Frederick McGovern

ICT 4405 Database Design & Programming

Email – [Frederick\_McGovern@msn.com](mailto:Frederick_McGovern@msn.com)

Phone – 508-846-2857

Assignment 3

Whenever a developer has been tasked with creating a new database, they would be well advised to follow the steps outlined in the Database Life Cycle (DBLC). The DBLC consists of six phases with multiple actions required in each phase. The first phase of the DBLC is known as the Database initial study phase. During this phase, the developer should determine, through communication with the business users, the exact nature of the business, the problem that is being addresses by the creation of a database, the data that the company utilizes and the business rules and conditions that apply to such data. Additionally, the scope of the project must be determined. Following this phase, the developer would engage in the database design, implement to database along with the initial loading of data, complete all necessary testing and put the database into operation. Once in operation, the database administrator would perform any maintenance on the database that is required.

The assignment that we are to engage in this week primarily focuses on the database design, implementation and testing phases of the DBLC. We are given an introduction to a fictionalized company known as ABC Real Estate in a problem statement. This statement is in fact a representation of the initial phases of the DBLC in that we are present with the information that the business deems critical to their success as well as several constraints that should be enacted upon the data that is collected. Utilizing this information, I will proceed through the creation of logical model utilizing Crow’s Foot notation in the Entity Relational Diagram. I will show through the use of dependency diagrams the normal forms of 1NF, 2NF and 3NF. Having completed these requirements I will then create the necessary SQL script to create the database, the resultant entity and views of the defined entities. Additionally, this script will allow for the initial population of the entities that are created.

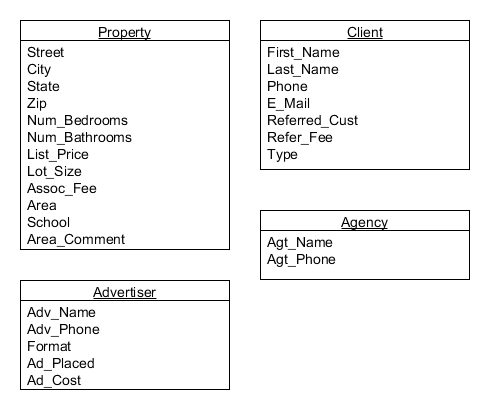
As indicated in the problem statement, property information is a core component of the database and should contain the following information – address, number of bedrooms and number of bathrooms, listing price, lot size and monthly association fee, area, schools in area and general comments about the area. It’s indicated that address should include the street, city, state and zip code for the property. Since these attributes relate to a specific object, that being a Property, the first entity I would create would be the Property entity with the attributes I’ve just listed.

The property is also used in conjunction when defining the advertiser utilized for a given property. In this case, the business needs to know what advertiser format was used, how much the ad cost and when the ad was placed. The name and phone number of the advertiser is also needed. Since the majority of the information relates to the advertiser, the second entity I would create would be then called Advertiser.

Additionally, the business requires that the seller and buyer information for each property be stored. This information should consist of a first name, last name, phone number and email address. Since the business offers a referral fee to clients that result is a sale, there should be a mechanism in place to identify the client they referred and whether or not a referral fee was paid to the referee. Again, there is a relationship to property but the majority of attributes are characteristics of a client object. Hence, my third entity would in fact be the Client entity.

Lastly, since the business can sell properties listed by other agencies, the listing agency for a specific property needs to be tracked. This information would include the contact information of a phone number as well as the name of the agency. Since these are attributes specific to the object agency, my last entity that I am creating prior to normalization is Agency.

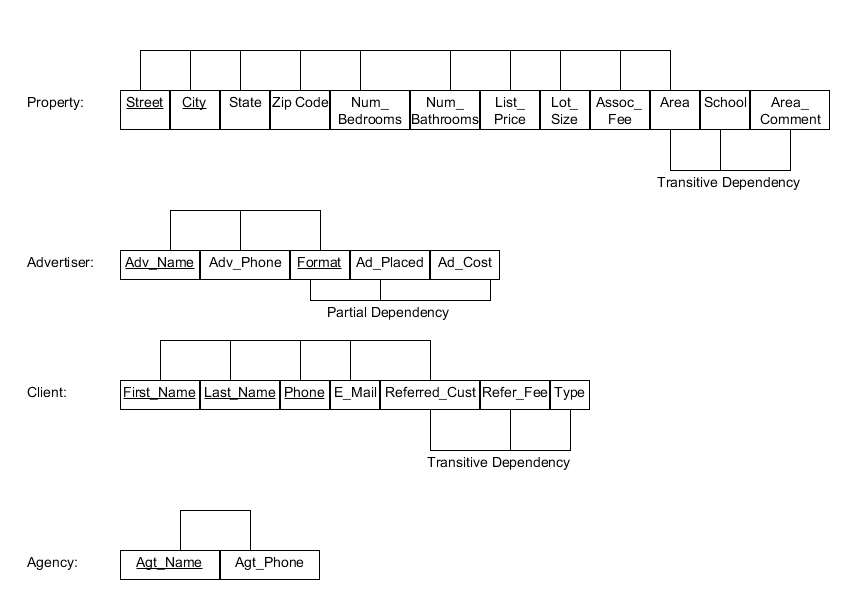
Using proper naming conventions, my initial entities and attributes would be as follows:



In order to convert an entity to first normal form one must first eliminate any repeating groups. In order to do just that, I first created an excel file that contained the four entities in question along with each attribute. Each worksheet represents a specific entity. I then populated each entity with raw data so as to be able to more clearly identify the repeating groups and dependencies. The attached excel file is called ABC Real Estate Raw Data.xls.

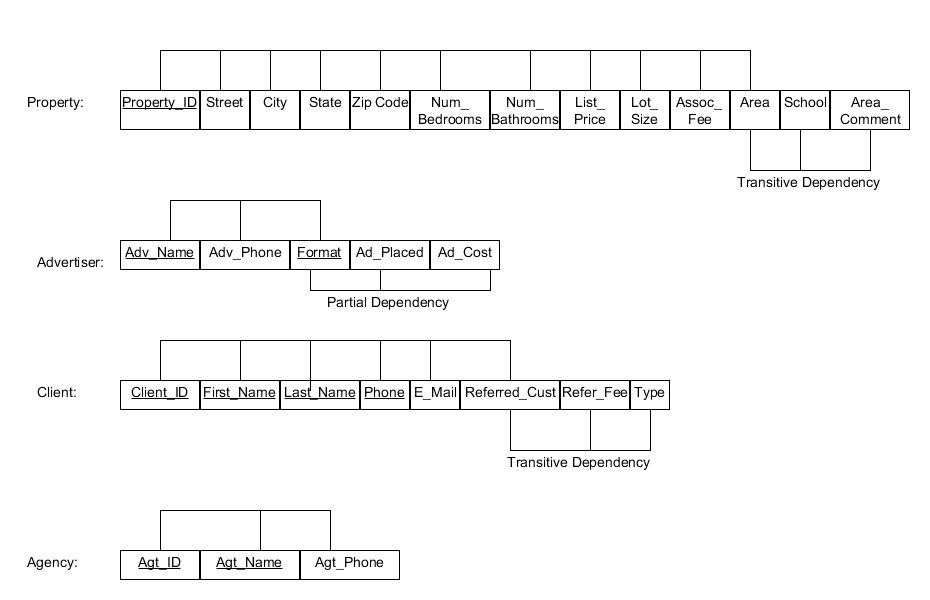
A 1NF table must contain an atomic value for each column. Thus for a table to be in 1NF, we must eliminate repeating groups. To eliminate the repeating groups, one must simply replace all null values for a given record with a data value. The attached excel file does this in four additional worksheets identified as EntityName\_1NF. As an example, looking at the Property Entity, since each property can be associated with multiple schools, only the first record contained all the data for each attribute. For the second and third schools the attributes were null. To get in 1NF, I started by replacing the null values with the corresponding street, city, state etc. The next step in obtaining 1NF is to identify the primary key and all dependencies.

