**GROUP C**

**Yousef Alhaizaey**

**Luis R. Buitrago**

**Haotian Du**

**Ankur Panchal**

**Chunan Shen**

**Saurabh Shetty**

**Ning Tong**

Design and Implementation of an Oracle Database

**George Washington University**

**Database Management Systems**

**Professor: Dr. Rolando Fernandez**

**Team Project**

**Spring 2014**



**Table of Contents**

[INTRODUCTION 2](#_Toc386975717)

[**PART 1** 5](#_Toc386975718)

[**PART 2** 5](#_Toc386975719)

[2.1 Business Information 5](#_Toc386975720)

[2.2 Data Flow Diagram (DFD) 7](#_Toc386975721)

[2.3 Entity Relationship Diagram (ERD) 11](#_Toc386975722)

[2.4 Lessons Learned in Part II 12](#_Toc386975723)

[**PART 3** 13](#_Toc386975724)

[3.1 Dependency Diagram 13](#_Toc386975725)

[3.2 Relational Diagram 14](#_Toc386975726)

[3.3 Data Definition Language (DDL) 15](#_Toc386975727)

[3.4 Concepts learned in part 3 20](#_Toc386975728)

[**PART 4** 21](#_Toc386975729)

[4.1 Optimization and Normalization 21](#_Toc386975730)

[4.2 Database Tables Population 22](#_Toc386975731)

[4.3 Solution of Queries and Reports 26](#_Toc386975732)

[4.4 Lessons Learned in Part IV 40](#_Toc386975733)

[**PART 5** 40](#_Toc386975734)

[5.1 History Triggers 40](#_Toc386975735)

[5.2 Auditing Triggers 41](#_Toc386975736)

[5.3 Prototypes 43](#_Toc386975737)

[5.3.1 Website Prototype 43](#_Toc386975738)

[5.3.2 Java-GUI Prototype 45](#_Toc386975739)

[5.4 Front-end Customers Interface 47](#_Toc386975740)

[5.5 System Administration Facility 48](#_Toc386975741)

[5.5 SQL Scripts 49](#_Toc386975742)

[5.6 Lessons Learned in Part V 49](#_Toc386975743)

[CONCLUSIONS 50](#_Toc386975744)

[BIBLIOGRAPHY 51](#_Toc386975745)

# INTRODUCTION

**A user account “G47757834” has been created in database “team2”.**

For this project is required to design and implement an Oracle database using the business rules and information proportioned in the class project documentation. The main idea of the project is to apply the concepts and methodology learned in class complemented with the Oracle material and software to create a new database for a Real State Agency. The tables, data and specific queries are the bases for the development of this project. Although, great amount of data and information is proportioned, some critical business facts will be assumed in order to complement the structure of the project.

My responsibility and role-played in this project are listed bleow.

* What was your role in Part 1 and what did you learn in this part?
  + Individually study the requirements of this project.
  + Analyze the information provided and find the best approach to efficiently develop the project through open discussion with team members.

In this part, I got familiar with how to write a proper project description and how to make your requirement clear to developers and designers.

* What was your role in Part 2 and what did you learn in this part?
  + Analyze and classify business information provided in the project description “Barkley\_Brothers\_Real\_Estate\_Proj.docx”.
  + Revise the Entity Relationship diagram created by team member to reduce redundancy.

This is one of the most critical steps in the process of constructing a new database. It is because the architect must be able to define the bases of the whole structure to later build the specifications and interfaces of top of them. Through the development of this part I learned the importance of every piece of information. Another important aspect is the fact that we were able to generate our own template or better said systematic approach to work on any new database system. We deeply believe that follow expertise advices is important in the learning process, as well as keeping the rules already established, but what really separate us from the rest of database architects is the implementation of our own methodology.

* What was your role in Part 3 and what did you learn in this part?
  + Re-evaluate the propriety of primary keys and foreign keys of each entity. Then create relational diagram using ERD.
  + Generate DDL from relational diagram and create all tables in database using Oracle data developer.

In this section, I have practiced creating logical model (Entity Relation Diagram), Relational Model, and generating DDL. This is an important part of the project because the Logical Data Model and Entity Relational Diagram (ERD) are converted to a relational diagram and model that includes the tables and schema so we can form the basis of the project’s relational database.

