**PM3\_Pioneers OCT\_25\_2018**

Created by: Qian Wang, Nan Liu, Shufan Xing, Wen Zhang

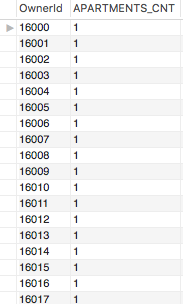
**Part1: 10 related query questions:**

**#1. How many apartments are owned by a given landlord?**

**Code:**



**Result(partial):**

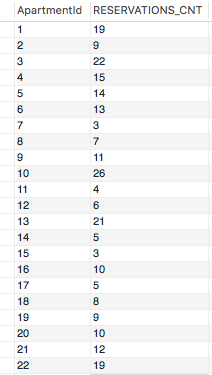


**#2. How many reservations are made on a given apartment?**

**Code:**



**Result(Partial):**

****

**#3. What are the available apartments near given University (University Name = 'University of Alabama at Birmingham')?**

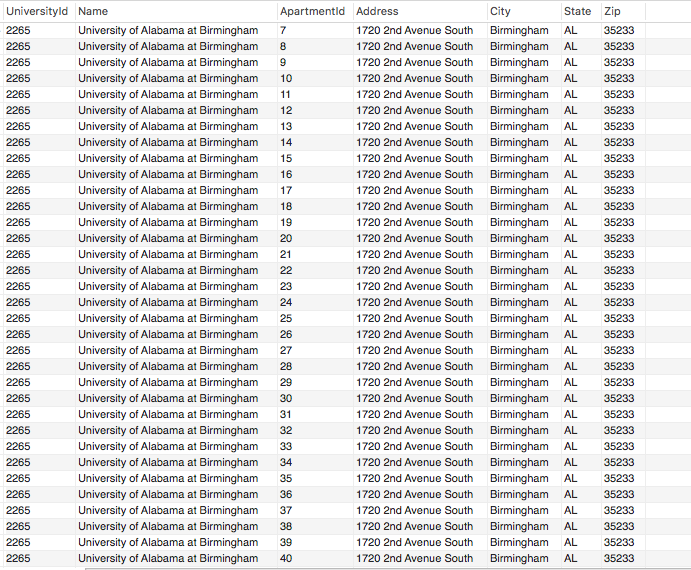
**Search by University Name and return apartments who has same zip code.**

**List out apartment ID and address.**

**Code:**



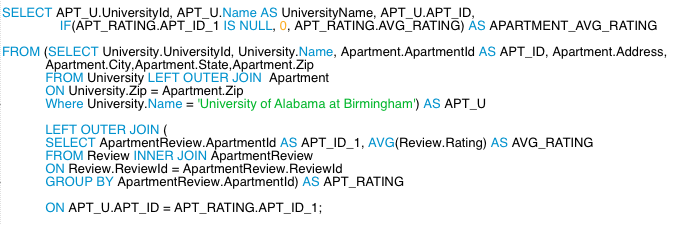
**Result(partial):**

****

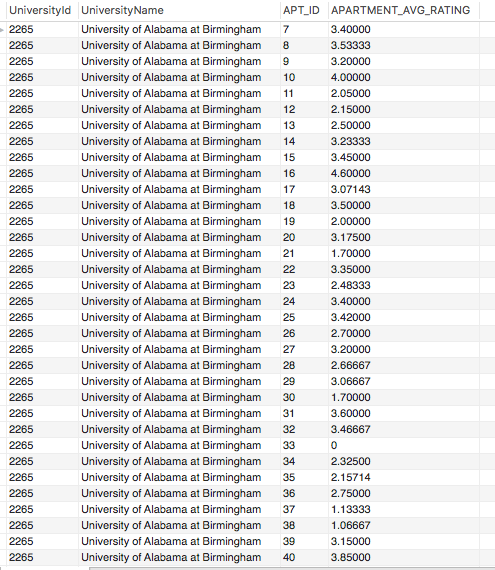
**#4. What are the average review rating for each available apartment around one certain University (UniversityName = 'University of Alabama at Birmingham')?**

