University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: July 16, 2024

Group Number: 11

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Rehnoor Saini	78995446	g0o9s	rehnoorsaini@outlook.com
Abdul Khurram	25159641	f0p7l	arkb25@student.ubc.ca
Udbhav Kansal	42392381	m8z9e	udbhav27@student.ubc.ca

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

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

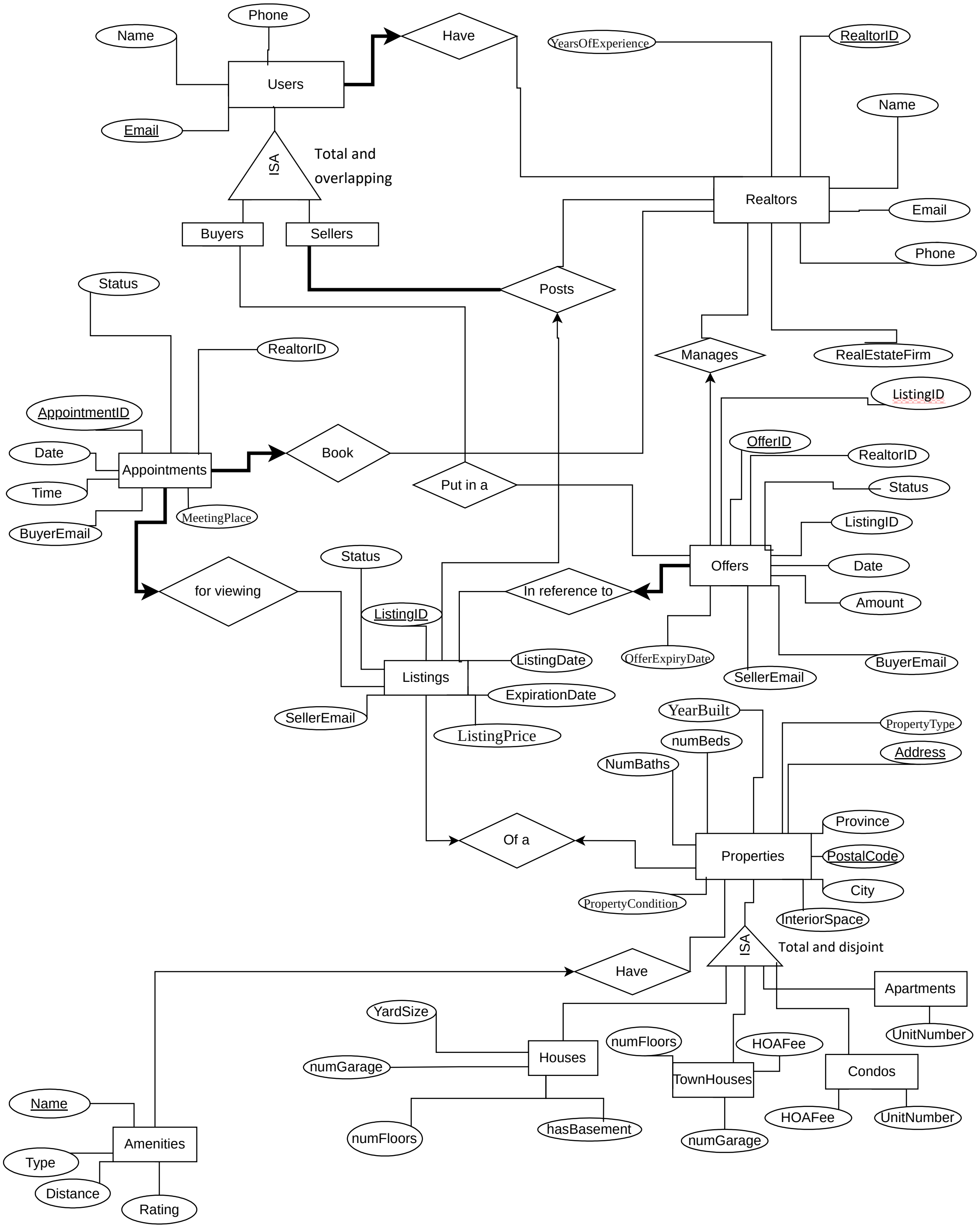
University of British Columbia, Vancouver
Department of Computer Science

Project Summary

The ER diagram outlines a real estate management system, with entities such as Users, Realtors, Listings, and Properties. Users can be buyers or sellers, while Realtors manage Listings and Offers, with Properties categorized into Houses, Apartments, Townhouses, and Condos, each having distinct attributes.

ER Diagram

We made several updates to the ER diagram for clarity and improved functionality. First, in the Users entity, we removed the `role` attribute, changed the primary key to `email`, and removed the `UserID` attribute as per our TA's feedback from M1. Additionally, we added a note beside the `ISA` relationships indicating it is "Total and overlapping". In the Properties entity, we removed the `PropertyID` attribute, opting to use `Address` and `Postal Code` as the composite key, and removed arrows pointing to `ISA` and `PropertyCondition` (these were errors pointed out by our TA). We added a note that the `ISA` relationship is "Total and disjoint." For the Appointments entity, we removed the arrow pointing to `MeetingPlace`. In the Offers entity, we added `ListingID` as an attribute to connect offers to listings, providing better traceability. Finally, we replaced all instances of `BuyerID` and `SellerID` with `BuyerEmail` and `SellerEmail` to maintain consistency with the updated Users entity. These changes streamline the diagram and ensure all entities are accurately represented.


