Designing and Exploring a Real Estate Agency Database

TASK: Consider an organization of your choice for which you are assigned to design and develop a database. Note that this is a real-world database and therefore all entities, attributes, and relationships, and assumptions must be reasonable.

The organization chosen for this project is Keller Williams Realty (KWR) which is an international real estate franchise with headquarters in Austin, Texas. It claimed to be the largest real estate franchise in number of agents and sales volume for 2018 and 2019. The database for this project will only be for all Keller Williams branches in Washington, D.C (Capitol Hill/Dupont Circle/Tenleytown). This database will reflect the sale, purchase and the leasing of properties in the aforementioned neighborhoods. In the interest of simplicity, other business transactions of KWR are ignored.

1) - Define the information content of your database.

a)-Define a set of entities and appropriate attributes for each entity. Minimum 10 entities.

1. Properties (Property ID, Property Type, Property Square Feet, Year Built, Built_By, Number of Bedrooms, Number of Bathrooms, Number of Garages, Number of Stories)
2. Agents (Agent ID, Branch_ID, First_Name, Last_Name, Type_of_Agent, Total Properties Managed, Email, Phone Number)
3. Agent_Properties(Agent_ID, Property_ID)
4. Branch(Branch _ID, Name, Address, Area_Served, Contact_Point)
5. Payment (Payment_ID, Sale Price, Agent Commission, Brokerage Commission, HOA fees, Property taxes, Down_Payment, Loan_Type, Monthly_Mortgage, Mortgage_Insurance, Payment_mode, Payment_Status)
6. Transactions (Transaction ID, Property_ID, Payment_ID Client_ID, Seller_Name, Buyer_Name, Leaser_Name, Leasee_Name)
7. Location (Property_ID, Street Address, City, Neigborhood, State, Zip Code)
8. Property_Client(Property_ID, Client_ID, Role_of_Client)
9. Clients (Client ID, First_Name, Last_Name, Street Address, State_Region, Phone, Email)
10. Sellers (Seller_ID, Client ID)

11. Buyers (Buyer_ID, Client ID)
12. Listing (Listing_ID, Client_ID, Property_ID, Agent_ID, listing_date, listing_price, Type of listing (rent or sale), open)
13. Offer(Offer_ID, Listing_ID, Property_ID, ValidFrom, PriceValidUntil, Price, Currency, Accepted)
14. Admin (Admin_ID, Branch_ID, Admin_Name, Contact, Address, Email_ID)
15. Appoinment (Appointment_ID, Client_ID, Agent_ID appointment_description, appointment_date, Appointment_Time, appointment_status)
16. Appointment_Agent(Appointment_ID, Agent_ID)
17. Appointment_Client(Appointment_ID,Client_ID)
18. Home_Tour(Tour_ID, Buyer_ID, Client_ID, Property_ID, Dateofvisit, Timeofvisit, Agent_ID)
19. Neighborhood_Property(neighborhood_ID, property_ID)
20. Neighborhood(Neighborhood_ID, Neighborhood_Name, Neighborhood_zipcode,)
21. Neighbourhood feature (Neighborhood_ID, Noise Level, walkability score, Number of Bus Stops)
22. Environmental_Risk(Neighborhood_ID, Flood factor, fire factor)
23. Schools (Neighborhood_ID, School_names, Rating, Grades, Type(public or private))
24. Market_stats(Neighborhood_ID, Number_of_homes_for_sale, Avg_days_on_Mkt_, Avg_Home_Price, Avg_Home_Price_per_sqft, avg_sold_price)
25. Property_History (Property_ID, date, event, price)
26. Property_tax (Property_ID, year, taxes, total assessments)
27. Showings(Showing_ID, Seller_ID, Client_ID Property_ID, Dateofshowing, Timeofshowing)

b)-Define a set of relationships that might exist between/among entities and attributes. Such relationships may include one-to-one, one-to-many and many-to-many associations.

One-to-one:

- Property and Transaction
- Transaction and Payment

- Property and Buyer
- Property and Seller
- Property and Location
- Neighborhood and neighborhood_features
- Neighborhood and environmental_risks
- Client and buyer
- Client and seller

One-to-many:

- Property and Listings
- Listing and Offer
- Client and Listing
- Property and Property_Tax
- Property and Property_History
- Neighborhood and Property
- Neighbourhood Schools
- Branch and Agent
- Branch and admin
- Appointment and Agent
- Appointment and Client
- Seller and Showings
- Buyer and Home Tours

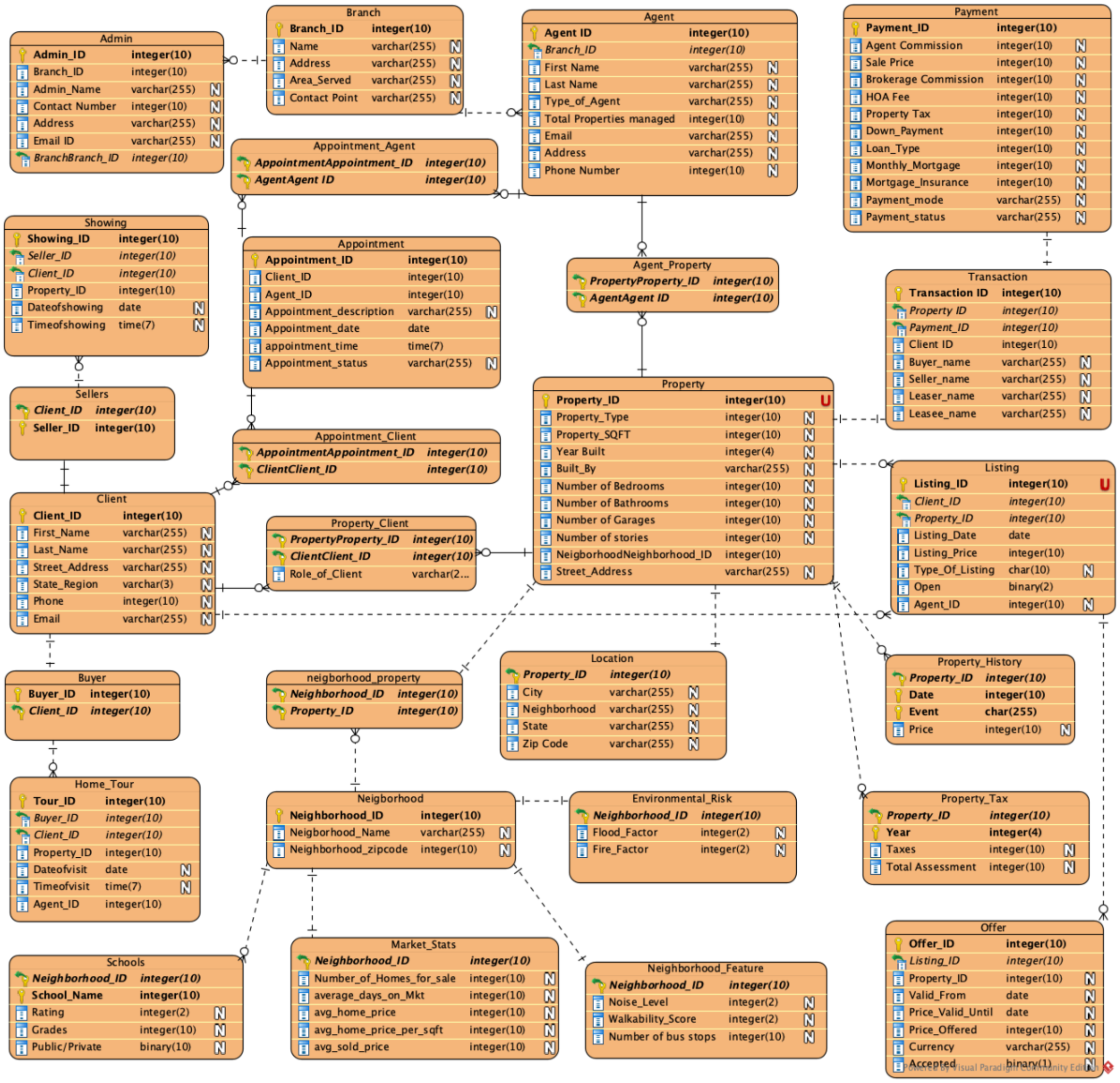
Many-to-many:

- Property and Agent
- Appointment and clients
- Appointment and Agent

c)-Define a set of constraints that may be imposed on data.