**Code:**

*For read*

****

**Results(Partial):**

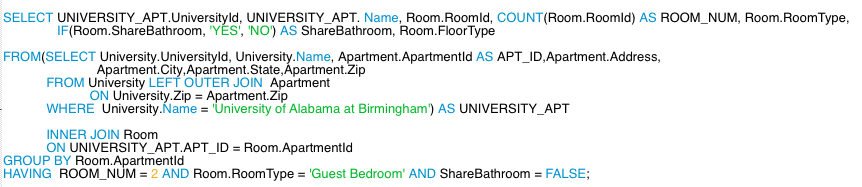


**#5. What are the available rooms that satisfied user’s specific requirement?**

**Given a university name (University of Alabama at Birmingham) find all available apartments nearby and then filter available rooms by room number = 2, room type = GuestBedroom, ShareBathroom = NO**

**Code:**

*For read:*

****

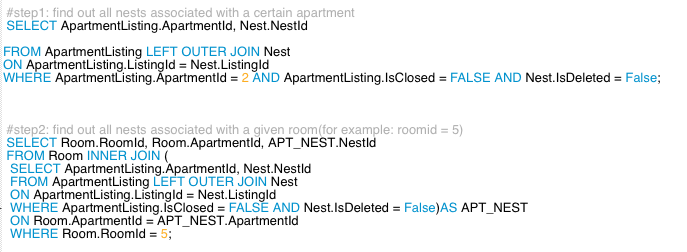
**Result:**



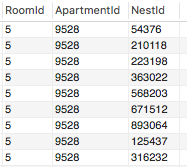
**#6. How many nests are associated with a given room（room id = 5）? List all the nest ID.**

**Code:**

*For read:*

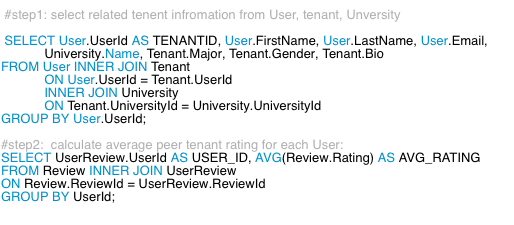
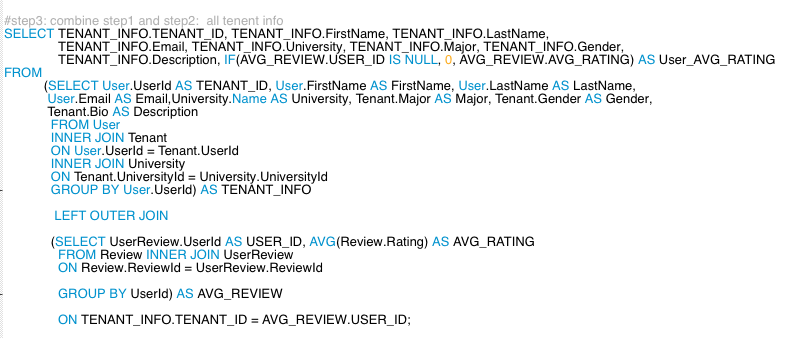
****

**Result:**

****

**#7. List information of all the Tenants in a certain nest (eg: nestId = 20), like first name, last name, university name, major, gender, Bio, average peer tenant review rating.**