Phone
Name
Have
YearsOfExperience
RealtorID
Users
Email
Name
ISA
Total and
overlapping
Realtors
Email
Buyers
Sellers
Phone
Status
Posts
RealtorID
Manages
RealEstateFirm
ListingID
AppointmentID
OfferID
Date
Appointments
Book
RealtorID
Put in a
Status
Time
MeetingPlace
ListingID
BuyerEmail
Status
Offers
Date
for viewing
In reference to
Amount
ListingID
OfferExpiryDate
BuyerEmail
SellerEmail
Listings
ListingDate
ExpirationDate
SellerEmail
ListingPrice
YearBuilt
PropertyType
numBeds
Address
NumBaths
Province
Of a
Properties
PostalCode
City
PropertyCondition
InteriorSpace
ISA
Total and disjoint
Have
Apartments
YardSize
UnitNumber
numFloors
HOAFee
numGarage
Houses
TownHouses
Condos
Name
HOAFee
UnitNumber
numFloors
hasBasement
numGarage
Amenities
Type
Distance
Rating

Schema

- Users(Name: VARCHAR, Email: VARCHAR, Phone: INTEGER)
    - PK is Email
    - Phone is UNIQUE and NOT NULL, Name NOT NULL
    - Email is NOT NULL
- Realtors(Name: VARCHAR, Email: VARCHAR, Phone: INTEGER, RealtorID: INTEGER, YearsOfExperience: INTEGER, RealEstateFirm: VARCHAR)
    - PK is RealtorID
    - CKs are Email and Phone
    - Email, Phone is UNIQUE and NOT NULL, RealEstateFirm NOT NULL, RealtorID NOT NULL
- Appointments(AppointmentID: INTEGER, Status: CHAR[9], **RealtorID**: INTEGER, Date: CHAR[10], Time: CHAR[5], **BuyerEMail**: VARCHAR, MeetingPlace: VARCHAR)
    - PK is AppointmentID
    - CK is RealtorID, Date, Time, BuyerEMail
    - FKs are RealtorID (references Realtors(RealtorID)) and BuyerEMail (references Users(BuyerEmail))
    - AppointmentID, RealtorID, Date, Time, BuyerEmail, Status, MeetingPlace must all be NOT NULL
    - (RealtorID, Date, Time, BuyerEmail) must be UNIQUE

- Offers(OfferID: INTEGER, **RealtorID**: INTEGER, Status: CHAR[8], Date: CHAR[10], Amount: INTEGER, **BuyerEmail**: VARCHAR, **SellerEmail**: VARCHAR, OfferExpiryDate: CHAR[10], **ListingID**: INTEGER)
    - PK is OfferID
    - CK is RealtorID, Date, Amount, SellerEmail, BuyerEmail
    - FKs are RealtorID (references Realtors(RealtorID)), BuyerEmail/SellerEmail (references Users(BuyerEmail/SellerEmail)), and ListingID (reference
    - OfferID, RealtorID, Status, Date, Amount, SellerEmail, BuyerEmail must all be NOT NULL
    - (RealtorID, Date, Amount, SellerEmail, BuyerEmail) must be UNIQUE
- Listings(ListingID: Integer, Status: CHAR[9], **SellerEmail**: VARCHAR, ListingPrice: Integer, ExpirationDate: CHAR[10], ListingDate: CHAR[10])
    - PK is ListingID
    - CK is SellerEmail, ListingPrice, ListingDate
    - FK is SellerEmail (references Users(SellerEmail))
    - ListingID, Status, SellerEmail, ListingPrice must all be NOT NULL
    - (SellerEmail, ListingPrice, ListingDate) must be UNIQUE

- Properties(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PropertyType: CHAR[10], PostalCode: CHAR[6], PropertyCondition: CHAR[6], NumBaths: INTEGER, NumBeds: INTEGER, YearBuilt: INTEGER, InteriorSpace: INTEGER)
  - PK is Address, PostalCode
  - CK is Address, City, Province
  - Address, City, Province, PropertyType, PostalCode, NumBaths, NumBeds, InteriorSpace must all be NOT NULL
  - (Address, City, Province) must be UNIQUE
- Houses(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6], YardSize: INTEGER, numGarage: INTEGER, numFloors: INTEGER, hasBasement: CHAR[3])
  - PK and FK is Address, PostalCode
  - CK is Address, City, Province
  - Address, City, Province, PostalCode, YardSize, numFloors, hasBasement must all be NOT NULL
  - (Address, City, Province) must be UNIQUE
- TownHouses(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6], numGarage: INTEGER, numFloors: INTEGER, HOAFee: INTEGER)
  - PK and FK is Address, PostalCode
  - CK is Address, City, Province
  - Address, City, Province, PostalCode, numFloors, HOAFee must all be NOT NULL
  - (Address, City, Province) must be UNIQUE
- Condos(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6], HOAFee: INTEGER, UnitNumber: INTEGER)
  - PK and FK is Address and PostalCode, UnitNumber is PK
  - CK is Address, City, Province
  - Address, City, Province, PostalCode, HOAFee, UnitNumber must all be NOT NULL
  - (Address, City, Province) must be UNIQUE
- Apartments(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6], UnitNumber: INTEGER)
  - PK and FK is Address and PostalCode, UnitNumber is PK
  - CK is Address, City, Province, UnitNumber
  - Address, City, Province, PostalCode, UnitNumber must all be NOT NULL
  - (Address, City, Province) must be UNIQUE
- Amenities(Name: VARCHAR, Type: CHAR[20], Distance: INTEGER, Rating: INTEGER)
  - PK is Name