- A client can be a seller, a buyer, or both(for different properties).
- A property can have multiple listings (at different times)
- A client can have multiple listings
- A property can have multiple Agents handling it.
- An Agent can manage multiple properties
- An agent can be a buyers agent, a sellers agent or a dual agent
- A property can have one buyer and one seller
- A listing can have many offers
- Appointments can only be made between agents and clients
- An appointment can be between multiple agents (for example, if multiple agents are handling one client or listing) and multiple clients (for example if both the seller are buyer are the clients of the agency)
- A client can have multiple appointments and an appointment can be between multiple clients
- ‘Event’ in Property History can take one of the following values: sold/listed/relisted/price changed)
- A Neighborhood can have multiple properties
- Home Tours can only be conducted by clients who are buyers (Sellers do not need to go on home tours)
- An Agent must be present at the time of the home tour, i.e Agent_ID cannot be null
- ‘Offer’ includes offers made by and received by our clients.
- ‘Showings’ can only be conducted by clients who are sellers (buyers do not need to do showings)
- ‘Transaction’ can have Buyer_Name, Seller_Name, Leaser_Name, Leasee_Name who are not clients of KWR but at leaset one of the above must be the client and must match the Client_ID
- In ‘Transaction’, Leaser Name and Leasee Name can be Null if the Transaction is between a Buyer and Seller.
- In ‘Transaction’, Buyer Name and Seller Name can be Null if the transaction is between Leasee and Leaser.

2) - Define an E-R Diagram for your database design.


Admin
Admin_ID integer(10)
Branch_ID integer(10)
Admin_Name varchar(255) N
Contact Number integer(10) N
Address varchar(255) N
Email ID varchar(255) N
BranchBranch_ID integer(10)
Branch
Branch_ID integer(10)
Name varchar(255) N
Address varchar(255) N
Area_Served varchar(255) N
Contact Point varchar(255) N
Agent
Agent ID integer(10)
Branch_ID integer(10)
First Name varchar(255) N
Last Name varchar(255) N
Type_of_Agent varchar(255) N
Total Properties managed integer(10) N
Email varchar(255) N
Address varchar(255) N
Phone Number integer(10) N
Payment
Payment_ID integer(10)
Agent Commission integer(10) N
Sale Price integer(10) N
Brokerage Commission integer(10) N
HOA Fee integer(10) N
Property Tax integer(10) N
Down_Payment integer(10) N
Loan_Type integer(10) N
Monthly_Mortgage integer(10) N
Mortgage_Insurance integer(10) N
Payment_mode varchar(255) N
Payment_status varchar(255) N
Appointment_Agent
AppointmentAppointment_ID integer(10)
AgentAgent ID integer(10)
Showing
Showing_ID integer(10)
Seller_ID integer(10)
Client_ID integer(10)
Property_ID integer(10)
Dateofshowing date N
Timeofshowing time(7) N
Appointment
Appointment_ID integer(10)
Client_ID integer(10)
Agent_ID integer(10)
Appointment_description varchar(255) N
Appointment_date date
appointment_time time(7)
Appointment_status varchar(255) N
Agent_Property
PropertyProperty_ID integer(10)
AgentAgent ID integer(10)
Transaction
Transaction ID integer(10)
Property ID integer(10)
Payment_ID integer(10)
Client ID integer(10)
Buyer_name varchar(255) N
Seller_name varchar(255) N
Leaser_name varchar(255) N
Leasee_name varchar(255) N
Sellers
Client_ID integer(10)
Seller_ID integer(10)
Appointment_Client
AppointmentAppointment_ID integer(10)
ClientClient_ID integer(10)
Property
Property_ID integer(10) U
Property_Type integer(10) N
Property_SQFT integer(10) N
Year Built integer(4) N
Built_By varchar(255) N
Number of Bedrooms integer(10) N
Number of Bathrooms integer(10) N
Number of Garages integer(10) N
Number of stories integer(10) N
NeigborhoodNeighborhood_ID integer(10)
Street_Address varchar(255) N
Listing
Listing_ID integer(10) U
Client_ID integer(10)
Property_ID integer(10)
Listing_Date date
Listing_Price integer(10)
Type_Of_Listing char(10) N
Open binary(2)
Agent_ID integer(10) N
Client
Client_ID integer(10)
First_Name varchar(255) N
Last_Name varchar(255) N
Street_Address varchar(255) N
State_Region varchar(3) N
Phone integer(10) N
Email varchar(255) N
Property_Client
PropertyProperty_ID integer(10)
ClientClient_ID integer(10)
Role_of_Client varchar(2...
Buyer
Buyer_ID integer(10)
Client_ID integer(10)
neigborhood_property
Neighborhood_ID integer(10)
Property_ID integer(10)
Location
Property_ID integer(10)
City varchar(255) N
Neighborhood varchar(255) N
State varchar(255) N
Zip Code varchar(255) N
Property_History
Property_ID integer(10)
Date integer(10)
Event char(255)
Price integer(10) N
Home_Tour
Tour_ID integer(10)
Buyer_ID integer(10)
Client_ID integer(10)
Property_ID integer(10)
Dateofvisit date N
Timeofvisit time(7) N
Agent_ID integer(10)
Neigborhood
Neighborhood_ID integer(10)
Neigborhood_Name varchar(255) N
Neighborhood_zipcode integer(10) N
Environmental_Risk
Neighborhood_ID integer(10)
Flood_Factor integer(2) N
Fire_Factor integer(2) N
Property_Tax
Property_ID integer(10)
Year integer(4)
Taxes integer(10) N
Total Assessment integer(10) N
Schools
Neighborhood_ID integer(10)
School_Name integer(10)
Rating integer(2) N
Grades integer(10) N
Public/Private binary(10) N
Market_Stats
Neighborhood_ID integer(10)
Number_of_Homes_for_sale integer(10) N
average_days_on_Mkt integer(10) N
avg_home_price integer(10) N
avg_home_price_per_sqft integer(10) N
avg_sold_price integer(10) N
Neighborhood_Feature
Neighborhood_ID integer(10)
Noise_Level integer(2) N
Walkability_Score integer(2) N
Number of bus stops integer(10) N
Offer
Offer_ID integer(10)
Listing_ID integer(10)
Property_ID integer(10) N
Valid_From date N
Price_Valid_Until date N
Price_Offered integer(10) N
Currency varchar(255) N
Accepted binary(1) N
Powered by Visual Paradigm Community Edition

3) - Define a relational schema for your database design.

Make sure that you have both one-to-many and many-to-many associations.

a)-Define one or more realistic key(s) for every relation scheme. Use both simple and composite keys.

All keys are indicated in Answer 1.a where the Primary Keys are underlined for every relation.

b)-Define a realistic set of Functional / Multi-Valued Dependencies (when appropriate) for every relation scheme.

1. Property ID → Property Type, Property Square Feet, Year Built, Built_By, Number of Bedrooms, Number of Bathrooms, Number of Garages, Number of Stories
2. Agent ID → Branch_ID, First_Name, Last_Name, Type_of_Agent, Total Properties Managed, Email, Phone Number
3. Branch _ID → Name, Address, Area_Served, Contact_Point)
4. Transaction ID→Property_ID, Payment_ID, Client_ID, Buyer_Name, Seller_Name, Leaser_name, Leasee_name
5. Payment_ID →Sale Price, Agent Commission, Brokerage Commission, HOA fees, Property taxes, Down_Payment, Loan_Type, Monthly_Mortgage, Mortgage_Insurance, Payment_mode, Payment_status
6. Property_ID →Street Address, City, Neighborhood, State, Zip Code
7. Client ID → First_Name, Last_Name, Street Address, State_Region, Phone, Email
8. Property_ID, Client_ID →Role_of_Client
9. Listing_ID → Client_ID, Property_ID, Agent_ID, listing_date, listing_price, Type of listing (rent or sale), open
10. Offer_ID →Listing_ID, Property_ID, ValidFrom, PriceValidUntil, Price, Currency, Accepted
11. Admin_ID →Branch_ID, Admin_Name, Contact, Address, Email_ID
12. Appointment_ID→client_ID, Agent_ID appointment_description, appointment_date, Appointment_Time, Appointment_status)
13. Tour_id→ Buyer_ID, Client_ID, Property_ID, Dateofvisit, Timeofvisit, Agent_ID
14. Neighborhood_ID →Neighborhood_Name, Neighborhood_zipcode
15. Neighborhood_ID →Noise Level, walkability score, Number of Bus stops
16. Neighborhood_ID→Flood factor, fire factor
17. Neighborhood_ID, School_name →Rating, Grades, Type(public or private)
18. Neighborhood_ID→Number_of_homes_for_sale, Avg_days_on_Mkt_, Avg_Home_Price, Avg_Home_Price_per_sqft, avg_sold_price
19. Property_ID, date, event →price

20. Property_ID, year →taxes, total assessments
21. Showing_ID→Seller_ID, Client_ID, Property_ID, Dateofshowing, Timeofshowing

C-Check whether your relational schema is in 2NF, 3NF, BCNF, 4NF.

