEY (res_name, school_name)
  );

CREATE TABLE Addresses_Main (
  street_address VARCHAR(128),
  postal_code VARCHAR(16),
  PRIMARY KEY (street_address, postal_code)
  );

CREATE TABLE Addresses_1 (
  postal_code VARCHAR(16),
  city VARCHAR(128),
  PRIMARY KEY (postal_code)
  );

CREATE TABLE Addresses_2 (
  postal_code VARCHAR(16),
  country VARCHAR(128),
  PRIMARY KEY (postal_code)
  );

 CREATE TABLE Addresses_3 (
  postal_code VARCHAR(16),
  province VARCHAR(128),
  PRIMARY KEY (postal_code)
  );
```

INSERTIONS

```
INSERT INTO Schools(school_name)
VALUES
   ('University of British Columbia'),
   ('University of Calgary'),
   ('University of Alberta'),
   ('University of Waterloo'),
   ('University of Toronto')

INSERT INTO Schools(school_name)
VALUES
   ('University of British Columbia'),
   ('University of Calgary'),
   ('University of Alberta'),
   ('University of Waterloo'),
   ('University of Toronto');

INSERT INTO Users(username, `password`, name, school_name)
VALUES
   ('Andy', 'Andy', 'Andy', 'University of British Columbia'),
   ('Imaad', 'Imaad', 'Imaad', 'University of British Columbia'),
   ('Yu Cheng', 'Yu Cheng', 'Yu Cheng', 'University of British Columbia'),
   ('Bob', 'Bob', 'Bob', 'University of Calgary'),
   ('Jason', 'Jason', 'Jason', 'University of Waterloo'),
   ('Ronald', 'Ronald', 'Ronald', 'University of Toronto'),
   ('Karen', 'Karen', 'Karen', 'University of Alberta');


INSERT INTO Unit_Types(type)
VALUES
   ('Single Connected'),
  ('Two Bedroom'),
  ('Studio'),
  ('Four Bedroom'),
  ('Nano Suite'),
  ('Townhouse'),
  ('One Bedroom'),
  ('Studio-Small');

INSERT INTO Listings(username, description, `status`, name, price)
VALUES
  ('Andy', 'lamp', TRUE, 'Lamp for sale!',1000),
  ('Imaad', 'drawing pad', TRUE, 'Drawing pad for sale!', 1259),
  ('Yu Cheng', 'old chair', TRUE, 'Chair for sale!', 1500),
  ('Jason', 'new chair', TRUE, 'Better chair for sale!', 1300),
  ('Ronald', 'best sublet deal', TRUE, 'Sublet for sale!', 1100),
  ('Ronald', 'best sublet 2', TRUE, 'Sublet2 for sale!', 1100),
  ('Ronald', 'best sublet 3', TRUE, 'Sublet3 for sale!', 1100),
  ('Ronald', 'best sublet 4', TRUE, 'Sublet4 for sale!', 1100),
  ('Ronald', 'best sublet 5', TRUE, 'Sublet5 for sale!', 1100),
  ('Ronald', 'best item', TRUE, 'Item for sale!', 1100);

INSERT INTO Addresses_Main(street_address, postal_code)
VALUES
	('2205 Lower Mall', 'V6T 1Z4'),
 ('2075 West Mall', 'V6T 1Z2'),
 ('5960 Student Union Blvd', 'V6T 1Z1'),
 ('6363 Agronomy Rd', 'V6T 1Z4'),
 ('6088 Walter Gage Rd', 'V6T 0B4'),
 ('1935 Lower Mall', 'V6T 1X1');

INSERT INTO Addresses_1(postal_code, city)
VALUES
	('V6T 1Z4', 'Vancouver'),
	('V6T 1Z2', 'Vancouver'),
	('V6T 1Z1', 'Vancouver'),
 ('V6T 0B4', 'Vancouver'),
 ('V6T 1X1', 'Vancouver');

INSERT INTO Addresses_2(postal_code, country)
VALUES
```

```
        ('V6T 1Z4', 'Canada'),
        ('V6T 1Z2', 'Canada'),
        ('V6T 1Z1', 'Canada'),
  ('V6T 0B4', 'Canada'),
  ('V6T 1X1', 'Canada');

INSERT INTO Addresses_3(postal_code, province)
VALUES
        ('V6T 1Z4', 'British Columbia'),
        ('V6T 1Z2', 'British Columbia'),
        ('V6T 1Z1', 'British Columbia'),
  ('V6T 0B4', 'British Columbia'),
  ('V6T 1X1', 'British Columbia');