Functional Dependencies

USERS

Let: UName = User name
UEmail = User email
UPhone = User phone

Then: UEmail -> Uname, UPhone
UPhone -> UName

REALTORS

Let: RName = Realtor name
REmail = Realtor email
RPhone = Realtor phone
RID = RealtorID
YOE = YearsOfExperience
Firm = RealEstateFirm

Then: RealtorID -> RName, REmail, RPhone, YOE, Firm
REmail, RPhone -> RName, YOE, Firm, RealtorID
RName, Firm -> REmail

APPOINTMENT

Let: AID = AppointmentID
AStat = Appointment status
RID = RealtorID
ADate = Appointment date
ATime = Appointment time
BEmail = Buyer email
Place = Appointment meeting place

Then: AID -> AStat, RID, ADate, ATime, BEmail, Place
RID, ADate, ATime, BEmail -> AID, AStat, Place
BEmail, ADate, RID, Place -> ATime, AID
Place, ADate, ATime -> RID, BEmail

OFFERS

Let: OID = OfferID
RID = RealtorID
OStat = Offer status
ODate = Offer date
OAmt = Offer amount
BEmail = Buyer email
SEmail = Seller email
OExp = Offer expiry date

Then: OID -> RID, OStat, ODate, OAmt, BEmail, SEmail, OExp
RID, ODate, OAmt, SEmail, BEmail -> OID, OStat, OExp
BEmail-> OAmt, ODate, RID, OExp

LISTINGS

Let: LID = ListingID
LStat = Listing status
SEmail = Seller email
Price = Listing price
LExp = Listing expiry date
LDate = Listing date

Then: LID -> LStat, SEmail, Price, LExp, LDate
SEmail, Price, LDate ->LID, LStat, LExp
LDate, LExp -> LStat

PROPERTIES

Let: Addr = Property address
City = Property city
Prov = Property province
Type = Property type
Code = Property postal code
Cond = Property condition
Bath = number of bathrooms
Bed = number of bedrooms
YB = Property year built
IS = Interior space

Then: Addr, Code -> City, Prov, Type, Cond, Bath, Bed, YB, IS
Addr, City, Prov -> Type, Code, Cond, Bath, Bed, YB, IS
Code -> City, Prov

HOUSES

Let: Addr = House address
City = House city
Prov = House province
Code = House postal code
YS = Yard size
NG = Number of garages
NF = Number of floors
HB = has basement

Then: Addr, Code -> City, Prov, YS, NG, NF, HB
Addr, City, Prov -> Code, YS, NG, NF, HB

TOWNHOUSES

Let: Addr = Town house address
City = Town house city
Prov = Town house province
Code = Town house postal code
NF = Number of floors
HF = HOA Fees

Then: Addr, Code -> City, Prov, NF, HF
Addr, City, Prov -> Code, NF, HF

APARTMENTS

Let: Addr = apartment address
City = apartment city
Prov = apartment province
Code = apartment postal code
UN = Unit number of apartment

Then: Addr, Code, UN -> City, Prov
Addr, Code -> City, Prov

CONDOS

Let: Addr = condos address
City = condos city
Prov = condos province
Code = condos postal code
HF = HOA fee condos
UN = condo unit number

Then: Addr, Code, UN -> HF, City, Prov
Addr, City, Prov -> Code, HF

AMENITIES

Let: AName = Amenity name
AType = Amenity type
ADist = Amenity distance
ARate = Amenity rating

Then: AName -> AType, ADist, ARate
ARate -> ADist

BCNF Normalization

Original Table

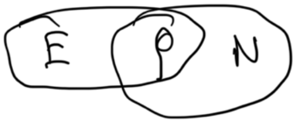
- Users(Name: VARCHAR, Email: VARCHAR, Phone: INTEGER)

Normalized Tables:

- Users1(Name: VARCHAR, Phone: INTEGER)
    - PK is Phone
    - Name NOT NULL
- Users2(Phone: INTEGER, Email: VARCHAR)
    - PK is Email
    - Phone is UNIQUE and NOT NULL
    - Email is NOT NULL

Users(Name, Email, Phone)

phone -> Name:



U1(P, N) U2(P, E)

Original Table:

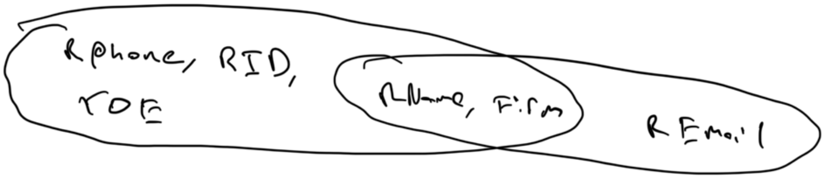
- Realtors(Name: VARCHAR, Email: VARCHAR, Phone: INTEGER, RealtorID: INTEGER, YearsOfExperience: INTEGER, RealEstateFirm: VARCHAR)