**Code:**

****



**Result:**

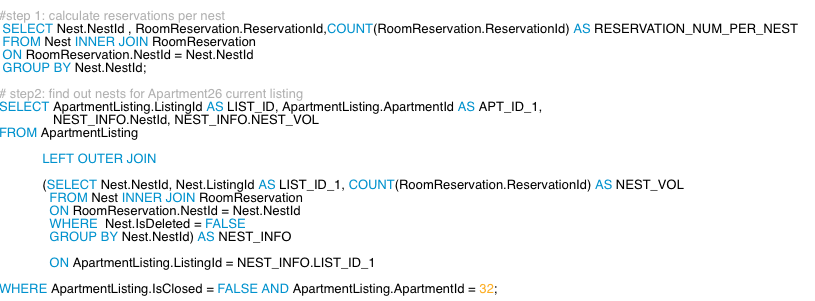


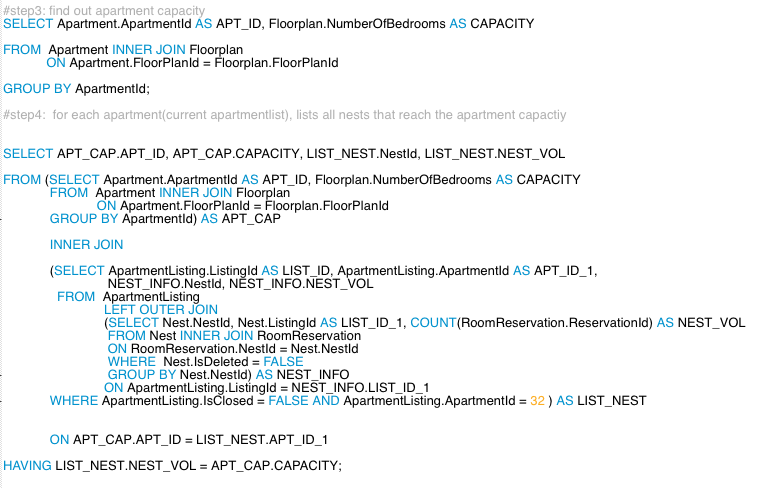
**#8. How many nests have reached the Apartment capacity (for Apartment Id = 26)?**