Looking at the various entities, we can create the dependency diagrams and identify the primary keys as follows:



The primary key for each entity is identified by the attribute being underlined. Since all key attributes are defined, repeating groups have been eliminated and all attributes are dependent on the primary key the tables are now in 1NF. However, most of the existing tables are utilizing a composite key. For example, in the propety table, you can’t utilize street only as the primary key since the same street may exist in another city. Likewise in client, the first and last name alone does not guarentee a uniqueness but by adding phone number we are more assured of a unique record. This last example does introduce an issue. Since a primary key is unique it cannot contain any attribute that contains a null value. Based on the existing design, this would require that a phone number be entered for each client. Due to the complexity of utilizing composite keys, I am replacing them with surrogate keys resulting in the following:

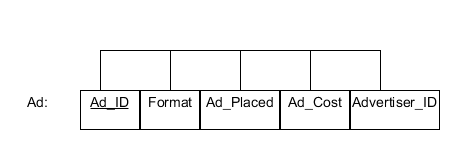
1NF Form:



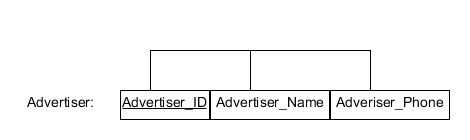
The addition of a surrogate key ensure uniqueness since the client, advertiser and agency all have a relationship to the property id from the Property entity and would be included in those entities as a foreign key.

2NF Form:

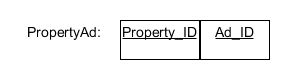
For a table to be in 2NF, we must first eliminate any partial dependencies from the existing tables by breaking out the associated components into new tables along with a copy of the part of the primary key that the attributes are dependant on. In my example, I have a partial dependencies on the Advertiser table. To eliminate that dependency we must move the determinant portion of the primary key to a new entity along with the attributes that depending on that determinant. In this case, I’ve created a new entity called Ad as follows:



This results in a change to the Advertiser table as follows:

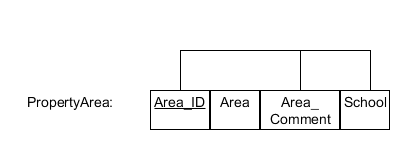


Since each property can have multiple advertising, I created a new entity called PropertyAd which acts as a bridge table between the Property table and the Ad table:

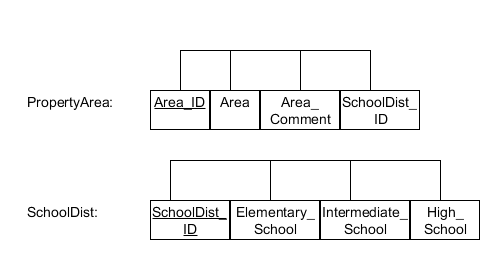


Since all of the current structures are in 1NF and all partial dependencies have been eliminated then all my structures are now in 2NF.

Third normal form requires the developer to remove any transitive dependencies from the tables that are in 2NF. This is accomplished by creating a new table to eliminate the dependency utilizing the determinant as the primary key for the new table. Looking at the Property entity, we can see that the determinant of Area is all that is needed to determine the Area\_Description and School so we can break out these atrributes into there own table called PropertyArea.

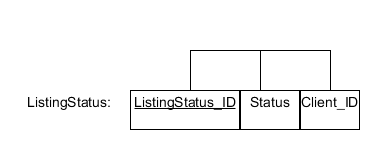


This still poses an issue as can be seen by the raw data. The attribute school can contain up to three different values representing the elementary school, middle school and high school. Additionally, the data is very redundant in that a specific school can in more than one area. The solution to this problem is to create another entity called SchoolDist with a primary key of SchoolDist\_ID which would be added in place of School in the PropertyArea entity as follows:

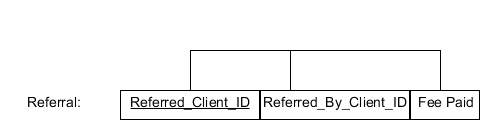


The attribute Area\_ID is then added to the Property table. This resolves the transitive dependency on the property table.

The remaining transitive dependency exists on the client table with regards to the referred customer attribute and the associate fee and type. Since a client can have sold and bought multiple properties we need a mechanism to identify whether or not a specific property has been sold or not. This would eliminate the need for a type attribute as we could just associate the property status with the assigned Client\_ID. We can therefore create a new entity called ListingStatus with the attributes Status and Client\_ID with Client\_ID being a foreign key to the Client table. Since both attributes are required for uniqueness we will also add a surrogate key. We initial can list the status as active along with the sellers Client\_ID. Once the property has a sales pending or is in fact sold the status can be updated and the Client\_ID will reflect the buyers Client\_ID. The field ListingStatus\_ID will need to be added to the Property Entity.