1. Property ID → Property Type, Property Square Feet, Year Built, Built_By, Number of Bedrooms, Number of Bathrooms, Number of Garages, Number of Stories
    a. It is in 2NF because it is in 1NF (there is atomicity of value in every attribute) and there are no partial dependencies.
    b. It is in 3NF because it is in 2NF and there are no non-key attributes that are transitively dependent on the primary key.
    c. It is BCNF because there are no non-trivial functional dependencies of attributes on anything other the primary key
    d. It is 4NF because there is no multi-vlued dependency.
2. Agent ID → Branch_ID, First_Name, Last_Name, Type_of_Agent, Total Properties Managed, Email, Phone Number
    a. It is in 2NF because it is in 1NF (there is atomicity of value in every attribute) and there are no partial dependencies.
    b. It is in 3NF because it is in 2NF and there are no non-key attributes that are transitively dependent on the primary key.
    c. It is BCNF because there are no non-trivial functional dependencies of attributes on anything other the primary key
    d. It is 4NF because there is no multi-vlued dependency.
3. Branch_ID → Name, Address, Area_Served, Contact_Point
    a. It is in 2NF because it is in 1NF (there is atomicity of value in every attribute) and there are no partial dependencies.
    b. It is in 3NF because it is in 2NF and there are no non-key attributes that are transitively dependent on the primary key.
    c. It is BCNF because there are no non-trivial functional dependencies of attributes on anything other the primary key
    d. It is 4NF because there is no multi-vlued dependency

4. Payment_ID →Sale Price, Agent Commission, Brokerage Commission, HOA fees, Property taxes, Down_Payment, Loan_Type, Monthly_Mortgage, Mortgage_Insurance, Payment_mode, Payment_status
    a. It is in 2NF because it is in 1NF (there is atomicity of value in every attribute) and there are no partial dependencies.
    b. It is in 3NF because it is in 2NF and there are no non-key attributes that are transitively dependent on the primary key.
    c. It is BCNF because there are no non-trivial functional dependencies of attributes on anything other the primary key
    d. It is 4NF because there is no multi-vlued dependency
5. Transaction ID→Property_ID, Payment_ID, Client_ID, Buyer_name, seller_name, Leaser_name, Leasee_name
6. Property_ID →Street Address, City, Neighborhood, State, Zip Code, Latitude, Longitude
7. Client ID → Client Name, Address, Phone, Email
8. Listing_ID →Property_ID, agent, listing_date, listing_price, Type of listing (rent or sale), open
9. Offer_id →Listing_ID, Property_ID, ValidFrom, PriceValidUntil, Price, Currency, Accepted
10. Admin_ID →Branch_ID, Admin_name, contact, address, email)
11. appointment_id→client_ID, Agent_ID, appointment_description, appointment_date, appointment_time, appointment_status
12. Tour_id→Buyer_ID, Property_ID, Dateofvisit, Timeofvisit, Agent_ID
13. Neighborhood_ID →Neighborhood_Name, Neighborhood_zipcode
14. Neighborhood_ID →Noise Level, walkability score, Number of Bus stops
15. Neighborhood_ID→Flood factor, fire factor
16. Neighborhood_ID, School_name →Rating, Grades, Type(public or private)
17. Neighborhood_ID→Number_of_homes_for_sale, Avg_days_on_Mkt_, Avg_Home_Price, Avg_Home_Price_per_sqft, avg_sold_price
18. Property_ID, date, event →price
19. Property_ID, year →taxes, total assessments
20. Showing_ID→seller_ID, client_ID, Property_ID, Dateofshowing, Timeofshowing

d)-Put your relational schema in the highest normal form that is possible.
Note that, every relation scheme should be in a specific normal form in order to have the relational schema in that normal form.

NOTE: Please provide a detailed explanation for every question when appropriate.

All functional dependencies above (from 1 to 20) are in the highest normal form i.e they satisfy the requirements mentioned aboved to be in 1NF, 2NF, 3NF, BCNF and 4NF. The following conditions are true for all the FDs mentioned above but have not been repeated in the interest of space:

a. It is in 2NF because it is in 1NF (there is atomicity of value in every attribute) and there are no partial dependencies.
b. It is in 3NF because it is in 2NF and there are no non-key attributes that are transitively dependent on the primary key.
c. It is BCNF because there are no non-trivial functional dependencies of attributes on anything other the primary key
d. It is 4NF because there is no multi-vlued dependency

4) Implementation: Create your database using MySQL, or... to Perform the following operations.

Create 4 tables from your database project that are connected/linked together and insert few dummy records into these tables. Then use these tables to answer the following queries.

Creating Database

create database if not exists RealEstate;

use RealEstate;

Creating and Populating the table Property

create table if not exists Property

```
(Property_ID double not null primary key,

Property_Type varchar(40),

Property_SQFT double,

Year_Built YEAR,

Built_By varchar(40),

Number_of_Bedrooms double,

Number_of_Bathrooms double,

Number_of_Garages double,

Number_of_Stories double);

insert into Property values('101','Single Family','2352',1939,'Seller','4','3','1','2');

insert into Property values('102','Apartment','700',2021,'Seller','1','1','0','1');

insert into Property values('103','Townhouse','3200',1975,'Third Party','3','3','1','2');

insert into Property values('104','Single Family','4500',1989,'Seller','6','3','2','2');

insert into Property values('105','Townhouse','1900',2001,'Third Party','2','2','1','3');

insert into Property values('106','Single Family','4885',1979,'Seller','5','4','2','3');

insert into Property values('107','Apartment','1200',2007,'Third Party','2','2','0','1');

insert into Property values('108','Townhouse','2330',2009,'Third Party','4','4','1','2');

insert into Property values('109','Single Family','8320',1947,'Seller','8','9','4','3');

insert into Property values('110','Single Family','1352',1959,'Third Party','4','3','1','2');
insert into Property values('111','Townhouse','2100',1990,'Third Party','4','3','1','3');

insert into Property values('112','Single Family','4352',1992,'Third Party','7','4','1','2');

insert into Property values('113','Single Family','1750',2022,'Seller','4','3','1','2');

insert into Property values('114','Townhouse','2700',2000,'Third Party','4','3','1','2');

insert into Property values('115','Apartment','475',1932,'Seller','0','1','0','1');
```

Creating and Populating the table Location

```
create table if not exists Location

(Property_ID double not null primary key,

Street_Address varchar(50),

Neighborhood varchar(50),

City varchar(50),

State varchar(50),

Zip_Code varchar(10));

insert into Location values('101','7708 Georgia Ave NW','Tanleytown','Washington
DC','Washington DC','20024');

insert into Location values('102','1471 Bangor Ste SE','Du Pont Circle','Washington
DC','Washington DC','20002');

insert into Location values('103','1811 Connecticut Ave NW','Capitol Hill','Washington
DC','Washington DC','20022');
```

```
insert into Location values('104','1365 Kennedy St NW','Tanleytown','Washington
DC','Washington DC','20011');

insert into Location values('105','5315 Connecticut Ave NW','Capitol Hill','Washington
DC','Washington DC','20023');

insert into Location values('106','1343 Otis Pl NW','Du Pont Circle','Washington
DC','Washington DC','20011');

insert into Location values('107','3108 Westover Dr SE','Tanleytown','Washington
DC','Washington DC','20020');

insert into Location values('108','922 Connecticut Ave NW','Capitol Hill','Washington
DC','Washington DC','20024');

insert into Location values('109','4600 Connecticut Ave NW','Capitol Hill','Washington
DC','Washington DC','20001');

insert into Location values('110','4308 Connecticut Ave NW','Tanleytown','Washington
DC','Washington DC','20033');

insert into Location values('111','4400 Texas Ave NW','Du Pont Circle','Washington
DC','Washington DC','20021');

insert into Location values('112','3111 Cypress Dr SE','Du Pont Circle','Washington
DC','Washington DC','20007');

insert into Location values('113','957 Cameron Pl NW','Tanleytown','Washington
DC','Washington DC','20008');

insert into Location values('114','2789 Connecticut Ave NW','Capitol Hill','Washington
DC','Washington DC','20016');

insert into Location values('115','1811 M St SE','Tanleytown','Washington DC','Washington
DC','20002');
```

Creating and Populating the table Agent

```
create table if not exists Agent

(Agent_ID varchar(10) not null primary key,

Branch_ID double,

First_Name varchar(10),

Last_Name varchar(10),

Type_of_Agent varchar(20),

Total_Properties_managed double,

Email varchar(30) UNIQUE,

Phone_Number double UNIQUE);

insert into Agent values('1001','2001','Sarah','Adams','Buyers
Agent','39','sa@kwr.com','9374446669');

insert into Agent values('1002','2002','Jim','Hopper','Sellers
Agent','29','jh@kwr.com','1235556669');

insert into Agent values('1003','2003','Nancy','Wheeler','Buyers
Agent','40','nw@kwr.com','9987776669');

insert into Agent values('1004','2003','Mike','Wheeler','Sellers
Agent','83','mw@kwr.com','1239996669');

insert into Agent values('1005','2002','Ali','Meer','Dual Agent','22','am@kwr.com','3334446669');

insert into Agent values('1006','2001','Will','Byers','Dual
Agent','12','wb@kwr.com','2984446669');
```

insert into Agent values('1007','2003','Jane','johnson','Dual Agent','62','jj@kwr.com','1239446669');

insert into Agent values('1008','2002','Dustin','Bishop','Sellers Agent','9','db@kwr.com','9834446669');

insert into Agent values('1009','2002','Nick','Miller','Dual Agent','8','nm@kwr.com','128896669');

insert into Agent values('1010','2001','Winston','Schmidt','Buyers Agent','3','ws@kwr.com','1094446669');