It’s known that Dependency Diagram and Relational Diagram are all based on ERD in part 2, the difference is that Dependency Diagram can give the relationship between each entities, and Relational Diagram can show us the primary keys, foreign keys and attributes’ data types. But both of them can give us a clear perspective about the whole project. Moreover, they serve as solid preparation for the following Data Definition Language that we are going to program.

* What was your role in Part 4 and what did you learn in this part?
  + Normalize the tables into the proper level based on their properties and the project requirement.
  + Write the SQL script for inserting data into the tables.
  + Write PL/SQL code for query 6 and query 7. And test them using data developer.

In this section I have learned the processes of normalization and optimization to analyzed the relations among attributes in each entity and establish improvements in their structure. As we mentioned in previous sections, every time that in our iterative approach we generate a new arrangement of entities and relationships, normalization and optimization were the following obligated steps. After models were generated and improved, we weighted each alternative to select the optimal model. The criteria used were model efficiency and accessibility to information requited by the stakeholders.

I also learned how to insert information systematically into the entities and how to resolve the queries and reports. By resolving all the queries we assured that the model was working correctly and opened our imagination to interfaces ideas.

* What was your role in Part 5 and what did you learn in this part?
  + Design and implement the Java GUI Prototype, which could remotely access and manipulate your database.
  + Combine all SQL scripts of this project and test it on data developer.

In this section I have learned the important concept of automation of business procedures and automatically maintaining data integrity and consistency. By using the triggers in data history and auditing we have practice how to implement these concepts that are very useful in architecting real life databases.

I have also learned how to implement two different prototypes in real life using totally different approaches, but based on the same construction concept. As a result of these two experiments we feel completely loaded to the necessary tools and knowledge to start any database endeavor.

* What did you learn in this project?

Through the development of this project we have designed and implemented an Oracle database using the business rules, Oracle software and information proportioned in class and referenced books. We now clearly understand how important and critical are the management of data for any organization and the implications of good or bad information proportioned to the decision makers. Quality of data which is the result of an efficiently database model is a comprehensive approach to ensure accuracy, validity and timeliness of data that might be used for critical decisions that will determine the future of an organization.

We have extracted the most of this project by creating a template for architecting a systematic approach to construct the optimal database model for any new database construction process. Our approach has to be tune-up while developing the next (probably real life) database projects. What is guarantee in the next endeavor is that the time to develop the new database will be considerable reduced and the template upgraded for better future functionality.

**PART 1**

In this part it is required to analyze the project requirements specified in the document “Barkley\_Brothers\_Real\_Estate\_Proj.docx”. We approached this part of the project first individually and then we scheduled an open discussion to analyze the information provided and to find the best approach to efficiently develop the project. All members have analyzed the information included in the document.

**PART 2**

Based on what is stipulated in the first paragraph of the document: “You are to design and implement an Oracle Database” we have deduced there is not existing database, and then we start for constructing a new one. Based on the Oracle documentation the best approach in this case is the Top-Down Modeling from which the reorganization of the points to develop in this part is as follows:

1. Business information: We will identify the business background to establish the company’s business direction
2. Process Model: construction of Dependency Flow Diagram (DFD) using Oracle software
3. Logical Model: Construction of Entity Relationship Diagram (ERD) using Oracle software

# 2.1 Business Information

Based on the atypical situation in which there is not contact with the stakeholders of the project and the only information is provided throughout a single document, we have decided to take the most from the information provided. Every sentence and word has been analyzed and classified in five different categories: Business objectives, assumptions, critical success factors, key performance indicators and problems. The quoted information provided by our client, the information classification, and team members comments are displayed in table 1.

**Table 1: Business Information Classification** (1=Business Objectives, 2=Assumption, 3=Critical Success Factor, 4=Key Performance Indicator, 5=Problem)