Normalized Tables:

- Realtors2(Name: VARCHAR, Phone: INTEGER, RealtorID: INTEGER, YearsOfExperience: INTEGER, RealEstateFirm: VARCHAR)
- Realtors2(Name: VARCHAR, RealEstateFirm: VARCHAR, Email: VARCHAR)
  - PK is RealtorID
  - CKs are Email and Phone
  - Email, Phone is UNIQUE and NOT NULL, RealEstateFirm NOT NULL, RealtorID NOT NULL

Realtors(RName, REmail, Rphone, RID, YOE
Firm)

RName, Firm -> REmail ;



Realtors1(RName, Firm, REmail)
Realtors2(RName, Firm, Rphone, RID, YOE)

Normalized and Original Table:

- Appointments(AppointmentID: INTEGER, Status: CHAR[9], **RealtorID**: INTEGER, Date: CHAR[10], Time: CHAR[5], **BuyerEMail**: VARCHAR, MeetingPlace: VARCHAR)

- PK is <u>AppointmentID</u>
- CK is RealtorID, Date, Time, BuyerEMail
- FKs are RealtorID (references Realtors(RealtorID)) and BuyerEMail (references Users(BuyerEmail))
- AppointmentID, RealtorID, Date, Time, BuyerEmail, Status, MeetingPlace must all be NOT NULL
- (RealtorID, Date, Time, BuyerEmail) must be UNIQUE

Original Table:

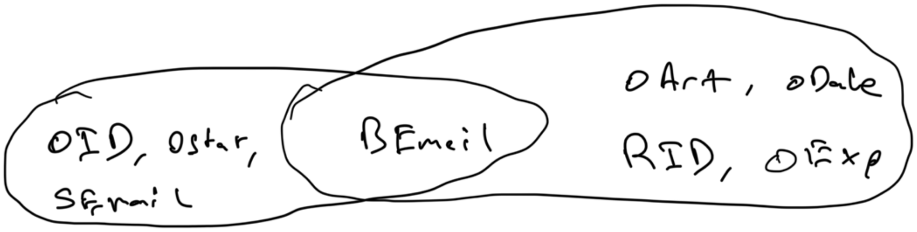
- Offers(<u>OfferID</u>: INTEGER, **RealtorID**: INTEGER, Status: CHAR[8], Date: CHAR[10], Amount: INTEGER, **BuyerEmail**: VARCHAR, **SellerEmail**: VARCHAR, OfferExpiryDate: CHAR[10], **ListingID**: INTEGER)

Normalized Tables:

- Offers1(**RealtorID**: INTEGER, Date: CHAR[10], Amount: INTEGER, <u>**BuyerEmail: VARCHAR**</u>, OfferExpiryDate: CHAR[10], **ListingID**: INTEGER)
- Offers2<u>(OfferID</u>: INTEGER, Status: CHAR[8], **BuyerEmail: VARCHAR**, **SellerEmail**: VARCHAR)
  - PK is <u>OfferID</u>
  - CK is RealtorID, Date, Amount, SellerEmail, BuyerEmail
  - FKs are RealtorID (references Realtors(RealtorID)), BuyerEmail/SellerEmail (references Users(BuyerEmail/SellerEmail)), and ListingID (reference
  - OfferID, RealtorID, Status, Date, Amount, SellerEmail, BuyerEmail must all be NOT NULL
  - (RealtorID, Date, Amount, SellerEmail, BuyerEmail) must be UNIQUE

Offers (OfferID, RealtorID, Status,
Date, Amount, BuyerEmail, SellerEmail,
OfferExpiryDate, ListingID)

BEmail -> OAmt, ODate, RID, OExp;



Offer1 (BEmail, OAmt, ODate, RID, OExp)
Offer2 (BEmail, OID, Ostat, SEmail)

Original Table:

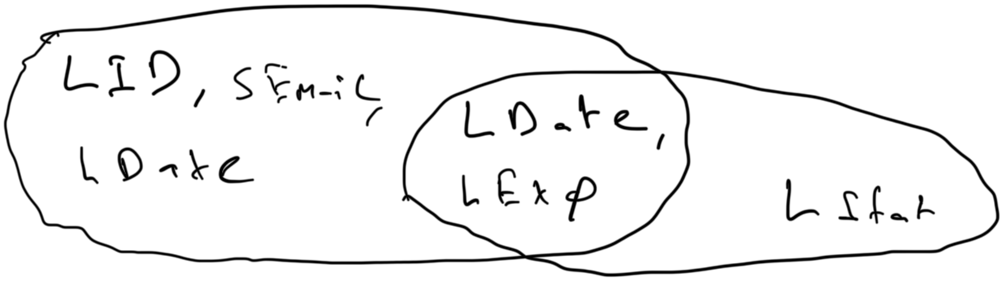
- Listings(ListingID: Integer, Status: CHAR[9], **SellerEmail**: VARCHAR, ListingPrice: Integer, ExpirationDate: CHAR[10], ListingDate: CHAR[10])

Normalized Tables:

- Listings1(ListingID: Integer, **SellerEmail**: VARCHAR, **ListingPrice**: Integer, **ListingDate**: CHAR[10])
- Listings2(**ListingDate**: CHAR[10], **ExpirationDate**: CHAR[10], Status: CHAR[9])
    - PK is ListingID
    - CK is SellerEmail, ListingPrice, ListingDate, ListingExp
    - FKs are SellerEmail (references Users(SellerEmail)), ListingDate, ListingExp
    - ListingID, Status, SellerEmail, ListingPrice must all be NOT NULL
    - (SellerEmail, ListingPrice, ListingDate) must be UNIQUE

Listings (LID, LStat, SEmail, @Price, LExp, LDate)

LDate, LExp -> LStat :



Listings1 (LDate, LExp, LStat)
Listings2 (LDate, LID, LExp, SEmail, LDate)

Original Table

- Properties(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PropertyType: CHAR[10], PostalCode: CHAR[6], PropertyCondition: CHAR[6], NumBaths: INTEGER, NumBeds: INTEGER, YearBuilt: INTEGER, InteriorSpace: INTEGER)

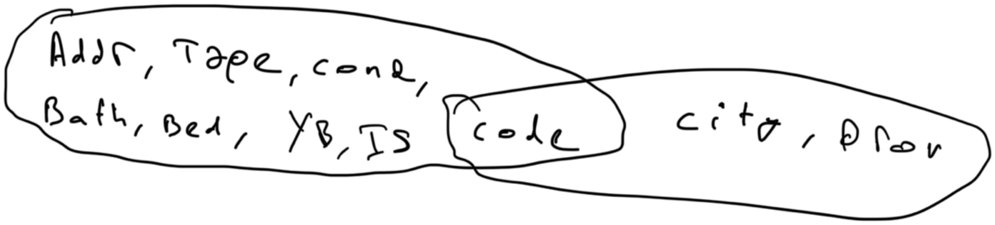
Normalized Tables:

- Properties1(PostalCode: CHAR[6], City: VARCHAR, Province: VARCHAR)
- Properties2(Address: VARCHAR, PropertyType: CHAR[10], PostalCode: CHAR[6], PropertyCondition: CHAR[6], NumBaths: INTEGER, NumBeds: INTEGER, YearBuilt: INTEGER, InteriorSpace: INTEGER)
    - PK is Address, PostalCode
    - CK is Address, City, Province

- Address, City, Province, PropertyType, PostalCode, NumBaths, NumBeds, InteriorSpace must all be NOT NULL
- (Address, City, Province) must be UNIQUE

Properties (Addr, city, prov, type, code, cond, Bath, Bed, YB, IS)

code -> city, prov:



properties1 (<u>Code</u>, city, prov)
properties2 (<u>Addr</u>, Tale, cond, Bath, <u>Code</u>, Bed, YB, IS)

Original and Normalized Table:

- Houses(<u>Address</u>: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6], YardSize: INTEGER, numGarage: INTEGER, numFloors: INTEGER, hasBasement: CHAR[3])
  - PK and FK is <u>Address</u>, <u>PostalCode</u>
  - CK is Address, City, Province
  - Address, City, Province, PostalCode, YardSize, numFloors, hasBasement must all be NOT NULL
  - (Address, City, Province) must be UNIQUE

Original and Normalized Table:

- TownHouses(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6], numGarage: INTEGER, numFloors: INTEGER, HOAFee: INTEGER)
  - PK and FK is Address, PostalCode
  - CK is Address, City, Province
  - Address, City, Province, PostalCode, numFloors, HOAFee must all be NOT NULL
  - (Address, City, Province) must be UNIQUE

Original Table:

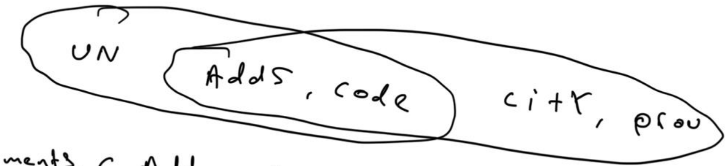
- Apartments(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6], UnitNumber: INTEGER)

Normalized Tables:

- Apartments1(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6])
- Apartments2(Address: VARCHAR, PostalCode: CHAR[6]), UnitNumber: INTEGER)
  - PK and FK is Address and PostalCode, UnitNumber is PK
  - CK is Address, City, Province, UnitNumber
  - Address, City, Province, PostalCode, UnitNumber must all be NOT NULL
  - (Address, City, Province) must be UNIQUE

Apartments(Addr, City, Prov, code, UN)

Addr, code -> city, prov ;



Apartments1(Addr, code, city, prov)

Apartments2(Addr, code, UN)

Original Table:

- Condos(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6], HOAFee: INTEGER, UnitNumber: INTEGER)

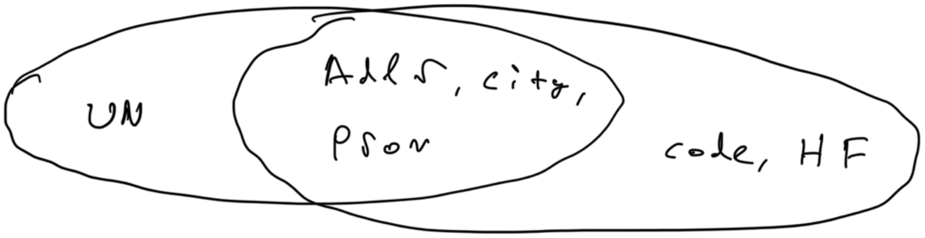
Normalized Tables:

- Condos1(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, PostalCode: CHAR[6], HOAFee: INTEGER)

- Condos2(Address: VARCHAR, City: VARCHAR, Province: VARCHAR, UnitNumber: INTEGER)
    - PK and FK is Address and PostalCode, UnitNumber is PK
    - CK is Address, City, Province
    - Address, City, Province, PostalCode, HOAFee, UnitNumber must all be NOT NULL
    - (Address, City, Province) must be UNIQUE

condos(Addr, City, Prov, code, HF, UN)

Addr, city, Prov -> Code, HF;



condos1(Addr, city, Prov, code HF)

condos2(Addr, city, Prov, UN)

Original Table:

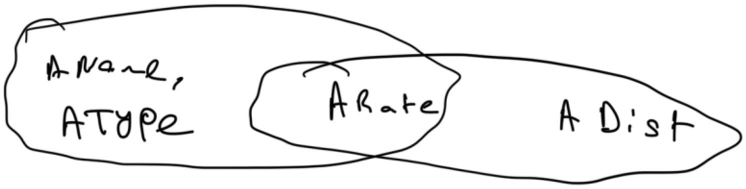
- Amenities(Name: VARCHAR, Type: CHAR[20], Distance: INTEGER, Rating: INTEGER)

Normalized Tables:

- Amenities1(Distance: INTEGER, Rating: INTEGER)
- Amenities2(Rating: INTEGER, Type: CHAR[20], Name: VARCHAR)
    - PK is Name
    - CKs are Rating, Distance

Amenities (Aname, AType, Adist, ARate)

ARate -> ADist:



Amenities1 (ARate, ADist)
Amenities2 (ARate, AName, AType)

SQL DDL

```
-- Users Table
CREATE TABLE Users (
   Name VARCHAR(255) NOT NULL,
   Email VARCHAR(255) NOT NULL,
   Phone INTEGER NOT NULL,
   PRIMARY KEY (Email),
   UNIQUE (Phone)
);

-- Realtors Table
CREATE TABLE Realtors (
   RealtorID INTEGER NOT NULL,
   Name VARCHAR(255) NOT NULL,
   Email VARCHAR(255) NOT NULL,
   Phone INTEGER NOT NULL,
   YearsOfExperience INTEGER,
   RealEstateFirm VARCHAR(255) NOT NULL,
   PRIMARY KEY (RealtorID),
   UNIQUE (Email),
   UNIQUE (Phone)
);

-- Appointments Table
CREATE TABLE Appointments (
   AppointmentID INTEGER NOT NULL,
   Status CHAR(9) NOT NULL,
   RealtorID INTEGER NOT NULL,
   Date CHAR(10) NOT NULL,
   Time CHAR(5) NOT NULL,
   BuyerEmail VARCHAR(255) NOT NULL,
   MeetingPlace VARCHAR(255) NOT NULL,
   PRIMARY KEY (AppointmentID),
   UNIQUE (RealtorID, Date, Time, BuyerEmail),
   FOREIGN KEY (RealtorID) REFERENCES Realtors(RealtorID),
   FOREIGN KEY (BuyerEmail) REFERENCES Users(Email)
);

-- Offers Table
CREATE TABLE Offers (
   OfferID INTEGER NOT NULL,
   RealtorID INTEGER NOT NULL,
   Status CHAR(8) NOT NULL,
   Date CHAR(10) NOT NULL,
   Amount INTEGER NOT NULL,
   BuyerEmail VARCHAR(255) NOT NULL,
   SellerEmail VARCHAR(255) NOT NULL,
   OfferExpiryDate CHAR(10) NOT NULL,
   ListingID INTEGER NOT NULL,
   PRIMARY KEY (OfferID),
   UNIQUE (RealtorID, Date, Amount, SellerEmail, BuyerEmail),
   FOREIGN KEY (RealtorID) REFERENCES Realtors(RealtorID),
   FOREIGN KEY (BuyerEmail, SellerEmail) REFERENCES Users(Email),
   FOREIGN KEY (ListingID) REFERENCES Listings(ListingID)
);

-- Listings Table
CREATE TABLE Listings (
   ListingID INTEGER NOT NULL,
   Status CHAR(9) NOT NULL,
   SellerEmail VARCHAR(255) NOT NULL,
   ListingPrice INTEGER NOT NULL,
   ExpirationDate CHAR(10),
   ListingDate CHAR(10) NOT NULL,
   PRIMARY KEY (ListingID),
   UNIQUE (SellerEmail, ListingPrice, ListingDate),
   FOREIGN KEY (SellerEmail) REFERENCES Users(Email)
);

-- Properties Table
CREATE TABLE Properties (
   Address VARCHAR(255) NOT NULL,
   City VARCHAR(255) NOT NULL,
   Province VARCHAR(255) NOT NULL,
   PropertyType VARCHAR(10) NOT NULL,
   PostalCode CHAR(6) NOT NULL,
   PropertyCondition VARCHAR(6),
   NumBaths INTEGER NOT NULL,
   NumBeds INTEGER NOT NULL,
   YearBuilt INTEGER,
   InteriorSpace INTEGER NOT NULL,
   PRIMARY KEY (Address, PostalCode),
   UNIQUE (Address, City, Province)
);

-- Houses Table
CREATE TABLE Houses (
   Address VARCHAR(255) NOT NULL,
   City VARCHAR(255) NOT NULL,
   Province VARCHAR(255) NOT NULL,
   PostalCode CHAR(6) NOT NULL,
   YardSize INTEGER NOT NULL,
   NumGarage INTEGER,
   NumFloors INTEGER NOT NULL,
   HasBasement CHAR(3) NOT NULL,
   PRIMARY KEY (Address, PostalCode),
   FOREIGN KEY (Address, PostalCode) REFERENCES Properties(Address, PostalCode),
   UNIQUE (Address, City, Province)
);

-- TownHouses Table
CREATE TABLE TownHouses (
   Address VARCHAR(255) NOT NULL,
   City VARCHAR(255) NOT NULL,
   Province VARCHAR(255) NOT NULL,
   PostalCode CHAR(6) NOT NULL,
   NumGarage INTEGER,
   NumFloors INTEGER NOT NULL,
   HOAFee INTEGER NOT NULL,
   PRIMARY KEY (Address, PostalCode),
   FOREIGN KEY (Address, PostalCode) REFERENCES Properties(Address, PostalCode),
   UNIQUE (Address, City, Province)
);

-- Condos Table
CREATE TABLE Condos (
   Address VARCHAR(255) NOT NULL,
   City VARCHAR(255) NOT NULL,
   Province VARCHAR(255) NOT NULL,
   PostalCode CHAR(6) NOT NULL,
   HOAFee INTEGER NOT NULL,
   UnitNumber INTEGER NOT NULL,
   PRIMARY KEY (Address, PostalCode, UnitNumber),
   FOREIGN KEY (Address, PostalCode) REFERENCES Properties(Address, PostalCode),
   UNIQUE (Address, City, Province)
);

-- Apartments Table
CREATE TABLE Apartments (
   Address VARCHAR(255) NOT NULL,
   City VARCHAR(255) NOT NULL,
   Province VARCHAR(255) NOT NULL,
   PostalCode CHAR(6) NOT NULL,
   UnitNumber INTEGER NOT NULL,
   PRIMARY KEY (Address, PostalCode, UnitNumber),
   FOREIGN KEY (Address, PostalCode) REFERENCES Properties(Address, PostalCode),
   UNIQUE (Address, City, Province, UnitNumber)
);

-- Amenities Table
CREATE TABLE Amenities (
   Name VARCHAR(255) NOT NULL,
   Type VARCHAR(20),
   Distance INTEGER,
   Rating INTEGER,
   PRIMARY KEY (Name)
);
```