Creating and Populating the table Client

create table if not exists Client

(Client_ID double not null primary key,

First_Name varchar(20),

Last_Name varchar(20),

Street_Address varchar(60),

State_Region varchar(3),

Phone_Number double UNIQUE,

Email varchar(30) UNIQUE);

insert into Client values('301','Billy','Hargrove','2323 Dulles Station Ave, Herndon', 'VA','1094678669','bh@kwr.com');

insert into Client values('302','Ian','Thomas','4343 Baron Rolfe Ave, Ashburn', 'VA','1094448649','it@kwr.com');

```
insert into Client values('303','Steve','Harrington','7865 Autumn Cameron Dr, Woodbridge',
'VA','3575446669','sh@kwr.com');

insert into Client values('304','Max','Mayfield','2234 Fuller Back BLVD, Springfield',
'VA','7865446669','mm@kwr.com');

insert into Client values('305','Robin','Buckley','9938 Fuller Baron Dr, Oakton',
'VA','7856446669','rb@kwr.com');

insert into Client values('306','Lucas','Sinclair','1657 Rolfe Station Ave, Charlestown',
'WV','7642146669','ad@kwr.com');

insert into Client values('307','Alison','DiLaurentis','7379 Stoney Fuller Dr, Bethesda',
'PA','1094345679','ls@kwr.com');

insert into Client values('308','Aria','Hastings','7435 Brook Back BLVD, Rockville',
'LA','1094765439','23dvv@kwr.com');

insert into Client values('309','Ezra','Fitz','5707 Vaughn Amonate Dr, Alexandria',
'CA','34579086669','dscsc@kwr.com');

insert into Client values('310','Hanna','Marin','6435 Venture Dunn BLVD, Frederick',
'MD','1053456769','sdcsd@kwr.com');

insert into Client values('311','Mona','Vanderwal','7799 Fuller Station Ave, Reston',
'CA','1094786544','dscsd@kwr.com');

insert into Client values('312','Toby','Cavanaugh','4367 Dunn Back Dr, Rockville',
'AR','4343446669','gfdfgf@kwr.com');

insert into Client values('313','Emily','Fields','8986 Gerald Fuller Dr, Arlingston',
'VA','1434346669','fdf@kwr.com');

insert into Client values('314','Jenna','Marshall','5689 Radio Rolfe BLVD, Herndon',
'VA','10944322669','dfdggd@kwr.com');
```

```
insert into Client values('315','Caleb','Rivers','4578 Fuller Back Dr, Silver Spring',
'CT','45678446669','dfgdf@kwr.com');

insert into Client values('316','Wren','Kingston','8966 Venture Baron Ave, Alexandria',
'DC','153346669','ksdhksd@kwr.com');

insert into Client values('317','Lucas','Gottesman','7654 Dunn Fuller Dr,  Silver
Spring','AZ','1445546669','ksdhfkjdf@kwr.com');

insert into Client values('318','Paige','McCullers','4856 Rolfe Venture Ave, Bethesda',
'MD','1093498469','skdhfksd@kwr.com');

insert into Client values('319','Noel','Kahn','6665 Dulles Baron Dr, Alexandria',
'VA','10932465469','ksdhf@kwr.com');

insert into Client values('320','Gabriel','Hargrove','6788 Fuller Venture Ave, Rockville',
'MD','4577446669','ksdjkd@kwr.com');
```

Creating and Populating the table Sellers

```
create table if not exists Sellers

(Client_ID double not null,

Seller_ID varchar(20) not null,

primary key(Client_ID, Seller_ID));

insert into Sellers values('301','S-301');

insert into Sellers values('303','S-303');

insert into Sellers values('305','S-305');

insert into Sellers values('307','S-307');
insert into Sellers values('309','S-309');

insert into Sellers values('311','S-311');

insert into Sellers values('313','S-313');

insert into Sellers values('315','S-315');

insert into Sellers values('317','S-317');

insert into Sellers values('318','S-318');

insert into Sellers values('319','S-319');

insert into Sellers values('320','S-320');
```

Creating and Populating the table Buyers

```
create table if not exists Buyers

(Client_ID double not null,

Buyer_ID varchar(20) not null,

primary key(Client_ID, Buyer_ID));

insert into Buyers values('302','B-302');

insert into Buyers values('304','B-304');

insert into Buyers values('306','B-306');

insert into Buyers values('308','B-308');

insert into Buyers values('310','B-310');

insert into Buyers values('312','B-312');
insert into Buyers values('314','B-314');

insert into Buyers values('316','B-316');
```

Creating and Populating the table Agent_Properties

```
create table if not exists Agent_Properties

(Agent_ID double not null,

Property_ID double not null,

primary key(Agent_ID, Property_ID));

insert into Agent_Properties values('1001','112');

insert into Agent_Properties values('1002','111');

insert into Agent_Properties values('1002','105');

insert into Agent_Properties values('1003','113');

insert into Agent_Properties values('1004','110');

insert into Agent_Properties values('1004','115');

insert into Agent_Properties values('1004','106');

insert into Agent_Properties values('1005','101');

insert into Agent_Properties values('1005','104');

insert into Agent_Properties values('1005','102');

insert into Agent_Properties values('1006','102');

insert into Agent_Properties values('1006','115');
insert into Agent_Properties values('1007','101');

insert into Agent_Properties values('1007','115');

insert into Agent_Properties values('1008','109');

insert into Agent_Properties values('1008','107');

insert into Agent_Properties values('1009','103');

insert into Agent_Properties values('1009','104');

insert into Agent_Properties values('1010','114');
```

Creating and Populating the table Property_Client

```
create table if not exists Property_Client

(Client_ID double not null,

Property_ID double not null,

Role_of_Client varchar(30) not null,

primary key(Client_ID, Property_ID));

insert into Property_Client values('301','101','Seller');

insert into Property_Client values('302','101', 'Buyer');

insert into Property_Client values('303','102', 'Seller');

insert into Property_Client values('304','102', 'Buyer');

insert into Property_Client values('305','103', 'Seller');

insert into Property_Client values('306','103', 'Buyer');
insert into Property_Client values('307','104', 'Seller');

insert into Property_Client values('308','104', 'Buyer');

insert into Property_Client values('309','105', 'Seller');

insert into Property_Client values('310','112', 'Buyer');

insert into Property_Client values('311','106', 'Seller');

insert into Property_Client values('312','113', 'Buyer');

insert into Property_Client values('313','107', 'Seller');

insert into Property_Client values('314','114', 'Buyer');

insert into Property_Client values('315','108', 'Seller');

insert into Property_Client values('316','115', 'Buyer');

insert into Property_Client values('317','109', 'Seller');

insert into Property_Client values('318','115', 'Seller');

insert into Property_Client values('319','110', 'Seller');

insert into Property_Client values('320','111', 'Seller');
```

A) You are required to execute SQL queries that include the following operations. For each query, provide the SQL statements along with the output. For each of the following, try different SQL statements (i.e., using one relation, more than one relations,...).

- Create tables: (just for creating 4 tables, not all)

- Select:

<u>Select involving one/more conditions in Where Clause</u>

Query: Find the names of all clients who sold property located in Tanleytown, had greater than 2 bedrooms and had greater than 2 bathrooms.

```
select First_Name, Last_Name

from Client

where Client_ID in

(select Client_ID

from Property_Client

where Role_of_Client='Seller'

and Property_ID in

(select Property_ID

from Property

where Number_of_Bedrooms>2 and

Number_of_Bathrooms>2 and

Property_ID in

(Select Property_ID

from Location

where Neighborhood="Tanleytown")));
```

```
# First_Name  Last_Name

Billy         Hargrove
```

```
Alison	DiLaurentis

Noel	Kahn
```

Select with aggregate functions (i.e., SUM,MIN,MAX,AVG,COUNT)

Query: Find the names of all agents who sold properties that have more bedrooms than the average number of bedrooms in a property

```
select avg(Number_of_Bedrooms)

from Property;
```

```
# avg(Number_of_Bedrooms)

3.866
```

```
select First_Name, Last_Name

from Agent

where Agent_ID in

(select Agent_ID

from Agent_Properties

where Property_ID in

(select Property_ID

from Property

where Number_of_Bedrooms>3.86));
```

```
# First_Name	Last_Name
```

Sarah	Adams
Jim	Hopper
Nancy	Wheeler
Mike	Wheeler
Ali	Meer
Jane	johnson
Dustin	Bishop
Nick	Miller
Winston	Schmidt