**for certain apartment (current listing), lists all nests that reach the apartment**

**capacity.**

**Code:**

****

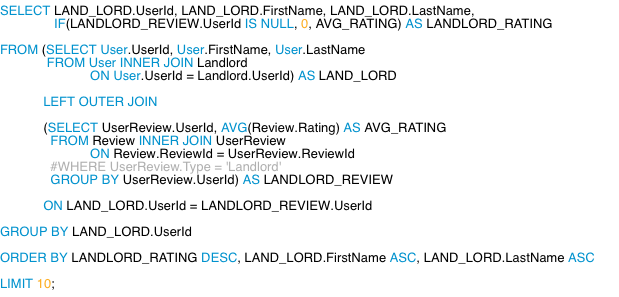
****

**Result:**

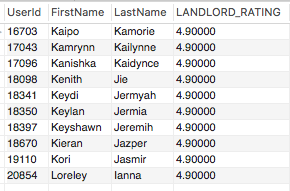
****

**#9. Who are the top 10 highest rating land lord this year?**

**Code:**

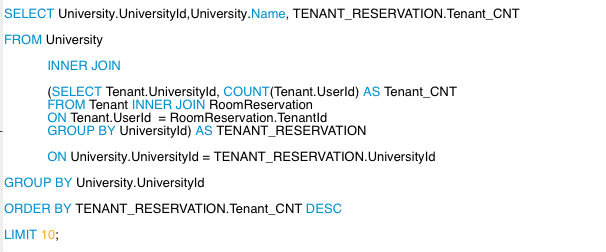
****

**Result:**

****

**#10. What are the top 10 universities that has maximum housing demand this year?**

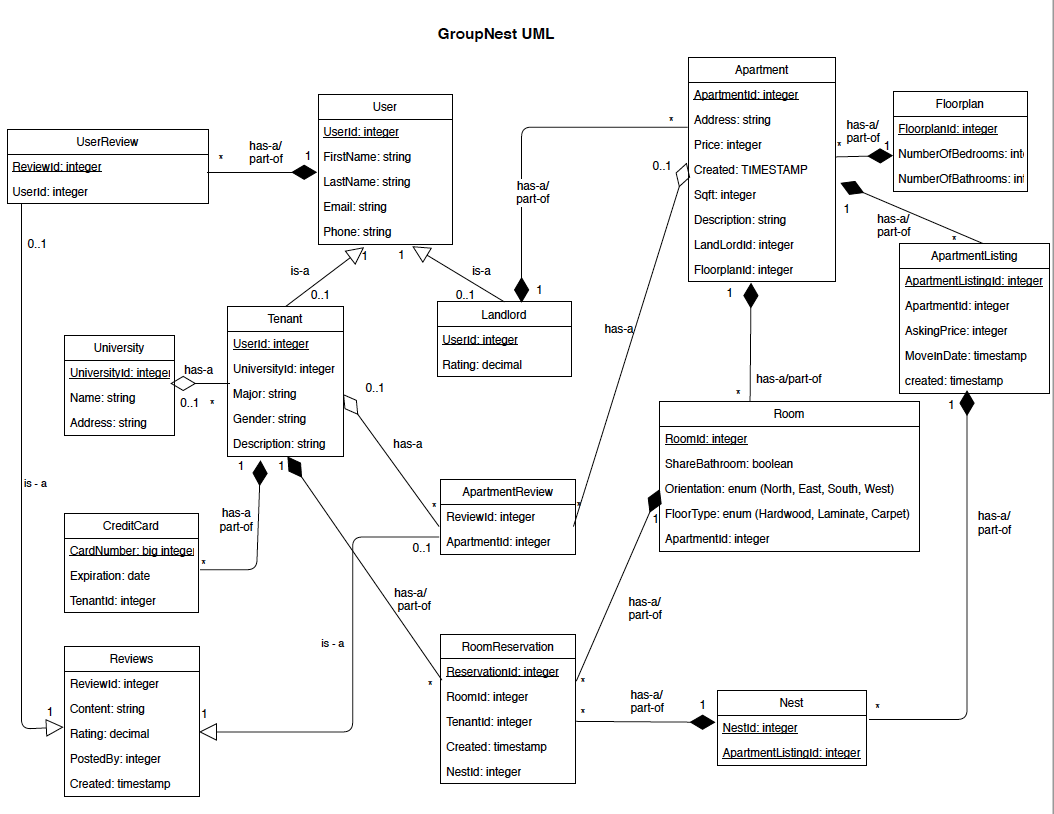
**Code:**

****

**Result:**

****

**Part2: Update UML**

****

**Part3: Update table:**

For Milestone 3, we made some modification on tables as following:

1. normalized all tables that violated 1NF, 2NF and 3NF, to make sure there is no duplicates of information in multiple tables.

2. abstracted review with common information from userReview and ApartmentReview classes. maintain the generalization relationship between review and two subclasses.

3. modified constraint in tables to make the relationship much more reasonable.

# Create the schema if necessary.

CREATE SCHEMA IF NOT EXISTS GruopNest;

USE GruopNest;

# Drop tables if necessary.

DROP TABLE IF EXISTS CreditCard;

DROP TABLE IF EXISTS UserReview;

DROP TABLE IF EXISTS ApartmentReview;

DROP TABLE IF EXISTS Review;

DROP TABLE IF EXISTS RoomReservation;

DROP TABLE IF EXISTS Nest;

DROP TABLE IF EXISTS ApartmentListing;

DROP TABLE IF EXISTS Room;

DROP TABLE IF EXISTS Apartment;

DROP TABLE IF EXISTS FloorPlan;

DROP TABLE IF EXISTS Landlord;

DROP TABLE IF EXISTS Tenant;

DROP TABLE IF EXISTS University;

DROP TABLE IF EXISTS User;

# Create tables if necessary.

CREATE TABLE User (

UserId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

FirstName VARCHAR(255) NOT NULL,

LastName VARCHAR(255) NOT NULL,

Email VARCHAR(255),

CONSTRAINT pk\_User\_UserId

PRIMARY KEY (UserId)

);

CREATE TABLE CreditCard (

CardNumber VARCHAR(19) ,#NOT NULL, # max number of credit card digits is 19

ExpirationDate DATE,#NOT NULL, # time is not needed for exp date

UserId INT UNSIGNED,

CONSTRAINT pk\_CreditCard\_CardNumber

PRIMARY KEY (CardNumber),

CONSTRAINT fk\_CreditCard\_User\_UserId

FOREIGN KEY (UserId)

REFERENCES User (UserId)

ON UPDATE CASCADE ON DELETE CASCADE # row updated/deleted if data in parent table updated/deleted

);

CREATE TABLE University (

UniversityId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

Name VARCHAR(255) NOT NULL,

Address VARCHAR(255),

City VARCHAR(255),

State VARCHAR(255),

Zip VARCHAR(255) NOT NULL,

CONSTRAINT pk\_University\_UniversityId

PRIMARY KEY (UniversityId)

);

CREATE TABLE Tenant (

UserId INT UNSIGNED NOT NULL,

UniversityId INT UNSIGNED NOT NULL,

Major VARCHAR(255) NOT NULL,

Gender ENUM('Male', 'Female', 'Brand', 'Unknown') NOT NULL, # 3 options to choose from + NULL for unknown