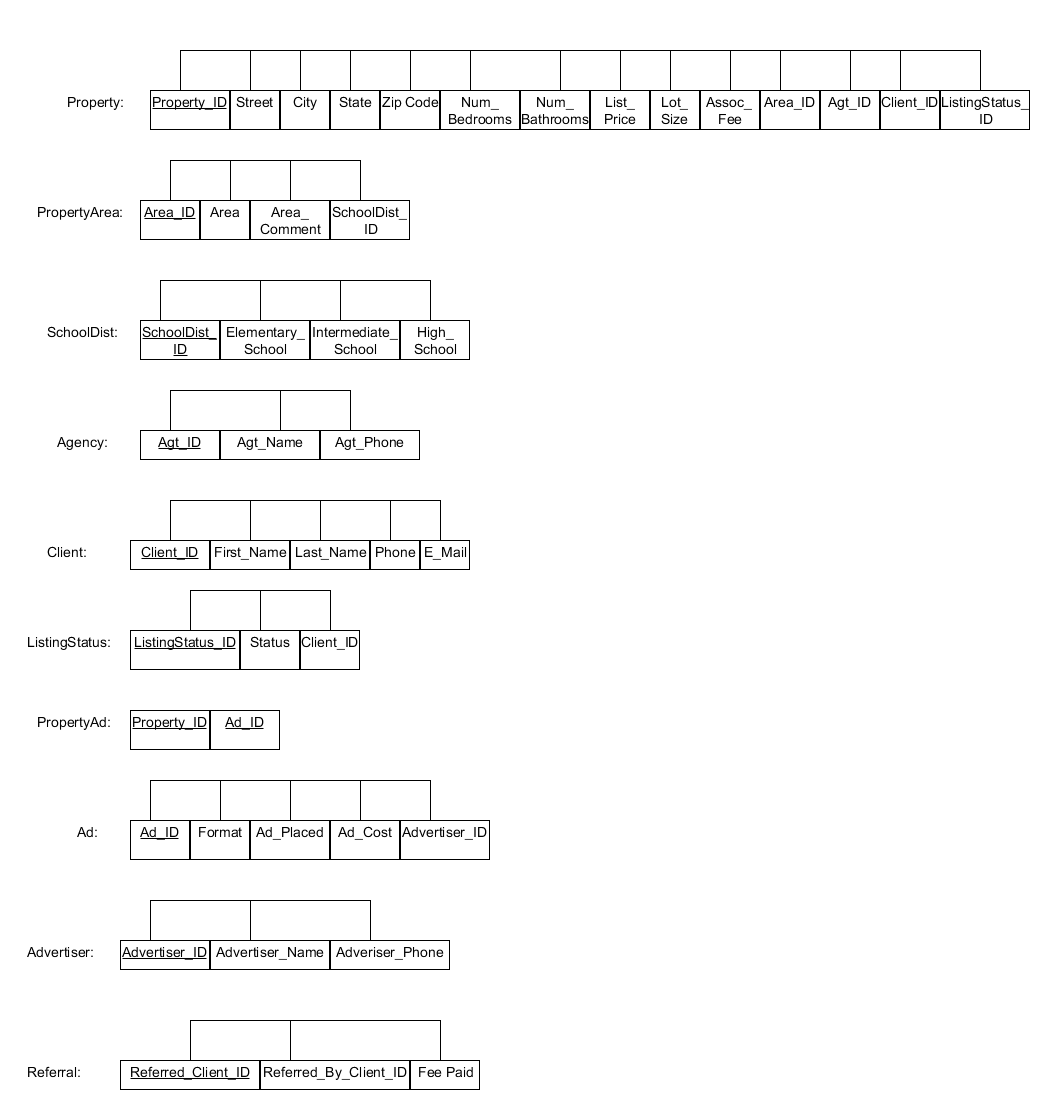


The remaining issue of referral\_cust and refer\_fee are a bit more complex. Since the referred\_client is in fact a client one could ascertain that there is a unary relationship here. However, since every client does not refer a client and since every client can refer more than one client, the best approach is to create a Referral entity which contains the Client\_ID of the referred client, the Client\_ID of the individual that made the referral and the fee paid for the referral. Since the referred client can only be referred once this ensures that we have uniqueness.

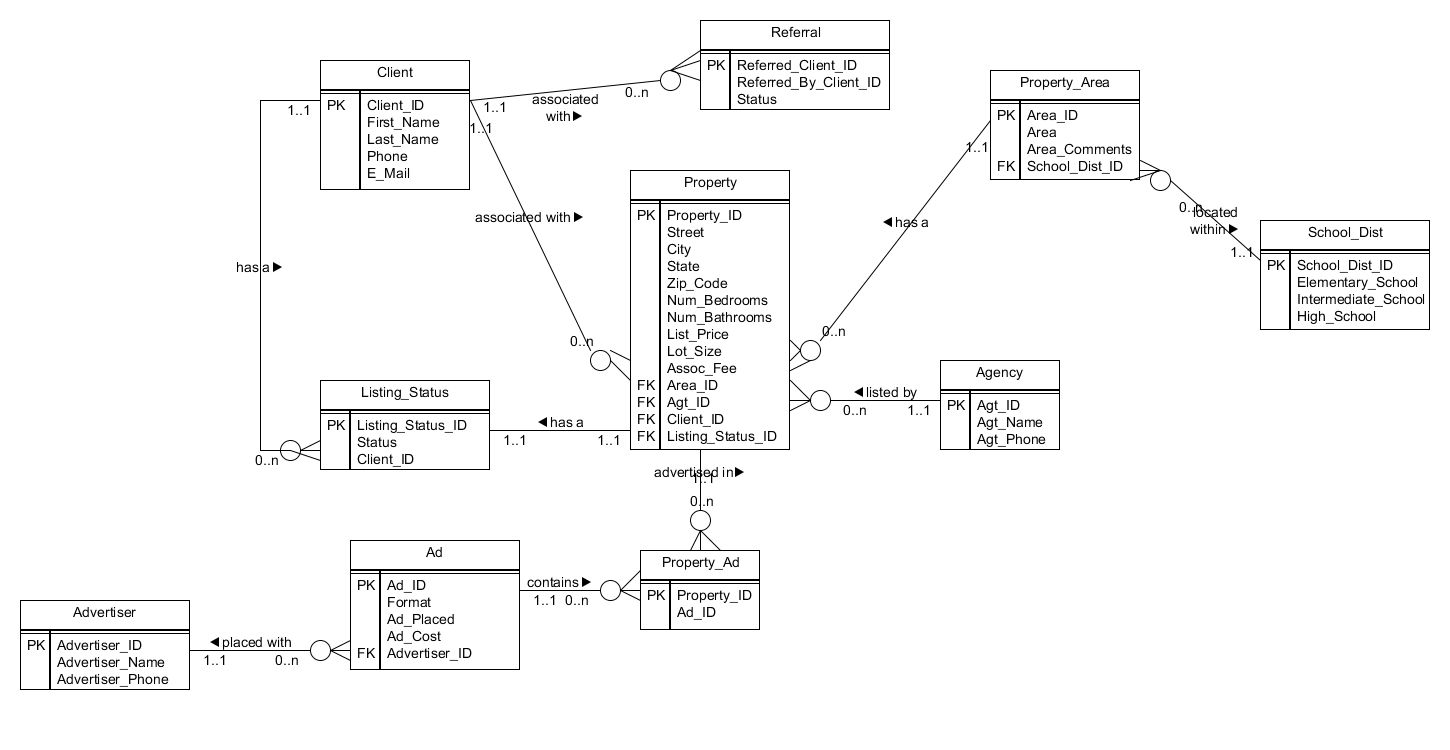


Since all transitive dependencies are now eliminated the tables are now in 3NF form as indicated below:

3NF Form:



Having completed the normalization of all tables to 3NF, the next step would be the produce the Entity Relational Diagram using Crow’s foot notation.



The following script will create and populate all tables in the database ABC Real Estate. Additionally, the script will define views on the tables so as to be able to do the required reporting.

/\*Check to see if the table we are about to create. If it does then drop any existing tables. \*/

DECLARE

c int;

BEGIN

--Check if table AD exists

select count(\*) into c from user\_tables where table\_name = 'AD';

if c = 1 then

execute immediate 'DROP TABLE AD';

end if;

--Check if table ADVERTISER exists

select count(\*) into c from user\_tables where table\_name = 'ADVERTISER';

if c = 1 then

execute immediate 'DROP TABLE ADVERTISER';

end if;

--Check if table AGENCY exists

select count(\*) into c from user\_tables where table\_name = 'AGENCY';

if c = 1 then

execute immediate 'DROP TABLE AGENCY';

end if;

--Check if table CLIENT exists

select count(\*) into c from user\_tables where table\_name = 'CLIENT';

if c = 1 then

execute immediate 'DROP TABLE CLIENT';

end if;

--Check if table LISTING\_STATUS exists

select count(\*) into c from user\_tables where table\_name = 'LISTING\_STATUS';

if c = 1 then

execute immediate 'DROP TABLE LISTING\_STATUS';

end if;

--Check if table PROPERTY exists

select count(\*) into c from user\_tables where table\_name = 'PROPERTY';

if c = 1 then

execute immediate 'DROP TABLE PROPERTY';

end if;

--Check if table PROPERTY\_AD exists

select count(\*) into c from user\_tables where table\_name = 'PROPERTY\_AD';

if c = 1 then

execute immediate 'DROP TABLE PROPERTY\_AD';

end if;

--Check if table PROPERTY\_AREA exists

select count(\*) into c from user\_tables where table\_name = 'PROPERTY\_AREA';

if c = 1 then

execute immediate 'DROP TABLE PROPERTY\_AREA';

end if;

--Check if table SCHOOL\_DIST exists

select count(\*) into c from user\_tables where table\_name = 'SCHOOL\_DIST';

if c = 1 then

execute immediate 'DROP TABLE SCHOOL\_DIST';

end if;

--Check if table REFERRAL exists

select count(\*) into c from user\_tables where table\_name = 'REFERRAL';

if c = 1 then

execute immediate 'DROP TABLE REFERRAL';

end if;

END;

/\*Now that we are sure that the table does not exist, we can execute a DDL script to create each table that we need in our database. \*/