Query: Find the name and address of the client who purchased the property with the maximum SQFT and the name and address of the client who sold the property with the maximum SQFT.

```
select max(Property_SQFT)

from Property

where Property_ID in

(Select Property_ID

from Property_Client

where Role_of_Client="Seller");

# max(Property_SQFT)

8320
```

```
select First_Name, Last_Name, Street_Address

from Client

where Client_ID in

(select Client_ID

from Property_Client

where Role_of_Client="Seller"

and Property_ID in

(select Property_ID

from Property

where Property_SQFT=8320));

# First_Name  Last_Name     Street_Address

Lucas         Gottesman     7654 Dunn Fuller Dr,  Silver Spring

select max(Property_SQFT)

from Property

where Property_ID in

(Select Property_ID

from Property_Client

where Role_of_Client="Buyer");

# max(Property_SQFT)
```

```
4500

select First_Name, Last_Name, Street_Address

from Client

where Client_ID in

(select Client_ID

from Property_Client

where Role_of_Client="Buyer"

and Property_ID in

(select Property_ID

from Property

where Property_SQFT=4500));

# First_Name  Last_Name      Street_Address

Aria             Hastings         7435 Brook Back BLVD, Rockville
```

<u>Nested Select</u>

Query: List the properties and their neighborhood managed by the agent who has managed the most properties

```
select max(Total_Properties_managed)

from Agent;

# max(Total_Properties_managed)
```

83

```
Select Property_ID, Neighborhood

from Location

where Property_ID in

(select Property_ID

from Agent_Properties

where Agent_ID in

(select Agent_ID

from Agent

where Total_Properties_managed=83));
```

```
# Property_ID Neighborhood

106           Du Pont Circle

110           Tanleytown

115           Tanleytown
```

By, Order By clause

Query:List down the agent names by the SQFT of the property that they managed in descending order

```
select First_Name, Last_Name, Property_SQFT

from Agent, Property, Agent_Properties
```

```
where Agent.Agent_ID=Agent_Properties.Agent_ID and

Agent_Properties.Property_ID=Property.Property_ID

ORDER by Property_SQFT desc;
```

```
# First_Name  Last_Name    Property_SQFT

Dustin  Bishop      8320

Mike    Wheeler     4885

Ali     Meer        4500

Nick    Miller      4500

Sarah   Adams       4352

Nick    Miller      3200

Winston Schmidt     2700

Ali     Meer        2352

Jane    johnson     2352

Jim     Hopper      2100

Jim     Hopper      1900

Nancy   Wheeler     1750

Mike    Wheeler     1352

Dustin  Bishop      1200

Ali     Meer        700
```

```
Will    Byers       700

Mike    Wheeler     475

Will    Byers       475

Jane    johnson     475
```

select with Having, Group

Query:Find all the property IDs and address of all the properties that were sold by and bought by clients of KWR

```
select Property_ID, Count(Property_ID)

FROM Property_Client

group by Property_ID

Having Count(Property_ID)>1;
```

```
# Property_ID Count(Property_ID)

101          2

102          2

103          2

104          2

115          2
```

select involving the Union operation

Query: Find the IDs and names of all agents who managed property 101 along with the IDs and names of the buyer and seller of that property.

```
select Agent_ID as ID,First_Name, Last_Name

from Agent

where Agent_ID in

(Select Agent_ID

from Agent_Properties

where Property_ID='101')

union

select Client_ID, First_Name, Last_Name

from Client

where Client_ID in

(Select Client_ID

from Property_Client

where Property_ID='101');
```

```
# ID    First_Name    Last_Name

1005    Ali     Meer

1007    Jane    johnson

301     Billy   Hargrove
```

```
302	Ian	Thomas
```

- Insert:

insert one tuple into a table (for 2 tables, just add 3 records for each table)

Query: insert a property with ID 116 that is a 10,250 SQFT single family home built in 1929, has 10 bedrooms, 12 bathrooms, 5 garages and 2 story.

```
insert into Property values('116','Single family','10250',1929, "Third Party",'10','12','5','2');
```

Property Before Query:

```
# Property_ID Property_TypeProperty_SQFT	Year_Built	Built_By
	Number_of_Bedrooms	Number_of_Bathrooms	Number_of_Garages
	Number_of_Stories

101	Single Family	2352	1939	Seller	4	3	1	2

102	Apartment	700	2021	Seller	1	1	0	1

103	Townhouse	3200	1975	Third Party	3	3	1	2

104	Single Family	4500	1989	Seller	6	3	2	2

105	Townhouse	1900	2001	Third Party	2	2	1	3

106	Single Family	4885	1979	Seller	5	4	2	3

107	Apartment	1200	2007	Third Party	2	2	0	1

108	Townhouse	2330	2009	Third Party	4	4	1	2

109	Single Family	8320	1947	Seller	8	9	4	3

110	Single Family	1352	1959	Third Party	4	3	1	2
```

```
111     Townhouse       2100    1990    Third Party     4       3       1       3

112     Single Family   4352    1992    Third Party     7       4       1       2

113     Single Family   1750    2022    Seller  4       3       1       2

114     Townhouse       2700    2000    Third Party     4       3       1       2

115     Apartment       475     1932    Seller  0       1       0       1
```

Property After Query:

```
# Property_ID Property_TypeProperty_SQFT        Year_Built      Built_By
        Number_of_Bedrooms      Number_of_Bathrooms     Number_of_Garages
        Number_of_Stories

101     Single Family   2352    1939    Seller  4       3       1       2

102     Apartment       700     2021    Seller  1       1       0       1

103     Townhouse       3200    1975    Third Party     3       3       1       2

104     Single Family   4500    1989    Seller  6       3       2       2

105     Townhouse       1900    2001    Third Party     2       2       1       3

106     Single Family   4885    1979    Seller  5       4       2       3

107     Apartment       1200    2007    Third Party     2       2       0       1

108     Townhouse       2330    2009    Third Party     4       4       1       2

109     Single Family   8320    1947    Seller  8       9       4       3

110     Single Family   1352    1959    Third Party     4       3       1       2

111     Townhouse       2100    1990    Third Party     4       3       1       3
```

112	Single Family	4352	1992	Third Party	7	4	1	2
113	Single Family	1750	2022	Seller	4	3	1	2
114	Townhouse	2700	2000	Third Party	4	3	1	2
115	Apartment	475	1932	Seller	0	1	0	1
116	**Single family**	**10250**	**1929**	**Third Party**	**10**	**12**	**5**	**2**

insert into Location values('116','7712 Georgia ct NW','Tanleytown','Washington DC','Washington DC','20214');

Location Before Query:

# Property_ID	Street_Address	Neighborhood	City	State	Zip_Code
101	7708 Georgia Ave NW	Tanleytown	Washington DC	Washington DC	20024
102	1471 Bangor Ste SE	Du Pont Circle	Washington DC	Washington DC	20002
103	1811 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20022
104	1365 Kennedy St NW	Tanleytown	Washington DC	Washington DC	20011
105	5315 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20023
106	1343 Otis PI NW	Du Pont Circle	Washington DC	Washington DC	20011
107	3108 Westover Dr SE	Tanleytown	Washington DC	Washington DC	20020

108	922 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20024
109	4600 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20001
110	4308 Connecticut Ave NW	Tanleytown	Washington DC	Washington DC	20033
111	4400 Texas Ave NW	Du Pont Circle	Washington DC	Washington DC	20021
112	3111 Cypress Dr SE	Du Pont Circle	Washington DC	Washington DC	20007
113	957 Cameron PI NW	Tanleytown	Washington DC	Washington DC	20008
114	2789 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20016
115	1811 M St SE	Tanleytown	Washington DC	Washington DC	20002

Property After Query:

# Property_ID	Street_Address	Neighborhood	City	State	Zip_Code
101	7708 Georgia Ave NW	Tanleytown	Washington DC	Washington DC	20024
102	1471 Bangor Ste SE	Du Pont Circle	Washington DC	Washington DC	20002
103	1811 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20022
104	1365 Kennedy St NW	Tanleytown	Washington DC	Washington DC	20011
105	5315 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20023

106	1343 Otis PI NW	Du Pont Circle	Washington DC	Washington DC	20011
107	3108 Westover Dr SE	Tanleytown	Washington DC	Washington DC	20020
108	922 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20024
109	4600 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20001
110	4308 Connecticut Ave NW	Tanleytown	Washington DC	Washington DC	20033
111	4400 Texas Ave NW	Du Pont Circle	Washington DC	Washington DC	20021
112	3111 Cypress Dr SE	Du Pont Circle	Washington DC	Washington DC	20007
113	957 Cameron PI NW	Tanleytown	Washington DC	Washington DC	20008
114	2789 Connecticut Ave NW	Capitol Hill	Washington DC	Washington DC	20016
115	1811 M St SE	Tanleytown	Washington DC	Washington DC	20002
116	**7712 Georgia ct NW**	**Tanleytown**	**Washington DC**	**Washington DC**	**20214**

insert a set of tuples (by using another select statement)