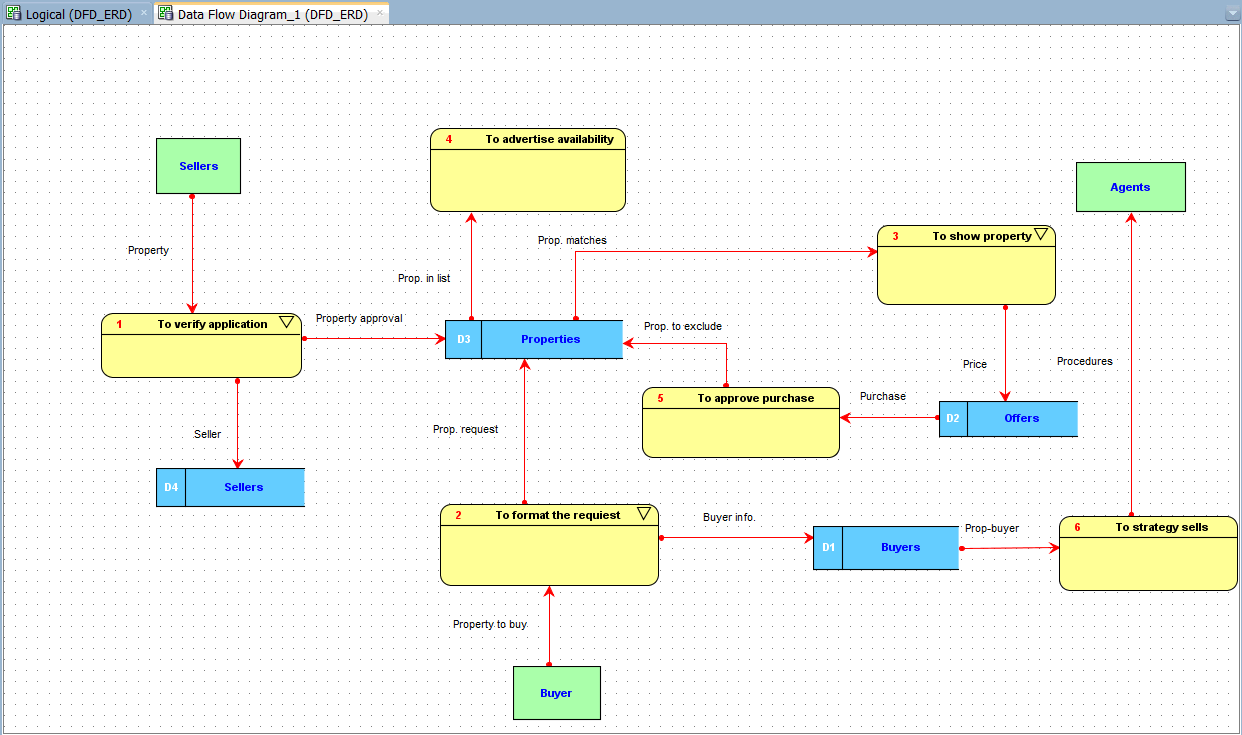
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Information provided** | **1** | **2** | **3** | **4** | **5** | **Team Assumptions/comments** |
| This is a real estate agency that has several branch offices |  |  |  |  |  | We will take main objectives from the type of business in general |
| Each branch office is located in a different city within the state of Virginia |  |  |  |  |  | Definition of the specific area of operation for the company |
| Each branch has a phone number, PO Box, mailing address, sales agent, and a manager |  |  |  |  |  | Information to verify against the tables given |
| Each branch office has a unique office name that is the same name as the city in which it is located |  |  |  |  |  | Information useful once determining the primary keys |
| The sales agents have a unique Agent Id. Each agent also has a first and last name, an office extension number and a home phone number |  |  |  |  |  | Information to verify against the tables given and the use of agent ID as a primary key (probably) |
| An agent works with both buyers and sellers |  |  |  |  |  | We are assuming that if not develop in this way, then the company will be affected either in its efficiency, or its revenue by incrementing the number of agents to cover the two types of clients |
| Agent- seller: the seller lists the property with the agent. The agent advertises and other duties necessary to notify the public and other agents of the property’s availability |  |  |  |  |  | The database must allow the agent to make the necessary inclusions to the database in efficient and intuitive way |
| Agent-buyer: the agent negotiates an offer for the property. An agent may work with a buyer for any property. This includes properties listed by him/herself, properties listed by other agents in his/her branch, or properties listed with agents of other branches |  |  |  |  |  | We believe that the company will be seriously impacted in achieving its main objectives if the agents cannot perform these activities as described in here. The database must allow the retrieval of all properties listed disregarding their location |
| The real estate agency also uses independent agents, they are not associated with any branch office |  |  |  |  |  |  |
| Sellers are those people that have listed property for sale with an agent |  |  |  |  |  | Just the definition of clients |
| Sellers have a unique Seller Id and keeps the name, address (City, state, zip), and phone number. |  |  |  |  |  | Important information to determine the primary keys |
| seller may have more than one property listed with an agent; however, all properties will be listed with the same agent |  |  |  |  |  | We believe that the company will face serous organizational problems if this rule is broken |
| If the seller is a bank or other type of holding company, the company will have a contact person. There will be only one contact person for a seller and a contact person can only represent one seller |  |  |  |  |  | We assume that this rule is important to maintain the efficiency and organization of the company |
| A listing number identifies properties that are listed with an agent. |  |  |  |  |  | This may define the entity with an attribute relationship |
| The agency keeps information about the Property (Type, location (Street, City, State, Zip) and Asking Price as well as a picture of the property. The agent also keeps track of the date the property was listed |  |  |  |  |  | This information will help us define the attributes to an specific entity |
| Buyers are people that have visited the branch office and viewed some of the properties listed with the office. |  |  |  |  |  | Description of a client: potential buyer is considered buyer without any transaction involved yet |
| A buyer is identified by a unique Buyer Id. Each buyer also leaves their name, address(Street, City, Sate, Zip), and phone number |  |  |  |  |  | Information useful to define a entity’s components including its primary key |
| A buyer will always work with the same agent |  |  |  |  |  | We assume that this rule is important to maintain the efficiency and organization of the company |
| Buyers will make an Offer for a Property to a Seller. This offer states the date of the offer and the amount of the offer. |  |  |  |  |  | This information is critical for the company so that follow ups can be organized more efficiently |
| A buyer may make an offer on more than one property and there may be more than one offer on a property |  |  |  |  |  | This information may help us to define the dependency relationships |
| The buyers may be interested in several properties. Several buyers may also be interested in the same property |  |  |  |  |  | This information may help us to define the dependency relationships |
| Some technique must be implemented to find all properties that a buyer is interested in and all buyers that may be interested in a specific property |  |  |  |  |  | We assume this is a critical factor so that improving this aspect will increase efficiency and revenue |