CREATE TABLE AD

(

AD\_ID NUMBER NOT NULL,

FORMAT VARCHAR2(20 BYTE) NOT NULL,

AD\_PLACED DATE NOT NULL,

AD\_COST NUMBER NOT NULL,

ADVERTISER\_ID NUMBER NOT NULL,

CONSTRAINT AD\_PK PRIMARY KEY

(

AD\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

CREATE TABLE ADVERTISER

(

ADVERTISER\_ID NUMBER NOT NULL

, ADVERTISER\_NAME VARCHAR2(50 BYTE) NOT NULL

, ADVERTISER\_PHONE VARCHAR2(12 BYTE) NOT NULL

, CONSTRAINT ADVERTISER\_PK PRIMARY KEY

(

ADVERTISER\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

CREATE TABLE AGENCY

(

AGT\_ID NUMBER NOT NULL,

AGT\_NAME VARCHAR2(50 BYTE) NOT NULL,

AGT\_PHONE VARCHAR2(12 BYTE) NOT NULL,

CONSTRAINT AGENCY\_PK PRIMARY KEY

(

AGT\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

CREATE TABLE CLIENT

(

CLIENT\_ID NUMBER NOT NULL,

FIRST\_NAME VARCHAR2(20 BYTE) NOT NULL,

LAST\_NAME VARCHAR2(20 BYTE) NOT NULL,

PHONE VARCHAR2(12 BYTE) NOT NULL,

E\_MAIL VARCHAR2(50 BYTE) NOT NULL,

CONSTRAINT CLIENT\_PK PRIMARY KEY

(

CLIENT\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

CREATE TABLE LISTING\_STATUS

(

LISTING\_STATUS\_ID NUMBER NOT NULL,

STATUS VARCHAR2(20 BYTE) NOT NULL,

CLIENT\_ID NUMBER NOT NULL,

CONSTRAINT LISTINGSTATUS\_PK PRIMARY KEY

(

LISTING\_STATUS\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

CREATE TABLE PROPERTY

(

PROPERTY\_ID NUMBER NOT NULL,

STREET VARCHAR2(30 BYTE) NOT NULL,

CITY VARCHAR2(30 BYTE) NOT NULL,

STATE VARCHAR2(2 BYTE) NOT NULL,

ZIP\_CODE VARCHAR2(5 BYTE) NOT NULL,

NUM\_BEDROOMS NUMBER NOT NULL,

NUM\_BATHROOMS NUMBER NOT NULL,

LIST\_PRICE NUMBER NOT NULL,

LOT\_SIZE NUMBER,

ASSOC\_FEE NUMBER,

AREA\_ID NUMBER NOT NULL,

AGT\_ID NUMBER NOT NULL,

CLIENT\_ID NUMBER NOT NULL,

LISTING\_STATUS\_ID NUMBER NOT NULL,

CONSTRAINT PROPERTY\_PK PRIMARY KEY

(

PROPERTY\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

CREATE TABLE PROPERTY\_AD

(

PROPERTY\_ID NUMBER NOT NULL,

AD\_ID NUMBER NOT NULL,

CONSTRAINT PROPERTY\_AD\_PK PRIMARY KEY

(

PROPERTY\_ID, AD\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

CREATE TABLE PROPERTY\_AREA

(

AREA\_ID NUMBER NOT NULL,

AREA VARCHAR2(50 BYTE) NOT NULL,

AREA\_COMMENTS VARCHAR2(200 BYTE) NOT NULL,

SCHOOL\_DIST\_ID NUMBER NOT NULL,

CONSTRAINT PROPERTY\_AREA\_PK PRIMARY KEY

(

AREA\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

CREATE TABLE SCHOOL\_DIST

(

SCHOOL\_DIST\_ID NUMBER NOT NULL,

ELEMENTARY\_SCHOOL VARCHAR2(100 BYTE) NOT NULL,

INTERMEDIATE\_SCHOOL VARCHAR2(100 BYTE) NOT NULL,

HIGH\_SCHOOL VARCHAR2(100 BYTE) NOT NULL,

CONSTRAINT SCHOOLDIST\_PK PRIMARY KEY

(

SCHOOL\_DIST\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

CREATE TABLE REFERRAL

(

REFERRED\_CLIENT\_ID NUMBER NOT NULL,

REFERRED\_BY\_CLIENT\_ID NUMBER NOT NULL,

FEE\_PAID NUMBER NOT NULL,

CONSTRAINT REFERRAL\_PK PRIMARY KEY

(

REFERRED\_CLIENT\_ID

)

ENABLE

)

LOGGING

TABLESPACE "USERS"

PCTFREE 10

INITRANS 1

STORAGE

(

INITIAL 65536

NEXT 1048576

MINEXTENTS 1

MAXEXTENTS 2147483645

BUFFER\_POOL DEFAULT

);

/\* With the tables now created we can populate the data into the tables. The tables on the 1 side of the 1:M relationship must be populated first.\*/

--Insert data into table SCHOOL\_DIST

INSERT into SCHOOL\_DIST(SCHOOL\_DIST\_ID, ELEMENTARY\_SCHOOL, INTERMEDIATE\_SCHOOL, HIGH\_SCHOOL)

VALUES(1, 'Halifax Elementary School', 'Silver Lake Middle School', 'Silver Lake High School');

INSERT into SCHOOL\_DIST(SCHOOL\_DIST\_ID, ELEMENTARY\_SCHOOL, INTERMEDIATE\_SCHOOL, HIGH\_SCHOOL)

VALUES(2, 'Dennet Elemetary School', 'Silver Lake Middle School', 'Silver Lake High School');

INSERT into SCHOOL\_DIST(SCHOOL\_DIST\_ID, ELEMENTARY\_SCHOOL, INTERMEDIATE\_SCHOOL, HIGH\_SCHOOL)

VALUES(3, 'Pembroke Community Middle School', ' ', 'Pembroke High School');

INSERT into SCHOOL\_DIST(SCHOOL\_DIST\_ID, ELEMENTARY\_SCHOOL, INTERMEDIATE\_SCHOOL, HIGH\_SCHOOL)

VALUES(4, 'Nathaniel Morton Elementary School', 'Plymouth Community Intermediate School', 'Plymouth North High School');

INSERT into SCHOOL\_DIST(SCHOOL\_DIST\_ID, ELEMENTARY\_SCHOOL, INTERMEDIATE\_SCHOOL, HIGH\_SCHOOL)

VALUES(5, 'Manomet Elementary School', 'Plymouth Community Intermediate School', 'Plymouth South High School');

COMMIT;

--Insert data into table PROPERTY\_AREA

INSERT into PROPERTY\_AREA (AREA\_ID, AREA, AREA\_COMMENTS, SCHOOL\_DIST\_ID)

VALUES(1, 'South Halifax', 'Main Routes 58 and 106 pass through town', 1);

INSERT into PROPERTY\_AREA (AREA\_ID, AREA, AREA\_COMMENTS, SCHOOL\_DIST\_ID)

VALUES(2, 'Plympton', 'Main Routes 44 and 58 pass through town. New 10 acre park being created.', 2);

INSERT into PROPERTY\_AREA (AREA\_ID, AREA, AREA\_COMMENTS, SCHOOL\_DIST\_ID)

VALUES(3, 'East Pembroke', 'Close to beach and local shopping', 3);

INSERT into PROPERTY\_AREA (AREA\_ID, AREA, AREA\_COMMENTS, SCHOOL\_DIST\_ID)

VALUES(4, 'Robbins Wharf', 'Ocean front area next to 18 acres conservation land', 4);

INSERT into PROPERTY\_AREA (AREA\_ID, AREA, AREA\_COMMENTS, SCHOOL\_DIST\_ID)

VALUES(5, 'Highlands at Ocean Point', 'Close to White House Beach. Includes swimming pool and tennis courts', 5);