Query: Insert agents who managed Property 116 into table 'Agent' and 'Agent_Properties'

```
insert into Agent values('1011','2002','Dustin','Crew','Sellers
Agent','27','11db@kwr.com','98347799669');
```

```
insert into Agent values('1012','2002','JC','Tag','Dual
Agent','48','12wnm@kwr.com','128899000');
```

```
insert into Agent values('1013','2001','Macy','Wills','Buyers
Agent','13','wwws@kwr.com','10944489789');
```

Agent Before Query:

```
# Agent_ID	Branch_ID	First_Name	Last_Name	Type_of_Agent
	Total_Properties_managed	Email	Phone_Number

1001	2001	Sarah	Adams	Buyers Agent	39	sa@kwr.com	9374446669

1002	2002	Jim	Hopper	Sellers Agent	29	jh@kwr.com	1235556669

1003	2003	Nancy	Wheeler		Buyers Agent	40	nw@kwr.com	9987776669

1004	2003	Mike	Wheeler		Sellers Agent	83	mw@kwr.com	1239996669

1005	2002	Ali	Meer	Dual Agent	22	am@kwr.com	3334446669

1006	2001	Will	Byers	Dual Agent	12	wb@kwr.com	2984446669

1007	2003	Jane	johnson		Dual Agent	62	jj@kwr.com	1239446669

1008	2002	Dustin	Bishop	Sellers Agent	9	db@kwr.com	9834446669

1009	2002	Nick	Miller	Dual Agent	8	nm@kwr.com	128896669

1010	2001	Winston		Schmidt		Buyers Agent	3	ws@kwr.com	1094446669
```

Agent After Query:

```
# Agent_ID	Branch_ID	First_Name	Last_Name	Type_of_Agent
	Total_Properties_managed	Email	Phone_Number
```

1001	2001	Sarah	Adams	Buyers Agent	39	sa@kwr.com	9374446669
1002	2002	Jim	Hopper	Sellers Agent	29	jh@kwr.com	1235556669
1003	2003	Nancy	Wheeler	Buyers Agent	40	nw@kwr.com	9987776669
1004	2003	Mike	Wheeler	Sellers Agent	83	mw@kwr.com	1239996669
1005	2002	Ali	Meer	Dual Agent	22	am@kwr.com	3334446669
1006	2001	Will	Byers	Dual Agent	12	wb@kwr.com	2984446669
1007	2003	Jane	johnson	Dual Agent	62	jj@kwr.com	1239446669
1008	2002	Dustin	Bishop	Sellers Agent	9	db@kwr.com	9834446669
1009	2002	Nick	Miller	Dual Agent	8	nm@kwr.com	128896669
1010	2001	Winston	Schmidt	Buyers Agent	3	ws@kwr.com	1094446669
1011	**2002**	**Dustin**	**Crew**	**Sellers Agent**	**27**	**11db@kwr.com**	**98347799669**
1012	**2002**	**JC**	**Tag**	**Dual Agent**	**48**	**12wnm@kwr.com**	**128899000**
1013	**2001**	**Macy**	**Wills**	**Buyers Agent**	**13**	**wwws@kwr.com**	**10944489789**

insert into Agent_Properties values('1011','116');

insert into Agent_Properties values('1012','116');

insert into Agent_Properties values('1013','116');

Agent_Properties Before Query:

'1001','112'

'1002','105'

'1002','111'

'1003','113'

'1004','106'

'1004','110'

'1004','115'

'1005','101'

'1005','102'

'1005','104'

'1006','102'

'1006','115'

'1007','101'

'1007','115'

'1008','107'

'1008','109'

'1009','103'

'1009','104'

'1010','114’

Agent_Properties After Query:

'1001','112'

'1002','105'

'1002','111'

'1003','113'

'1004','106'

'1004','110'

'1004','115'

'1005','101'

'1005','102'

'1005','104'

'1006','102'

'1006','115'

'1007','101'

'1007','115'

'1008','107'

'1008','109'

'1009','103'

'1009','104'

'1010','114'

'1011','116'

'1012','116'

'1013','116'

insert involving two tables

Query: Create a table which has the IDs, names and properties managed by all the Dual Agents of KWR (Dual Agents are those who represent both the Seller and buyer of the property)

```
Create table DualAgents as

select Agent.Agent_ID, First_Name, Last_Name, Property_ID

from Agent, Agent_Properties

where Agent.Agent_ID=Agent_Properties.Agent_ID

and

Type_of_Agent="Dual Agent";

Select * from DualAgents;
```

```
# Agent_ID    First_Name    Last_Name    Property_ID

1005          Ali           Meer         101

1005          Ali           Meer         102

1005          Ali           Meer         104

1006          Will          Byers        102

1006          Will          Byers        115
```

```
1007        Jane        johnson     101

1007        Jane        johnson     115

1009        Nick        Miller      103

1009        Nick        Miller      104
```

Query: Create a table which shows the years in which the oldest property, in each of the three neighborhood, was built, in ascending order.

```
create table Oldest_Properties as

select min(Year_Built), Neighborhood

from Property, Location

where Property.Property_ID=Location.Property_ID

group by Location.Neighborhood

order by min(Year_Built);

select * from Oldest_Properties;
```

```
# min(Year_Built)     Neighborhood

1932                  Tanleytown

1947                  Capitol Hill

1979                  Du Pont Circle
```

- Delete:

delete one tuple or a set of tuples: from one table, from multiple tables.

Query: Delete the records of the studio apartments from Property, i.e the properties with 0 bedrooms

```
delete from Property

where Number_of_Bedrooms="0";

select Property_ID, Number_of_Bedrooms

from Property;
```

Before Query:

```
# Property_ID Number_of_Bedrooms

101     4

102     1

103     3

104     6

105     2

106     5

107     2

108     4

109     8

110     4

111     4
```

112	7
113	4
114	4
115	**0**
116	10

After Query:

# Property_ID Number_of_Bedrooms

101	4
102	1
103	3
104	6
105	2
106	5
107	2
108	4
109	8
110	4
111	4
112	7

113	4
114	4
116	10

Query: Delete the property that does not have a buyer or a seller associated with it

```
delete from Property

where Property_ID not in

(Select Property_ID

from Property_Client);
```

Before Query:

```
# Property_ID Number_of_Bedrooms

101	4

102	1

103	3

104	6

105	2

106	5

107	2

108	4

109	8
```

110	4
111	4
112	7
113	4
114	4
116	**10**

After Query:

# Property_ID	Number_of_Bedrooms
101	4
102	1
103	3
104	6
105	2
106	5
107	2
108	4
109	8
110	4
111	4

112	7
113	4
114	4

- Update:

update one tuple or a set of tuples: from one table, from multiple tables.

Query: Update the type of the property to 'Multi-family' home when the property_sqft is greater than 3500

```
update Property

set Property_Type= "Multi-Family"

where Property_SQFT>3500;
```

Property Before Query:

# Property_ID	Property_Type	Property_SQFT	Year_Built	Built_By	Number_of_Bedrooms	Number_of_Bathrooms	Number_of_Garages	Number_of_Stories
101	Single Family	2352	1939	Seller	4	3	1	2
102	Apartment	700	2021	Seller	1	1	0	1
103	Townhouse	3200	1975	Third Party	3	3	1	2
104	Single Family	4500	1989	Seller	6	3	2	2
105	Townhouse	1900	2001	Third Party	2	2	1	3
106	Single Family	4885	1979	Seller	5	4	2	3

107	Apartment	1200	2007	Third Party	2	2	0	1
108	Townhouse	2330	2009	Third Party	4	4	1	2
109	Single Family	8320	1947	Seller	8	9	4	3
110	Single Family	1352	1959	Third Party	4	3	1	2
111	Townhouse	2100	1990	Third Party	4	3	1	3
112	Single Family	4352	1992	Third Party	7	4	1	2
113	Single Family	1750	2022	Seller	4	3	1	2
114	Townhouse	2700	2000	Third Party	4	3	1	2

Property After Query:

# Property_ID	Property_Type	Property_SQFT	Year_Built	Built_By	Number_of_Bedrooms	Number_of_Bathrooms	Number_of_Garages	Number_of_Stories
101	Single Family	2352	1939	Seller	4	3	1	2
102	Apartment	700	2021	Seller	1	1	0	1
103	Townhouse	3200	1975	Third Party	3	3	1	2
104	**Multi-Family**	**4500**	**1989**	**Seller**	**6**	**3**	**2**	**2**
105	Townhouse	1900	2001	Third Party	2	2	1	3
106	**Multi-Family**	**4885**	**1979**	**Seller**	**5**	**4**	**2**	**3**
107	Apartment	1200	2007	Third Party	2	2	0	1

108	Townhouse	2330	2009	Third Party	4	4	1	2
109	**Multi-Family**	**8320**	**1947**	**Seller**	**8**	**9**	**4**	**3**
110	Single Family	1352	1959	Third Party	4	3	1	2
111	Townhouse	2100	1990	Third Party	4	3	1	3
112	**Multi-Family**	**4352**	**1992**	**Third Party**	**7**	**4**	**1**	**2**
113	Single Family	1750	2022	Seller	4	3	1	2
114	Townhouse	2700	2000	Third Party	4	3	1	2