**1**=Business Objectives**, 2**=Assumption, **3=**Critical Success Factor, **4**=Key Performance Indicator, **5**=Problem

# 2.2 Data Flow Diagram (DFD)

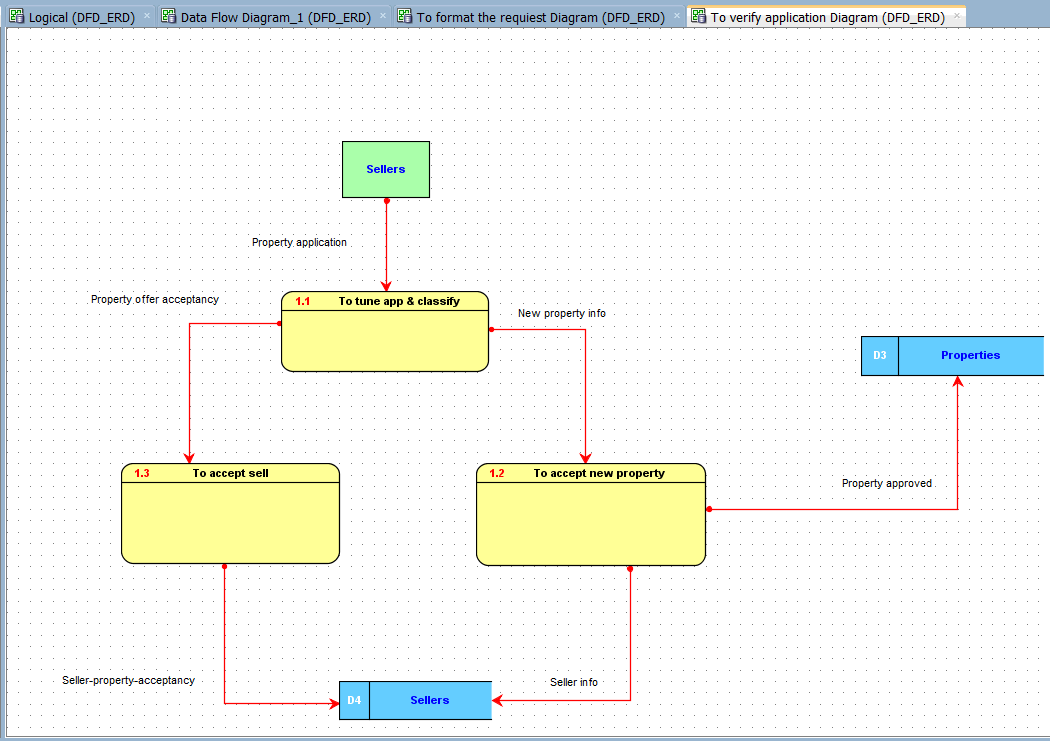
As it was mention in the introduction, important aspects of the business related to the Barkley Brothers has been taken from the general Real State business group: profit objectives, impact in the market place, competitive aspects in the vicinity area, etc.,. In general terms Barkley Brothers Real State moves around three major agents (sellers, buyers and sales agents), several processes and sub processes and a few information stores that keep vital information about who is involved in the process and what property or properties he/she is dealing with. The following figures show all detailed aspects of major business processes.