COMMIT;

--Insert data into table AGENCY

INSERT into AGENCY (AGT\_ID, AGT\_NAME, AGT\_PHONE)

VALUES(1, 'ABC Real Estate', '781-455-2000');

INSERT into AGENCY (AGT\_ID, AGT\_NAME, AGT\_PHONE)

VALUES(2, 'Select Real Estate', '508-588-4360');

INSERT into AGENCY (AGT\_ID, AGT\_NAME, AGT\_PHONE)

VALUES(3, 'Atlantic Coast Property Solutions', '617-838-8867');

INSERT into AGENCY (AGT\_ID, AGT\_NAME, AGT\_PHONE)

VALUES(4, 'Jack Conway Realtors', '888-754-4411');

INSERT into AGENCY (AGT\_ID, AGT\_NAME, AGT\_PHONE)

VALUES(5, 'River Farms Properties, LLC', '508-737-6742');

COMMIT;

--Insert data into table ADVERTISER

INSERT into ADVERTISER(ADVERTISER\_ID, ADVERTISER\_NAME, ADVERTISER\_PHONE)

VALUES(1, 'Boston Globe', '617-929-7011');

INSERT into ADVERTISER(ADVERTISER\_ID, ADVERTISER\_NAME, ADVERTISER\_PHONE)

VALUES(2, 'Patriot Ledger', '888-697-2737');

INSERT into ADVERTISER(ADVERTISER\_ID, ADVERTISER\_NAME, ADVERTISER\_PHONE)

VALUES(3, 'Old Colony Memorial', '508-927-6042');

INSERT into ADVERTISER(ADVERTISER\_ID, ADVERTISER\_NAME, ADVERTISER\_PHONE)

VALUES(4, 'WickedLocal.com', '800-624-7355');

INSERT into ADVERTISER(ADVERTISER\_ID, ADVERTISER\_NAME, ADVERTISER\_PHONE)

VALUES(5, 'BayState Realtor', '800-725-6272');

COMMIT;

--Insert data into table CLIENT

INSERT into CLIENT(CLIENT\_ID, FIRST\_NAME, LAST\_NAME, PHONE, E\_MAIL)

VALUES(1, 'George', 'Advorte', '781-658-2568', 'gregadv@sptarc.com');

INSERT into CLIENT(CLIENT\_ID, FIRST\_NAME, LAST\_NAME, PHONE, E\_MAIL)

VALUES(2, 'Frederick', 'McGovern', '781-293-2235', 'Frederick\_McGovern@msn.com');

INSERT into CLIENT(CLIENT\_ID, FIRST\_NAME, LAST\_NAME, PHONE, E\_MAIL)

VALUES(3, 'Lois','Knight', '508-941-6887', ' ');

INSERT into CLIENT(CLIENT\_ID, FIRST\_NAME, LAST\_NAME, PHONE, E\_MAIL)

VALUES(4, 'Henry', 'Adams', '508-941-2330', 'Hadams@comcast.net');

INSERT into CLIENT(CLIENT\_ID, FIRST\_NAME, LAST\_NAME, PHONE, E\_MAIL)

VALUES(5, 'James', 'Patterson', '508-946-6649', 'PatsFan456@verizon.net');

INSERT into CLIENT(CLIENT\_ID, FIRST\_NAME, LAST\_NAME, PHONE, E\_MAIL)

VALUES(6, 'Linda', 'Goodwin', '781-294-7883', 'LGoodwin@nstar.com');

COMMIT;

--Insert data into table LISTING\_STATUS

INSERT into LISTING\_STATUS(LISTING\_STATUS\_ID, STATUS, CLIENT\_ID)

VALUES(1, 'Active', 1);

INSERT into LISTING\_STATUS(LISTING\_STATUS\_ID, STATUS, CLIENT\_ID)

VALUES(2, 'Active', 2);

INSERT into LISTING\_STATUS(LISTING\_STATUS\_ID, STATUS, CLIENT\_ID)

VALUES(3, 'Sold', 2);

INSERT into LISTING\_STATUS(LISTING\_STATUS\_ID, STATUS, CLIENT\_ID)

VALUES(4, 'Active', 4);

INSERT into LISTING\_STATUS(LISTING\_STATUS\_ID, STATUS, CLIENT\_ID)

VALUES(5, 'Pending Sale', 3);

INSERT into LISTING\_STATUS(LISTING\_STATUS\_ID, STATUS, CLIENT\_ID)

VALUES(6, 'Active', 5);

COMMIT;

--Insert data into table AD

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(1, 'News Paper', '02 MAR 13', 150, 1);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(2, 'Web', '02 MAR 13', 45, 1);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(3, 'News Paper', '26 FEB 13', 150, 1);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(4, 'Web', '26 FEB 13', 45,1);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(5, 'Web', '26 FEB 13',45, 1);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(6, 'Web', '21 FEB 13',50, 1);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(7, 'News Paper', '05 MAR 13', 110, 3);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(8, 'News Paper', '01 MAR 13', 95, 3);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(9, 'Web', '28 FEB 13', 35, 4);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(10, 'Web', '27 FEB 13', 40, 4);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(11, 'Web', '02 MAR 13', 30, 4);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(12, 'Web', '27 FEB 13', 35, 4);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(13, 'Magazine', '04 MAR 13', 125, 5);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(14, 'Web', '04 MAR 13', 40, 5);

INSERT into AD(AD\_ID, FORMAT, AD\_PLACED, AD\_COST, ADVERTISER\_ID)

VALUES(15, 'Web', '28 FEB 13', 40, 5);

COMMIT;

--Insert data into table PROPERTY

INSERT into PROPERTY(PROPERTY\_ID, STREET, CITY, STATE, ZIP\_CODE, NUM\_BEDROOMS, NUM\_BATHROOMS, LIST\_PRICE,

LOT\_SIZE, ASSOC\_FEE, AREA\_ID, AGT\_ID, CLIENT\_ID, LISTING\_STATUS\_ID)

VALUES(1, '296 South Street', 'Halifax', 'MA', '02338', 3, 1, 225000, 0.92, NULL, 1, 1, 2, 2);

INSERT into PROPERTY(PROPERTY\_ID, STREET, CITY, STATE, ZIP\_CODE, NUM\_BEDROOMS, NUM\_BATHROOMS, LIST\_PRICE,

LOT\_SIZE, ASSOC\_FEE, AREA\_ID, AGT\_ID, CLIENT\_ID, LISTING\_STATUS\_ID)

VALUES(2, '10 Cross Street', 'Plympton', 'MA', '02367', 4, 2, 335000, 2.28, NULL, 2, 4, 5, 6);

INSERT into PROPERTY(PROPERTY\_ID, STREET, CITY, STATE, ZIP\_CODE, NUM\_BEDROOMS, NUM\_BATHROOMS, LIST\_PRICE,

LOT\_SIZE, ASSOC\_FEE, AREA\_ID, AGT\_ID, CLIENT\_ID, LISTING\_STATUS\_ID)

VALUES(3, '7 Cranberry Knoll Street #7', 'Plympton', 'MA', '02367', 2, 2, 259900, NULL, 283, 2, 1, 1, 1);

INSERT into PROPERTY(PROPERTY\_ID, STREET, CITY, STATE, ZIP\_CODE, NUM\_BEDROOMS, NUM\_BATHROOMS, LIST\_PRICE,

LOT\_SIZE, ASSOC\_FEE, AREA\_ID, AGT\_ID, CLIENT\_ID, LISTING\_STATUS\_ID)

VALUES(4, '24 Prince Way', 'Pembroke', 'MA', '02359', 4, 4, 519000, 0.93, NULL, 3, 4, 6, 3);