Bio TEXT, # anything you want to share about yourself

CONSTRAINT pk\_Tenant\_UserId

PRIMARY KEY(UserId),

CONSTRAINT fk\_Tenant\_UserId

FOREIGN KEY (UserId)

REFERENCES User (UserId)

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_Tenant\_University\_UniversityId

FOREIGN KEY (UniversityId)

REFERENCES University (UniversityId)

ON UPDATE CASCADE ON DELETE CASCADE

);

CREATE TABLE Landlord (

UserId INT UNSIGNED NOT NULL,

CONSTRAINT pk\_Landlord\_UserId

PRIMARY KEY(UserId),

CONSTRAINT fk\_Landlord\_UserId

FOREIGN KEY (UserId)

REFERENCES User (UserId)

ON UPDATE CASCADE ON DELETE CASCADE

);

CREATE TABLE FloorPlan (

FloorPlanId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

NumberOfBedrooms INT NOT NULL,

NumberOfBathrooms INT NOT NULL,

CONSTRAINT pk\_FloorPlan\_FloorPlanId

PRIMARY KEY(FloorPlanId)

);

CREATE TABLE Apartment (

ApartmentId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

FloorPlanId INT UNSIGNED NOT NULL,

Address VARCHAR(255),

City VARCHAR(255),

State VARCHAR(255),

Zip VARCHAR(255) NOT NULL,

Sqft INT UNSIGNED,

Name VARCHAR(255),

Description TEXT,

OwnerId INT UNSIGNED NOT NULL,

CONSTRAINT pk\_Apartment\_ApartmentId

PRIMARY KEY(ApartmentId),

CONSTRAINT fk\_Apartment\_FloorPlan\_FloorPlanId

FOREIGN KEY (FloorPlanId)

REFERENCES FloorPlan (FloorPlanId)

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_Apartment\_Landlord\_OwnerId

FOREIGN KEY (OwnerId)

REFERENCES Landlord (UserId)

ON UPDATE CASCADE ON DELETE CASCADE # if landlord is deleted, appts are deleted

);

CREATE TABLE Room (

RoomId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

ApartmentId INT UNSIGNED NOT NULL,

Sqrt INT UNSIGNED,

RoomType ENUM('Master Bedroom', 'Guest Bedroom', 'Other') NOT NULL,

ShareBathroom BOOLEAN,

FloorType ENUM('Hardwood', 'Laminate', 'Carpet', 'Other'),

Description TEXT,

CONSTRAINT pk\_Room\_RoomId

PRIMARY KEY(RoomId),

CONSTRAINT fk\_Room\_Apartment\_ApartmentId

FOREIGN KEY (ApartmentId)

REFERENCES Apartment (ApartmentId)

ON UPDATE CASCADE ON DELETE CASCADE

);

CREATE TABLE ApartmentListing (

ListingId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

ApartmentId INT UNSIGNED NOT NULL,

Title VARCHAR(255), #NOT NULL,

AskingPrice DECIMAL(13,2) NOT NULL,

MoveInDate DATE NOT NULL,

LeaseTermInDays INT,# UNSIGNED NOT NULL,

Content TEXT,# NOT NULL,

Contact VARCHAR(255),# NOT NULL, # required, either phone or email

IsClosed BOOLEAN NOT NULL, # if closed, not shown to public, not available for lease

PostedBy INT UNSIGNED, #NOT NULL,

PostedDateTime TIMESTAMP,# NOT NULL, # for first time listing, later modification on the listing not changing this value

LastModifiedDateTime TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

CONSTRAINT pk\_Listing\_ListingId

PRIMARY KEY(ListingId),

CONSTRAINT fk\_Listing\_Apartment\_ApartmentId

FOREIGN KEY (ApartmentId)

REFERENCES Apartment (ApartmentId)

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_Listing\_Landlord\_PostedBy

FOREIGN KEY (PostedBy)

REFERENCES Landlord (UserId)

ON UPDATE CASCADE ON DELETE CASCADE # if user is deleted, listings are deleted

);

CREATE TABLE Nest (

NestId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

ListingId INT UNSIGNED NOT NULL,

CreatedBy INT UNSIGNED,

CreationDateTime TIMESTAMP,# NOT NULL, # for first time creation, later modification not changing this value

IsDeleted TINYINT(1) NOT NULL, # if deleted, all room reservations are deleted. if no nest is created under a listing, a new nest has to be created in order to put reservations.