INSERT INTO Residences(res_name, school_name, street_address, postal_code)
VALUES
        ('Marine Drive Residence', 'University of British Columbia', '2205 Lower Mall', 'V6T
1Z4'),
  ('Ponderosa Commons', 'University of British Columbia', '2075 West Mall', 'V6T 1Z2'),
  ('Exchange Student Residence', 'University of British Columbia', '5960 Student Union Blvd',
'V6T 1Z1'),
  ('Orchard Commons', 'University of British Columbia', '6363 Agronomy Rd', 'V6T 1Z4'),
  ('Brock Commons', 'University of British Columbia', '6088 Walter Gage Rd', 'V6T 0B4'),
  ('Place Vanier', 'University of British Columbia', '1935 Lower Mall', 'V6T 1X1');

INSERT INTO Contains(res_name, school_name, type, price)
VALUES
    ('Ponderosa Commons', 'University of British Columbia', 'Two Bedroom', 1300),
    ('Ponderosa Commons', 'University of British Columbia', 'Studio', 1300),
    ('Ponderosa Commons', 'University of British Columbia', 'Four Bedroom', 1050),
    ('Ponderosa Commons', 'University of British Columbia', 'Studio-Small', 1300),
    ('Exchange', 'University of British Columbia', 'Two Bedroom', 1300),
    ('Exchange', 'University of British Columbia', 'Nano Suite', 1050),
    ('Exchange', 'University of British Columbia', 'Studio', 1300),
    ('Exchange', 'University of British Columbia', 'One Bedroom', 1600);

INSERT INTO Sublets(lid, type, res_name, school_name)
VALUES
    (20, 'Studio', 'Ponderosa Commons', 'University of British Columbia'),
    (21, 'Four Bedroom', 'Ponderosa Commons', 'University of British Columbia'),
    (22, 'Nano Suite', 'Exchange Student Residence', 'University of British Columbia'),
    (23, 'One Bedroom', 'Exchange Student Residence', 'University of British Columbia'),
    (24, 'Two Bedroom', 'Exchange Student Residence', 'University of British Columbia');

INSERT INTO Items(lid, quantity)
VALUES
    (11, 2),
    (12, 1),
    (13, 3),
    (14, 4),
    (25, 2);

INSERT INTO Comments(username, lid, content)
VALUES
    ('Imaad', 11, 'One of the worst lamps I have ever seen'),
    ('Andy', 12, 'I am interested!'),
    ('Ronald', 13, 'Where did this chair come from?'),
    ('Karen', 14, 'I am upset because I want this chair for free!!'),
    ('Yu Cheng', 25, 'This is the best');

INSERT INTO Written_Reviews(username, description, rating)
VALUES
    ('Imaad', 'Marine Drive is so old', 6.5),
    ('Andy', 'I enjoyed my time in Exchange!', 10),
    ('Ronald', 'Orchard is alright. My roommate smells.', 2.5),
    ('Karen', 'I am upset I have to pay rent at Brock Commons. It should be free!', 0),
    ('Yu Cheng', 'Delivery driver could not find my building', 2.3);

INSERT INTO Received_Reviews(rid, res_name, school_name)
VALUES
    ('80cc82c3-2cbd-11ee-bef0-806d970faeec', 'Marine Drive Residence', 'University of British
Columbia'),
    ('80cc8a1e-2cbd-11ee-bef0-806d970faeec', 'Exchange Student Residence', 'University of
British Columbia'),
    ('80cc8ba9-2cbd-11ee-bef0-806d970faeec', 'Orchard Commons', 'University of British
Columbia'),
    ('80cc8cb1-2cbd-11ee-bef0-806d970faeec', 'Brock Commons', 'University of British
Columbia'),
    ('80cc8da8-2cbd-11ee-bef0-806d970faeec', 'Marine Drive Residence', 'University of British
Columbia');
```

```
INSERT INTO Messages(sid, rid, content)
VALUES
   ('Imaad', 'Yu Cheng', 'I want McDonalds'),
   ('Andy', 'Ronald', 'Sheesh'),
   ('Karen', 'Bob', 'I have a problem with you Bob'),
   ('Bob', 'Karen', 'Well I do not care'),
   ('Jason', 'Yu Cheng', 'I am not going to clean the kitchen');

INSERT INTO Notification_Types(content, type)
VALUES
         ('You have a new message!', 'Message'),
  ('Welcome to the app!', 'System'),
  ('You have a new comment!', 'Listing'),
  ('There was a login attempt to your account', 'System'),
  ('New sublet available!', 'Listing');

INSERT INTO Notifications(username, content)
VALUES
         ('Imaad', 'You have a new message!'),
  ('Andy', 'Welcome to the app!'),
  ('Yu Cheng', 'You have a new comment!'),
  ('Karen', 'There was a login attempt to your account'),
  ('Ronald', 'New sublet available!');
```