INSERT into PROPERTY(PROPERTY\_ID, STREET, CITY, STATE, ZIP\_CODE, NUM\_BEDROOMS, NUM\_BATHROOMS, LIST\_PRICE,

LOT\_SIZE, ASSOC\_FEE, AREA\_ID, AGT\_ID, CLIENT\_ID, LISTING\_STATUS\_ID)

VALUES(5, '60 Robbins Wharf', 'Plymouth', 'MA', '02360', 3, 3, 675000, 4, NULL, 4, 3, 3, 5);

INSERT into PROPERTY(PROPERTY\_ID, STREET, CITY, STATE, ZIP\_CODE, NUM\_BEDROOMS, NUM\_BATHROOMS, LIST\_PRICE,

LOT\_SIZE, ASSOC\_FEE, AREA\_ID, AGT\_ID, CLIENT\_ID, LISTING\_STATUS\_ID)

VALUES(6, '25 Highland Terrace Unit 2508', 'Plymouth', 'MA', '02360', 2, 1, 230000, NULL, 245, 5, 1, 4, 4);

COMMIT;

--Insert data into table PROPERTY\_AD

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(1, 7);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(1, 9);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(2, 5);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(2, 10);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(3, 8);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(3, 11);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(4, 3);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(4, 4);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(4, 12);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(5, 1);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(5, 2);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(5, 13);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(5, 14);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(6, 6);

INSERT into PROPERTY\_AD(PROPERTY\_ID, AD\_ID)

VALUES(6, 15);

COMMIT;

--Insert data into table REFERRALS

INSERT into REFERRAL(REFERRED\_CLIENT\_ID, REFERRED\_BY\_CLIENT\_ID, FEE\_PAID)

VALUES(2,3,250);

COMMIT;

/\* With the tables now populated we can modify the tables to add the foreign keys necessary to ensure data integrity.\*/

ALTER TABLE PROPERTY

add CONSTRAINT AREA\_ID\_FK

FOREIGN KEY (AREA\_ID)

REFERENCES PROPERTY\_AREA(AREA\_ID);

ALTER TABLE PROPERTY

add CONSTRAINT AGT\_ID\_FK

FOREIGN KEY (AGT\_ID)

REFERENCES AGENCY(AGT\_ID);

ALTER TABLE PROPERTY

add CONSTRAINT CLIENT\_ID\_FK1

FOREIGN KEY (CLIENT\_ID)

REFERENCES CLIENT(CLIENT\_ID);

ALTER TABLE PROPERTY

add CONSTRAINT LISTING\_STATUS\_ID\_FK

FOREIGN KEY (LISTING\_STATUS\_ID)

REFERENCES LISTING\_STATUS(LISTING\_STATUS\_ID);

ALTER TABLE PROPERTY\_AREA

add CONSTRAINT SCHOOL\_DIST\_ID\_FK

FOREIGN KEY (SCHOOL\_DIST\_ID)

REFERENCES SCHOOL\_DIST(SCHOOL\_DIST\_ID);

ALTER TABLE LISTING\_STATUS

add CONSTRAINT CLIENT\_ID\_FK2

FOREIGN KEY (CLIENT\_ID)

REFERENCES CLIENT(CLIENT\_ID);

ALTER TABLE AD

add CONSTRAINT ADVERTISER\_ID\_FK

FOREIGN KEY (ADVERTISER\_ID)

REFERENCES ADVERTISER(ADVERTISER\_ID);

ALTER TABLE REFERRAL

add CONSTRAINT CLIENT\_ID\_FK3

FOREIGN KEY (REFERRED\_BY\_CLIENT\_ID)

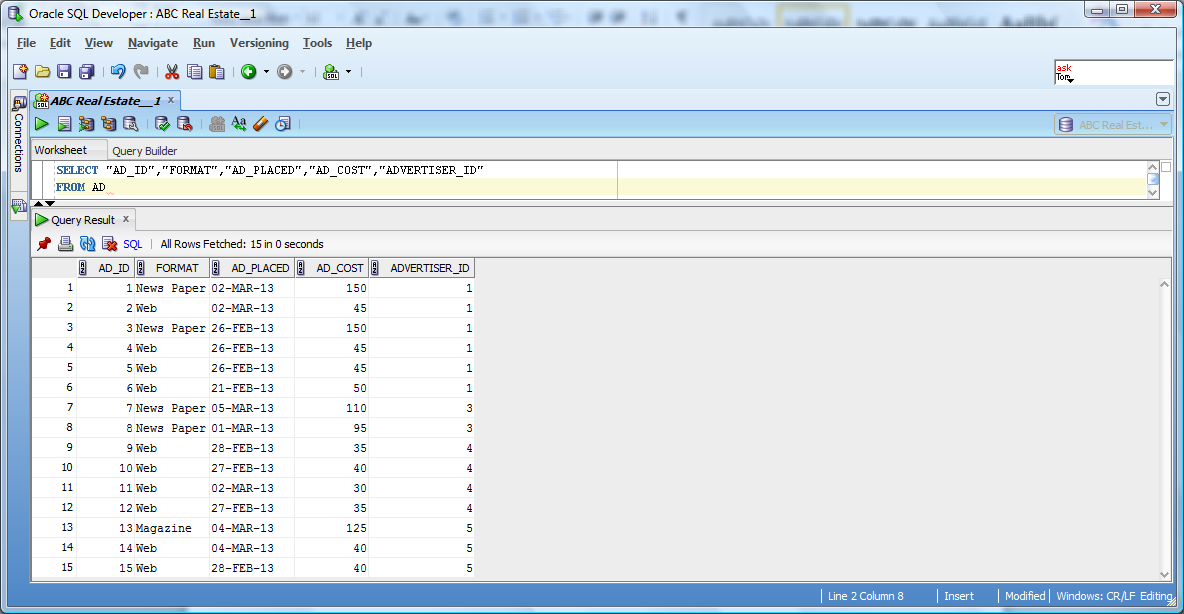
REFERENCES CLIENT(CLIENT\_ID);

/\* We can now create views to show the data in the various tables.\*/

--View for AD table – AD Report

SELECT "AD\_ID","FORMAT","AD\_PLACED","AD\_COST","ADVERTISER\_ID"

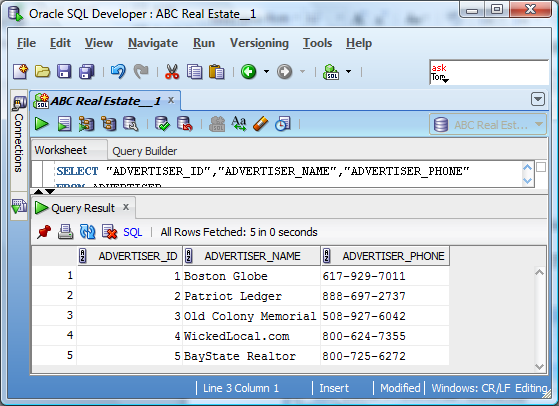
FROM AD



--View for ADVERTISER table – Advertiser Report

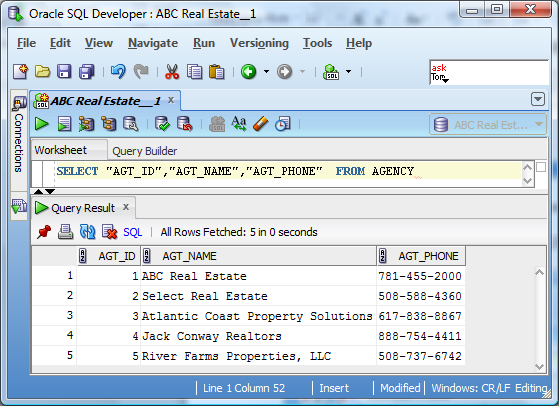
SELECT "ADVERTISER\_ID","ADVERTISER\_NAME","ADVERTISER\_PHONE"