Query: Update the client State_Region to DMV for all clients from DC, MD and VA

```
update Client

set State_Region="DMV"

where State_Region like '%MD%' or

State_Region like '%VA%' or

State_Region like '%DC%';
```

Client Before Query:

# Client_ID	State_Region
301	VA
302	VA
303	VA
304	VA

305	VA
306	WV
307	PA
308	LA
309	CA
310	MD
311	CA
312	AR
313	VA
314	VA
315	CT
316	DC
317	AZ
318	MD
319	VA
320	MD

Client After Query:

# Client_ID	State_Region
301	**DMV**

302	**DMV**
303	**DMV**
304	**DMV**
305	**DMV**
306	WV
307	PA
308	LA
309	CA
310	**DMV**
311	CA
312	AR
313	**DMV**
314	**DMV**
315	CT
316	**DMV**
317	AZ
318	**DMV**
319	**DMV**
320	**DMV**

Query: Update the 'Role_of_Client" to "Seller and Builder" for all those clients who sold properties that were built by them

```
Update Property_Client

set Role_of_Client ="Seller and Builder"

where Role_of_Client ="Seller" and

Property_ID in

(select Property_ID

from Property

where Built_By="Seller");
```

Property_Client Before Query:

```
# Client_ID    Property_ID    Role_of_Client

301      101      Seller

302      101      Buyer

303      102      Seller

304      102      Buyer

305      103      Seller

306      103      Buyer

307      104      Seller

308      104      Buyer
```

309	105	Seller
310	112	Buyer
311	106	Seller
312	113	Buyer
313	107	Seller
314	114	Buyer
315	108	Seller
316	115	Buyer
317	109	Seller
318	115	Seller
319	110	Seller
320	111	Seller

Property_Client After Query:

# Client_ID	Property_ID	Role_of_Client
301	101	Seller and Builder
302	101	Buyer
303	102	Seller and Builder
304	102	Buyer
305	103	Seller

306	103	Buyer
307	104	Seller and Builder
308	104	Buyer
309	105	Seller
310	112	Buyer
311	106	Seller and Builder
312	113	Buyer
313	107	Seller
314	114	Buyer
315	108	Seller
316	115	Buyer
317	109	Seller and Builder
318	115	Seller
319	110	Seller
320	111	Seller

- Create View:

based on one relation and more than one relation:

Query: Create a view with employees who manage more than the average number of properties or agents who are dual agents or both

```
select avg(Total_Properties_Managed)

from Agent;

# avg(Total_Properties_Managed)

30.384615384615383

create view HighPerformingAgents as

select Agent_ID, First_Name, Last_Name, Total_Properties_managed, Type_of_Agent

from Agent

where Total_Properties_Managed>30 or Type_of_Agent="Dual Agent";

# Agent_ID    First_Name    Last_Name    Total_Properties_managed    Type_of_Agent

1001          Sarah         Adams        39      Buyers Agent

1003          Nancy         Wheeler      40      Buyers Agent

1004          Mike          Wheeler      83      Sellers Agent

1005          Ali           Meer         22      Dual Agent

1006          Will          Byers        12      Dual Agent

1007          Jane          johnson      62      Dual Agent

1009          Nick          Miller       8       Dual Agent

1012          JC            Tag          48      Dual Agent
```

Query: Create a view showing Client IDs, their respective roles (i.e whether the client is a buyer, seller or seller and builder), their Property_ID and the Squarefeet of their properties

```
create view client_SQFT as

select Client_ID,Role_of_Client, Property_SQFT, Property.Property_ID

from Property, Property_Client

where Property.Property_ID= Property_Client.Property_ID;
```

```
# Client_ID   Role_of_Client          Property_SQFT          Property_ID

301     Seller and Builder              2352    101

302     Buyer                           2352    101

303     Seller and Builder              700     102

304     Buyer                           700     102

305     Seller                          3200    103

306     Buyer                           3200    103

307     Seller and Builder              4500    104

308     Buyer                           4500    104

309     Seller                          1900    105

310     Buyer                           4352    112

311     Seller and Builder              4885    106

312     Buyer                           1750    113
```

313	Seller	1200	107
314	Buyer	2700	114
315	Seller	2330	108
317	Seller and Builder	8320	109
319	Seller	1352	110
320	Seller	2100	111

- operate on View (i.e., select, insert, delete, update,..)

Query: Select the High performing agents who bought and sold properties in the Du Pont Neigborhood.

```
select *
from HighPerformingAgents
where Agent_ID in
(select Agent_ID
from Agent_Properties
where Property_ID in
(Select Property_ID
from Location
where Neighborhood="Du Pont Circle"));
```

```
# Agent_ID    First_Name    Last_Name    Total_Properties_managed    Type_of_Agent
```

1001	Sarah	Adams	39	Buyers Agent
1004	Mike	Wheeler	83	Sellers Agent
1005	Ali	Meer	22	Dual Agent
1006	Will	Byers	12	Dual Agent

Query: Delete the high performing agents who bought or sold less than 10 properties from HighPerformingAgents

delete from HighPerformingAgents

where total_Properties_managed<10;

HighPerformingAgents After Query:

# Agent_ID	First_Name	Last_Name	Total_Properties_managed	Type_of_Agent
1001	Sarah	Adams	39	Buyers Agent
1003	Nancy	Wheeler	40	Buyers Agent
1004	Mike	Wheeler	83	Sellers Agent
1005	Ali	Meer	22	Dual Agent
1006	Will	Byers	12	Dual Agent
1007	Jane	johnson	62	Dual Agent
1012	JC	Tag	48	Dual Agent

Query: Insert Agents ‘Asim Javed’ and ‘Asma Qureshi’ to HighPerformingAgents

insert into Agent values('1014','2002','Asim','Javed','Sellers Agent','55','11d33b@kwr.com','93339669');

```
insert into Agent values('1015','2001','Asma','Qureshi','Dual
Agent','89','11d3333b@kwr.com','93355555');
```

HighPerformingAgents After Query:

```
# Agent_ID	First_Name	Last_Name	Total_Properties_managed	Type_of_Agent

1001	Sarah	Adams	39	Buyers Agent

1003	Nancy	Wheeler	40	Buyers Agent

1004	Mike	Wheeler	83	Sellers Agent

1005	Ali	Meer	22	Dual Agent

1006	Will	Byers	12	Dual Agent

1007	Jane	johnson	62	Dual Agent

1012	JC	Tag	48	Dual Agent

1014	Asim	Javed	55	Sellers Agent

1015	Asma	Qureshi	89	Dual Agent
```

Query: Increase the 'Total Properties Managed' by 2 for all the Dual Agents in HighPerformingAgents

```
Update HighPerformingAgents

set Total_Properties_managed=2+Total_Properties_managed

where Type_of_Agent="Dual Agent";
```

HighPerformingAgents After Query:

```
# Agent_ID	First_Name	Last_Name	Total_Properties_managed	Type_of_Agent
```

```
1001	Sarah	Adams	39	Buyers Agent

1003	Nancy	Wheeler	40	Buyers Agent

1004	Mike	Wheeler	83	Sellers Agent

1005	Ali	Meer	24	Dual Agent

1006	Will	Byers	14	Dual Agent

1007	Jane	johnson	64	Dual Agent

1012	JC	Tag	50	Dual Agent

1014	Asim	Javed	55	Sellers Agent

1015	Asma	Qureshi	91	Dual Agent
```

Query: Show the average squarefeet bought/sold by buyer, sellers, and seller and builers grouped by Role_of_Client

```
select Role_of_Client, avg(Property_SQFT)

from client_SQFT

group by Role_of_Client;
```

```
# Role_of_Client	avg(Property_SQFT)

Seller and Builder	4151.4

Buyer	2793.4285714285716

Seller	2013.6666666666667
```

Query: insert data into the view Client_SQFT of a property with ID 130 sold by Client 321

```
insert into Property_Client values('321','130', 'Seller');

insert into Property values('130','Single Family','2323',1939,'Seller','4','3','1','2');
```

Client_SQFT before Query:

```
# Client_ID    Role_of_Client         Property_SQFT        Property_ID

301     Seller and Builder     2352    101

302     Buyer                  2352    101

303     Seller and Builder     700     102

304     Buyer                  700     102

305     Seller                 3200    103

306     Buyer                  3200    103

307     Seller and Builder     4500    104

308     Buyer                  4500    104

309     Seller                 1900    105

310     Buyer                  4352    112

311     Seller and Builder     4885    106

312     Buyer                  1750    113

313     Seller                 1200    107

314     Buyer                  2700    114

315     Seller                 2330    108
```

```
317	Seller and Builder	8320	109

319	Seller			1352	110

320	Seller				2100	111
```