IsAcceptedByLandlord TINYINT(1) NOT NULL, # multiple nests under one listing is possible, the full nest has the highest possibility to be accepted by the landlord

LastModifiedDateTime TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

CONSTRAINT pk\_Nest\_NestId

PRIMARY KEY(NestId),

CONSTRAINT fk\_Nest\_Listing\_ListingId

FOREIGN KEY (ListingId)

REFERENCES ApartmentListing (ListingId)

ON UPDATE CASCADE ON DELETE CASCADE, # if listing is deleted, nests are deleted

CONSTRAINT fk\_Nest\_Tenant\_CreatedBy

FOREIGN KEY (CreatedBy)

REFERENCES Tenant (UserId)

ON UPDATE CASCADE ON DELETE SET NULL

);

CREATE TABLE RoomReservation (

ReservationId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

RoomId INT UNSIGNED NOT NULL,

TenantId INT UNSIGNED NOT NULL,

ReservationDateTime TIMESTAMP, # for first time reservation, later modification on the reservation not changing this value

NestId INT UNSIGNED NOT NULL,

OfferedPrice DECIMAL(13,2), # a negotiable price that the tenant is willing to offer. should be lower than the apartment listing price

Contact VARCHAR(255), # contact for negotiation

IsCancelled TINYINT(1), # if cancelled, room under the nest can still be reserved by others, but at any time, only one active reservation for one room is allowed

LastModifiedDateTime TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

CONSTRAINT pk\_RoomReservation\_ReservationId

PRIMARY KEY(ReservationId),

CONSTRAINT uq\_RoomReservation\_Reserve

UNIQUE KEY(RoomId, NestId),

CONSTRAINT fk\_RoomReservation\_Room\_RoomId

FOREIGN KEY (RoomId)

REFERENCES Room (RoomId)

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_RoomReservation\_Tenant\_TenantId

FOREIGN KEY (TenantId)

REFERENCES Tenant (UserId)

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_RoomReservation\_Nest\_NestId

FOREIGN KEY (NestId)

REFERENCES Nest (NestId)

ON UPDATE CASCADE ON DELETE CASCADE # if nest is deleted, reservations are deleted

);

CREATE TABLE Review (

ReviewId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

PostedBy INT UNSIGNED,

PostedDateTime TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP, # for first time review, later modification not changing this value

LastModifiedDateTime TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

Content TEXT,

Rating DECIMAL(2,1) NOT NULL, # 2 digits precision and 1 decimal digits for 0.0 to 5.0 rating range

IsDeleted TINYINT(1), # if poster decided to have it deleted, not shown/calculated into user's average rating

CONSTRAINT pk\_Review\_ReviewId

PRIMARY KEY(ReviewId),

CONSTRAINT fk\_Review\_User\_PostedBy

FOREIGN KEY (PostedBy)

REFERENCES User (UserId)

ON UPDATE CASCADE ON DELETE SET NULL

);

CREATE TABLE UserReview (

ReviewId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

UserId INT UNSIGNED NOT NULL, # either tenant or landlord. tenant can review tenant and landlord; landlord can review tenant

Type ENUM('Tenant','Landlord') NOT NULL,

CONSTRAINT pk\_UserReview\_ReviewId

PRIMARY KEY(ReviewId),

CONSTRAINT fk\_UserReview\_ReviewId

FOREIGN KEY(ReviewId)

REFERENCES Review(ReviewId)

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_UserReview\_User\_UserId

FOREIGN KEY (UserId)

REFERENCES User (UserId)

ON UPDATE CASCADE ON DELETE CASCADE

);

CREATE TABLE ApartmentReview (

ReviewId INT UNSIGNED NOT NULL AUTO\_INCREMENT,

ApartmentId INT UNSIGNED,

CONSTRAINT pk\_ApartmentReview\_ReviewId

PRIMARY KEY(ReviewId),

CONSTRAINT fk\_ApartmentReview\_ReviewId

FOREIGN KEY(ReviewId)

REFERENCES Review(ReviewId)

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_ApartmentReview\_Apartment\_ApartmentId

FOREIGN KEY (ApartmentId)

REFERENCES Apartment (ApartmentId)

ON UPDATE CASCADE ON DELETE CASCADE

)