FROM ADVERTISER



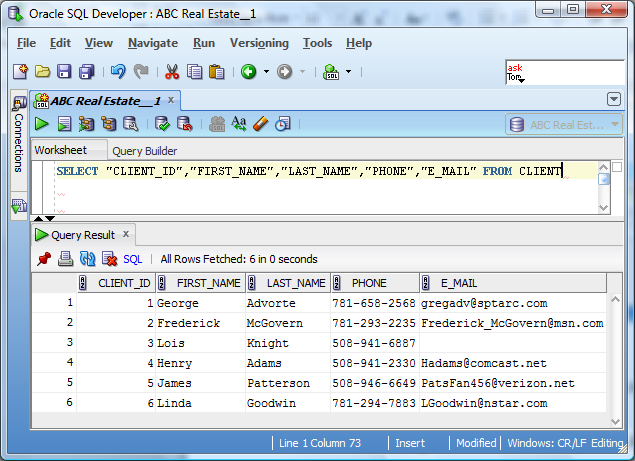
--View for AGENCY table – Agency Report

SELECT "AGT\_ID","AGT\_NAME","AGT\_PHONE" FROM AGENCY



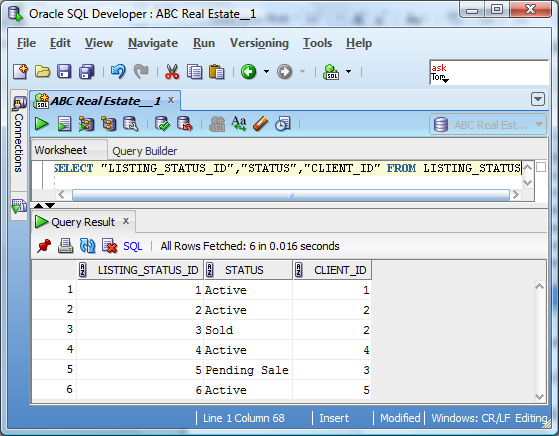
--View for CLIENT table – Client Report

SELECT "CLIENT\_ID","FIRST\_NAME","LAST\_NAME","PHONE","E\_MAIL" FROM CLIENT



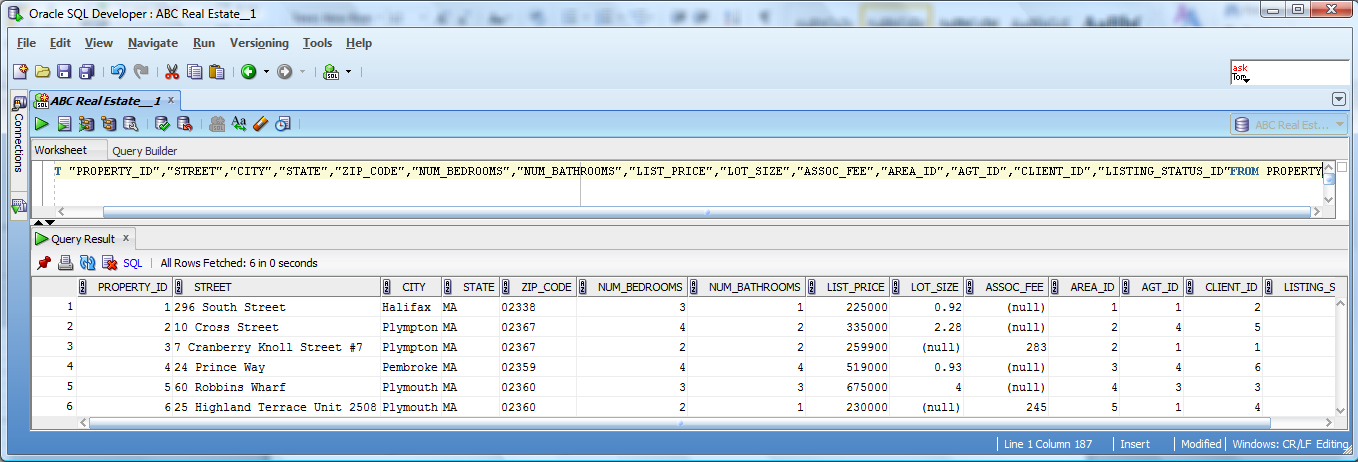
--View for LISTING\_STATUS table – Listing Status Report

SELECT "LISTING\_STATUS\_ID","STATUS","CLIENT\_ID" FROM LISTING\_STATUS



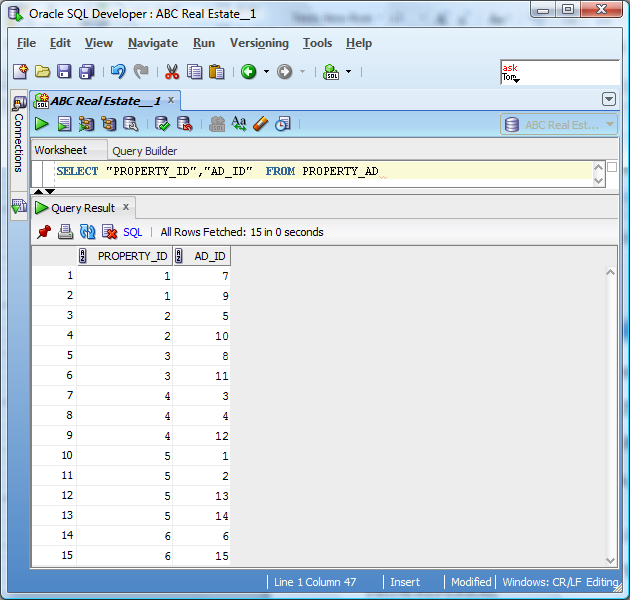
--View for PROPERTY table – Property Report

SELECT "PROPERTY\_ID","STREET","CITY","STATE","ZIP\_CODE","NUM\_BEDROOMS","NUM\_BATHROOMS","LIST\_PRICE","LOT\_SIZE","ASSOC\_FEE","AREA\_ID","AGT\_ID","CLIENT\_ID","LISTING\_STATUS\_ID"FROM PROPERTY



--View for PROPERTY\_AD table – Property Ad Report

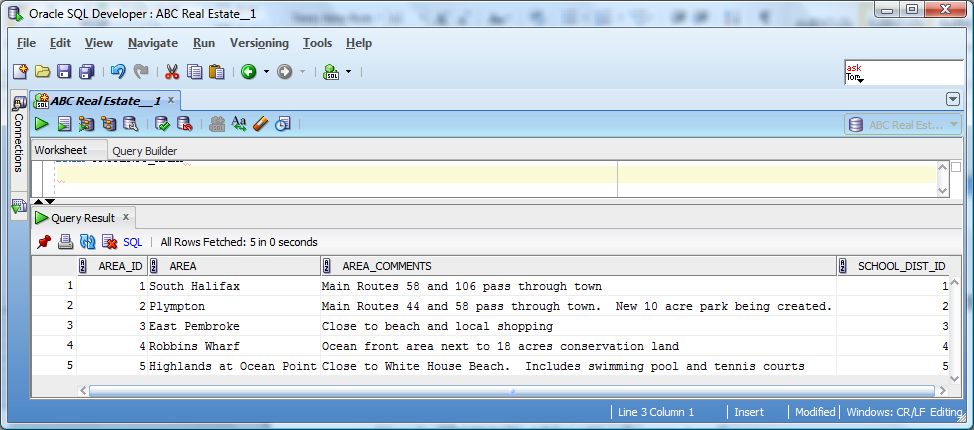
SELECT "PROPERTY\_ID","AD\_ID" FROM PROPERTY\_AD