Populated Tables

1.Users

```
INSERT INTO Users (Name, Email, Phone)
VALUES
('Sam Jane', 'samjane@hotmail.com', '123-456-7890'),
('George Washington', 'georgewash@laundry.com', '223-456-7990'),
('Will Smith', 'willsmith@alaadin.com', '123-786-7893'),
('Sundar Pichai', 'sundarpichai@gmail.com', '235-455-7899),
('Brett Lee', 'brettlee@rediffmail.com', '515-444-7890'),
('Steve Smith', 'stevesmith@google.com', '234-567-8901');
```

2.Realtors

```
INSERT INTO Realtors (RealtorID, Name, Email, Phone, YearsOfExperience,
RealEstateFirm)
VALUES
(1, 'Emily Brown', 'emilybrown@gmail.com', '345-678-9012', 5, 'Dream Homes
Realty'),
(2, 'Michael Johnson', 'michaeljohnson@gmail.com', '456-789-0123', 8, 'Premium
Estates'),
(3, 'Sarah Green', 'sarahgreen@gmail.com', '567-890-1234', 4, 'Top Choice Realty'),
(4, 'David Wilson', 'davidwilson@gmail.com', '678-901-2345', 10, 'Elite Realty
Group'),
(5, 'Jessica Taylor', 'jessicataylor@gmail.com', '789-012-3456', 7, 'Star Real Estate'),
(6, 'Daniel White', 'danielwhite@gmail.com', '890-123-4567', 6, 'Prime Properties');
```

3. Appointments

```
INSERT INTO Appointments (AppointmentID, Date, Time, MeetingPlace, BuyerEmail,
ListingID, AppointmentStatus)
VALUES
(1, '2024-03-05', '10:00', '123 Elm St', 'samjane@hotmail.com', 1, 'Scheduled'),
(2, '2024-04-05', '14:00', '456 Oak St', 'georgewash@laundry.com', 2, 'Completed'),
(3, '2024-05-10', '11:00', '789 Pine St', 'willsmith@alaadin.com', 3, 'Scheduled'),
(4, '2024-06-10', '15:00', '321 Maple St', 'sundarpichai@gmail.com', 4, 'Cancelled'),
(5, '2024-07-15', '09:00', '654 Birch St', ' brettlee@rediffmail.com', 5, 'Scheduled'),
(6, '2024-08-20', '13:00', '987 Cedar St', ' stevesmith@google.com', 6, 'Completed');
```