Client_SQFT After Query:

```
# Client_ID	Role_of_Client		Property_SQFT		Property_ID

301	Seller and Builder	2352	101

302	Buyer			2352	101

303	Seller and Builder	700	102

304	Buyer			700	102

305	Seller			3200	103

306	Buyer			3200	103

307	Seller and Builder	4500	104

308	Buyer			4500	104

309	Seller			1900	105

310	Buyer			4352	112

311	Seller and Builder	4885	106

312	Buyer			1750	113

313	Seller			1200	107

314	Buyer			2700	114
```

315	Seller	2330	108
317	Seller and Builder	8320	109
319	Seller	1352	110
320	Seller	2100	111
321	Seller	2323	130

Query: Reduce the Squarefeet of all Properties on the street Connecticut Ave NW by 50 Squarefeet.

```
Update client_SQFT

set Property_SQFT=Property_SQFT-50

WHERE Property_ID in

(select Property_ID

from Location

where Street_Address like '%Connecticut Ave NW%');
```

Client_SQFT Before Query:

# Client_ID	Role_of_Client	Property_SQFT	Property_ID
301	Seller and Builder	2352	101
302	Buyer	2352	101
303	Seller and Builder	700	102
304	Buyer	700	102

305	Seller	3200	103
306	Buyer	3200	103
307	Seller and Builder	4500	104
308	Buyer	4500	104
309	Seller	1900	105
310	Buyer	4352	112
311	Seller and Builder	4885	106
312	Buyer	1750	113
313	Seller	1200	107
314	Buyer	2700	114
315	Seller	2330	108
317	Seller and Builder	8320	109
319	Seller	1352	110
320	Seller	2100	111
321	Seller	2323	130

Client_SQFT After Query:

```
# Client_ID     Role_of_Client          Property_SQFT          Property_ID

301       Seller and Builder       2352    101
```

302	Buyer	2352	101
303	Seller and Builder	700	102
304	Buyer	700	102
305	**Seller**	**3150**	**103**
306	**Buyer**	**3150**	**103**
307	Seller and Builder	4500	104
308	Buyer	4500	104
309	**Seller**	**1850**	**105**
310	Buyer	4352	112
311	Seller and Builder	4885	106
312	Buyer	1750	113
313	Seller	1200	107
314	**Buyer**	**2650**	**114**
315	**Seller**	**2280**	**108**
317	**Seller and Builder**	**8270**	**109**
319	**Seller**	**1302**	**110**
320	Seller	2100	111
321	Seller	2323	130

Query: Delete all sellers who built their properties from Client_SQFT

```
delete from Property_Client

where Role_of_Client="Seller and Builder";
```

Client_SQFT before Query:

```
# Client_ID	Role_of_Client		Property_SQFT		Property_ID

301	Seller and Builder	2352	101

302	Buyer	2352	101

303	Seller and Builder	700	102

304	Buyer	700	102

305	Seller	3150	103

306	Buyer	3150	103

307	Seller and Builder	4500	104

308	Buyer	4500	104

309	Seller	1850	105

310	Buyer	4352	112

311	Seller and Builder	4885	106

312	Buyer	1750	113

313	Seller	1200	107

314	Buyer	2650	114

315	Seller	2280	108
```

```
317	Seller and Builder	8270	109

319	Seller	1302	110

320	Seller	2100	111

321	Seller	2323	130
```

Client_SQFT After Query:

```
# Client_ID	Role_of_Client	Property_SQFT	Property_ID

302	Buyer	2352	101

304	Buyer	700	102

305	Seller	3150	103

306	Buyer	3150	103

308	Buyer	4500	104

309	Seller	1850	105

310	Buyer	4352	112

312	Buyer	1750	113

313	Seller	1200	107

314	Buyer	2650	114

315	Seller	2280	108

319	Seller	1302	110

320	Seller	2100	111
```

```
321     Seller  2323    130
```

B) Also, create at least 4 different practical/useful triggers (written in MySQL) for your database to perform the following tasks:

Show how these triggers are used and what these triggers produce (outputs).

<u>- enforcing referential integrity</u>

Query: Referrential integrity : to add a child=PROPERTY, the parent=Client must exist. The client (who is the buyer or seller of the property) must exist before the property they buy or sell is added into the database.

```
Delimiter //

create trigger addproperty before insert on Property

for each row

begin

declare temp Int;

set temp=0;

        select count(*) into temp from Property_Client where Property_ID=NEW.Property_ID;

        if temp=0 then

        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT ="Client is not in the system";

        end if;

end; //

delimiter ;
```

```
#Activation

insert into Property value('117','Single Family','3332',1939,'Seller','5','4','2','2');
```

Output:

```
22:37:42	insert into Property value('117','Single Family','3332',1939,'Seller','5','4','2','2')
	Error Code: 1644. Client is not in the system	0.0033 sec
```

When we try to add property 117, we get an error which says that the corresponding client does not exist.

- enforcing attribute domain constraints

```
create table DomainPropertyType(Property_Type varchar(30));

insert into DomainPropertyType values("Townhouse");

insert into DomainPropertyType values("Single Family");

insert into DomainPropertyType values("Apartment");

select * from DomainPropertyType;

# Property_Type

Townhouse

Single Family

Apartment



Delimiter //

create trigger domain_PropertyType_checking before insert on Property
```

```
for each row

begin

declare temp Int;

set temp=0;

            select count(*) into temp from DomainPropertyType where
Property_Type=new.Property_Type;

            if temp=0 then

    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = ' Invaid Type ';

    end if;

    end; //

delimiter ;

#Activation

insert into Property values('1111','Family','2352',1939,'Seller','4','3','1','2');
```

Output:

```
22:44:18          insert into Property values('1111','Family','2352',1939,'Seller','4','3','1','2')   Error
Code: 1644.  Invaid Type      0.0029 sec
```

- creating database log

```
Create table MyLog (message varchar(70));

Delimiter //

create trigger add_agent after insert on Agent
```

```
for each row

begin

insert into Mylog values(concat('Agent ',new.Last_Name,' has been added by ',current_user(), '
on ',current_date()));

end//

delimiter ;

#Activation

insert into Agent values('1420','2003','Michael','Jackson','Buyers
Agent','39','sasjs@kwr.com','9112233669');

select * from Agent;

# Agent_ID    Branch_ID    First_Name    Last_Name    Type_of_Agent
        Total_Properties_managed    Email   Phone_Number

1001    2001    Sarah   Adams Buyers Agent  39      sa@kwr.com   9374446669

1002    2002    Jim     HopperSellers Agent  29      jh@kwr.com   1235556669

1003    2003    Nancy  Wheeler         Buyers Agent  40      nw@kwr.com 9987776669

1004    2003    Mike    Wheeler         Sellers Agent  83      mw@kwr.com1239996669

1005    2002    Ali     Meer    Dual Agent     24      am@kwr.com 3334446669

1006    2001    Will    Byers   Dual Agent     14      wb@kwr.com 2984446669

1007    2003    Jane    johnson         Dual Agent     64      jj@kwr.com    1239446669

1008    2002    Dustin Bishop Sellers Agent  9       db@kwr.com  9834446669
```

1010	2001	Winston	Schmidt	Buyers Agent	3	ws@kwr.com	1094446669
1011	2002	Dustin	Crew	Sellers Agent	27	11db@kwr.com	98347799669
1012	2002	JC	Tag	Dual Agent	50	12wnm@kwr.com	128899000
1013	2001	Macy	Wills	Buyers Agent	13	wwws@kwr.com	10944489789
1014	2002	Asim	Javed	Sellers Agent	55	11d33b@kwr.com	93339669
1015	2001	Asma	Qureshi	Dual Agent	91	11d3333b@kwr.com	93355555
1420	**2003**	**Michael**	**Jackson**	**Buyers Agent**	**39**	**sasjs@kwr.com**	**9112233669**

```
Select * from Mylog;
```

```
# message
Agent Jackson has been added by root@localhost on 2022-06-19
```

<u>- gathering statistics</u>

```
create table Property_summary(Property_Type varchar(15),minSQFT double, maxSQFT double, avgSQFT double);

Delimiter //

create trigger Property_insert after insert on Property

for each row

begin

delete from Property_summary;

insert Property_summary
```

```
select Property_Type, min(Property_SQFT),max(Property_SQFT),avg(Property_SQFT) from
Property group by Property_Type;

end//

delimiter ;
```

```
#Activation

insert into Property value('117','Single Family','3332',1939,'Seller','5','4','2','2');


# Property_Type     minSQFT     maxSQFT     avgSQFT

Single Family  1302    3332    2211.8

Apartment      700     1200    950

Townhouse      1850    3150    2406

Multi-Family   4352    8270    5501.75

insert into Property value('118','Apartment','4000',1939,'Seller','5','4','2','2');

# Property_Type     minSQFT     maxSQFT     avgSQFT

Single Family  1302    3332    2211.8

Apartment      700     4000    1966.6666666666667

Townhouse      1850    3150    2406

Multi-Family   4352    8270    5501.75
```