--View for PROPERTY\_AREA table – Property Area Report

SELECT "AREA\_ID","AREA","AREA\_COMMENTS","SCHOOL\_DIST\_ID"

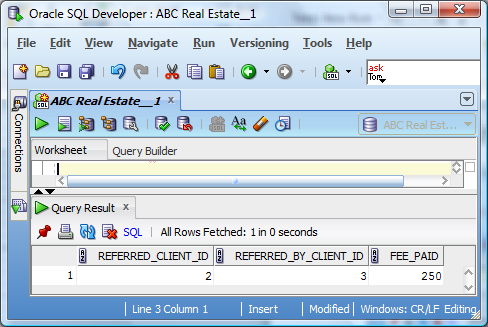
FROM PROPERTY\_AREA



--View for REFERRAL table – Referral Report

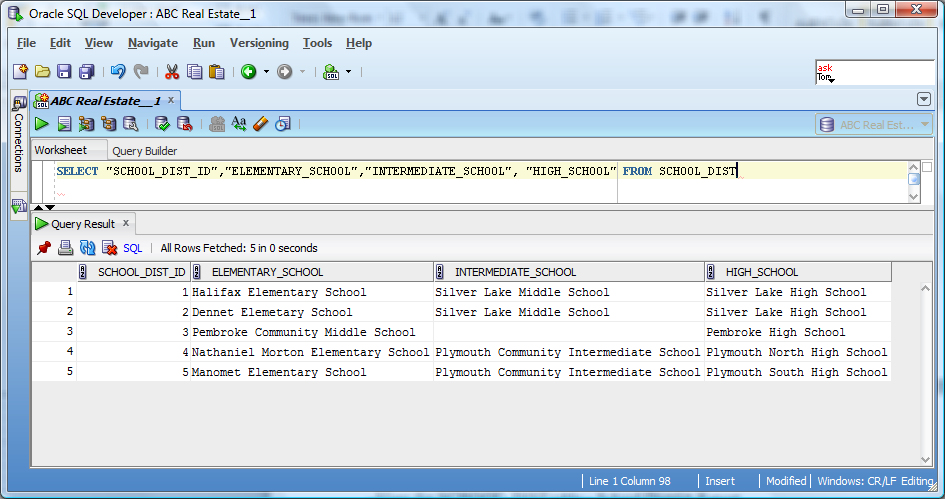
SELECT "REFERRED\_CLIENT\_ID","REFERRED\_BY\_CLIENT\_ID","FEE\_PAID"

FROM REFERRAL



--View for SCHOOL\_DIST table – School District Report

SELECT "SCHOOL\_DIST\_ID","ELEMENTARY\_SCHOOL","INTERMEDIATE\_SCHOOL", "HIGH\_SCHOOL" FROM SCHOOL\_DIST



--Custom View – Returns a listing of all active properties for sale in the town of Plympton with both the seller and agency contact information.

SELECT A.STREET, A.CITY, A.STATE, A.ZIP\_CODE, A.NUM\_BEDROOMS AS BEDROOMS, A.NUM\_BATHROOMS AS BATHROOMS, A.LIST\_PRICE, A.LOT\_SIZE, A.ASSOC\_FEE, B.FIRST\_NAME AS SELLERS\_FIRST\_NAME, B.LAST\_NAME AS SELLERS\_LAST\_NAME, B.PHONE AS SELLERS\_PHONE, B.E\_MAIL AS SELLERS\_EMAIL, C.AGT\_NAME AS AGENCY\_NAME, C.AGT\_PHONE AS AGENCY\_PHONE

FROM

PROPERTY A

INNER JOIN

CLIENT B

ON A.CLIENT\_ID = B.CLIENT\_ID

INNER JOIN

AGENCY C

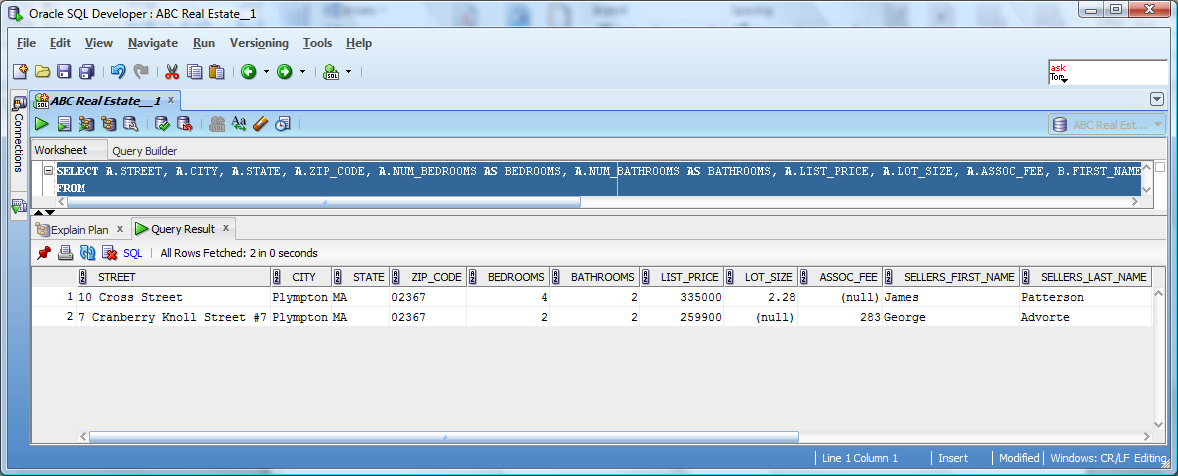
ON A.AGT\_ID = C.AGT\_ID

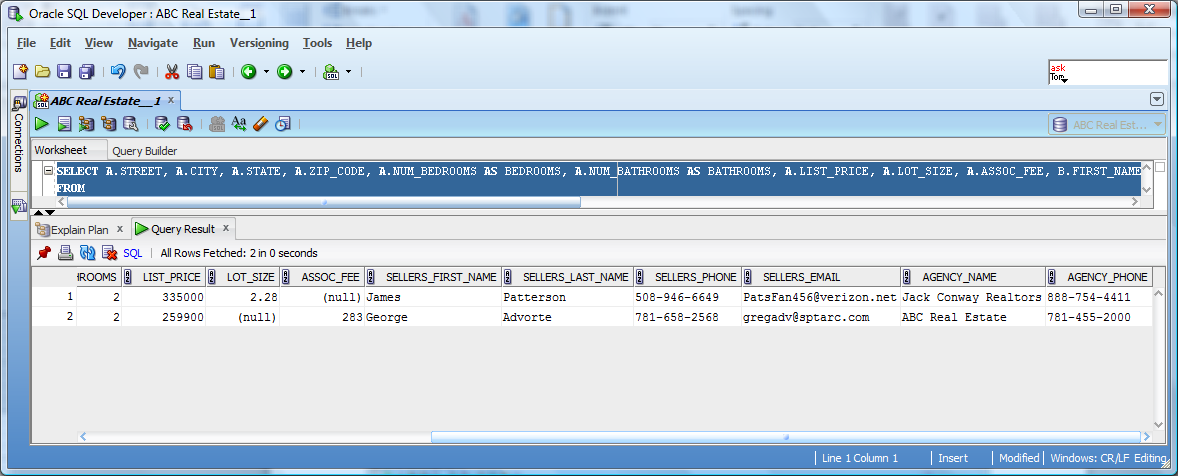
INNER JOIN

LISTING\_STATUS D

ON A.LISTING\_STATUS\_ID = D.LISTING\_STATUS\_ID

WHERE D.STATUS = 'Active' AND A.CITY = 'Plympton'





Data Dictionary

The corresponding data dictionary is located in the attached excel file under the Data Dictionary tab.