3.Listings

```
INSERT INTO Listings (ListingID, ListingDate, ExpirationDate, ListingPrice, YearBuilt,
NumBeds, NumBaths, ListingStatus, SellerEmail, PropertyType)
VALUES
(1, '2024-01-01', '2024-06-01', 500000, 2010, 4, 3, 'Active', 'samjane@hotmail.com',
'House'),
(2, '2024-02-01', '2024-07-01', 300000, 2015, 2, 1, 'Pending',
'georgewash@laundry.com', 'Condo'),
(3, '2024-03-01', '2024-08-01', 450000, 2012, 3, 2, 'Active', 'willsmith@alaadin.com',
'TownHouse'),
(4, '2024-04-01', '2024-09-01', 350000, 2018, 3, 2, 'Active',
'sundarpichai@gmail.com', 'Apartment'),
(5, '2024-05-01', '2024-10-01', 550000, 2020, 5, 4, 'Active',
'brettlee@rediffmail.com', 'House'),
(6, '2024-06-01', '2024-11-01', 400000, 2016, 3, 2, 'Pending',
'stevesmith@google.com', 'Condo');
```

4. Appointments

```
INSERT INTO Appointments (AppointmentID, Date, Time, MeetingPlace, BuyerEmail,
ListingID, AppointmentStatus)
VALUES
(1, '2024-03-05', '10:00', '123 Elm St', 'samjane@hotmail.com', 1, 'Scheduled'),
(2, '2024-04-05', '14:00', '456 Oak St', 'georgewash@laundry.com', 2, 'Completed'),
(3, '2024-05-10', '11:00', '789 Pine St', 'willsmith@alaadin.com', 3, 'Scheduled'),
(4, '2024-06-10', '15:00', '321 Maple St', 'sundarpichai@gmail.com', 4, 'Cancelled'),
(5, '2024-07-15', '09:00', '654 Birch St', 'brettlee@rediffmail.com', 5, 'Scheduled'),
(6, '2024-08-20', '13:00', '987 Cedar St', 'stevesmith@google.com', 6, 'Completed');
```

5.Offers

```
INSERT INTO Offers (OfferID, ListingID, OfferDate, OfferExpiryDate, OfferAmount,
OfferStatus, BuyerEmail, SellerEmail, RealtorID)
VALUES
(1, 1, '2024-03-01', '2024-03-15', 510000, 'Pending', 'samjane@hotmail.com',
'georgewash@laundry.com', 1),
(2, 2, '2024-04-01', '2024-04-10', 310000, 'Accepted', 'georgewash@laundry.com',
'samjane@hotmail.com', 2),
(3, 3, '2024-05-01', '2024-05-15', 460000, 'Pending', 'willsmith@alaadin.com',
'sundarpichai@gmail.com', 3),
(4, 4, '2024-06-01', '2024-06-15', 360000, 'Rejected', 'sundarpichai@gmail.com',
'brettlee@rediffmail.com', 4),
(5, 5, '2024-07-01', '2024-07-10', 560000, 'Pending', 'brettlee@rediffmail.com',
'stevesmith@google.com', 5),
(6, 6, '2024-08-01', '2024-08-10', 410000, 'Accepted', 'stevesmith@google.com',
'willsmith@alaadin.com', 6);
```

6.Properties

```
INSERT INTO Properties (PropertyID, Address, City, Province, PostalCode,
InteriorSpace, PropertyCondition)
VALUES
(1, '123 Elm St', 'Vancouver', 'BC', 'V5K 1A1', 2000, 'Good'),
(2, '456 Oak St', 'Burnaby', 'BC', 'V5H 2B2', 900, 'Excellent'),
(3, '789 Pine St', 'Richmond', 'BC', 'V6X 3A4', 1500, 'Good'),
(4, '321 Maple St', 'Surrey', 'BC', 'V3T 4B5', 1200, 'Fair'),
(5, '654 Birch St', 'Coquitlam', 'BC', 'V3J 6B7', 1800, 'Good'),
(6, '987 Cedar St', 'Langley', 'BC', 'V1M 2S3', 1300, 'Excellent');
```

7.Houses

```
INSERT INTO Houses (PropertyID, YardSize, NumGarage, NumFloors, HasBasement)
VALUES
(1, 500, 2, 2, TRUE),
(2, 600, 1, 2, FALSE),
(3, 550, 2, 1, TRUE),
(4, 700, 3, 2, TRUE),
(5, 650, 2, 3, FALSE),
(6, 800, 2, 2, TRUE);
```

8. Townhouses

```
INSERT INTO TownHouses (PropertyID, NumFloors, NumGarage, HOA Fee)
VALUES
(1, 2, 1, 150),
(2, 3, 2, 200),
(3, 2, 1, 175),
(4, 2, 2, 180),
(5, 3, 1, 160),
(6, 2, 1, 190);
```

9. Amenities

```
INSERT INTO Amenities (Name, Type, Distance, Rating)
VALUES
('Central Park', 'Park', 0.5, 4.5),
('Vancouver Library', 'Library', 1.0, 4.8),
('Community Pool', 'Pool', 0.8, 4.2),
('Fitness Center', 'Gym', 1.5, 4.7),
('Shopping Mall', 'Mall', 2.0, 4.3),
('City Theater', 'Theater', 1.2, 4.6);
```