***Figure 1: DFD (Main Diagram)***



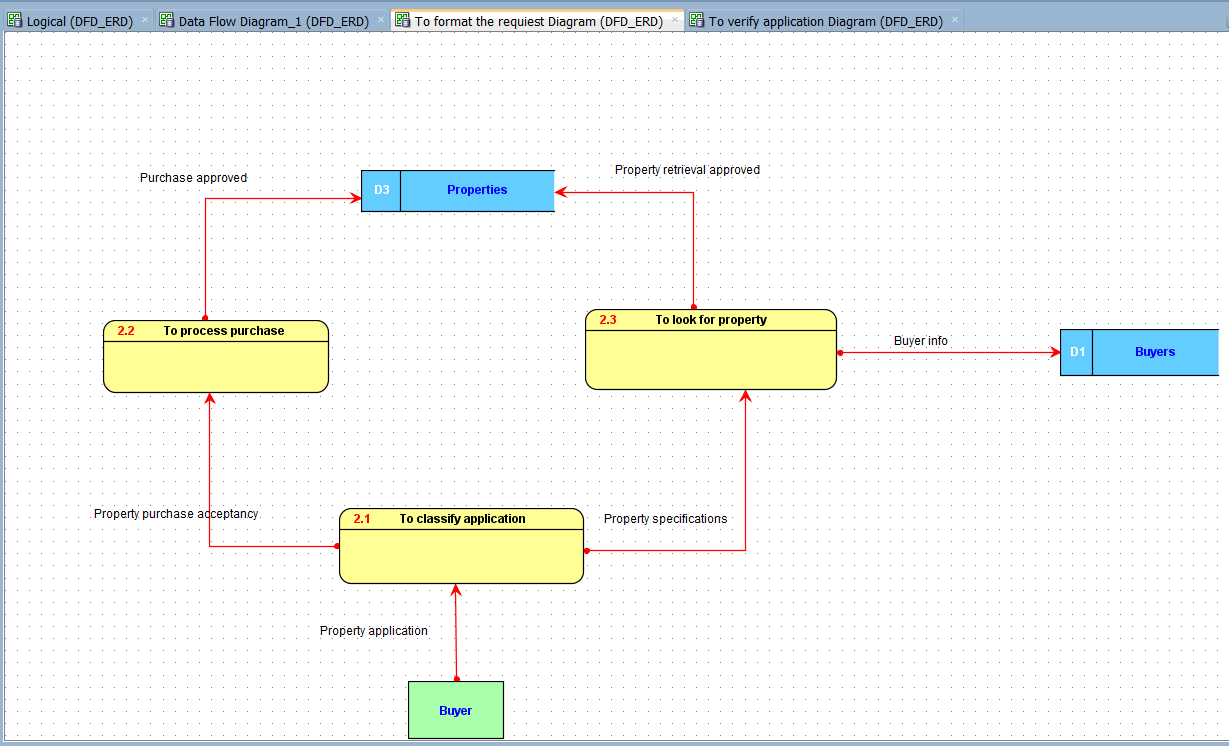
As it is shown in the previous figure, the process of verification of an application that contains a potential new property to offer is considered as a sub process due to the importance to the company. The Real State business’ row material is the properties to sell or rent. The sub process of verification of an application is described in Figure 2.

***Figure 2: Sub-process of process 1 (To verify application)***



Another important sub process is how to deal with a buyer request in a way that the transaction is secure, agile, and reflects enough confidence to the customer. The first step is classify the request, either the customer wants to buy a specific property and he/she want to place an offer, or there is a desire to see properties under specific characteristics. The Figure 3 shows in detail these two different ways to process the request.

***Figure 3: Sub-process of process 2 (To format the request)***



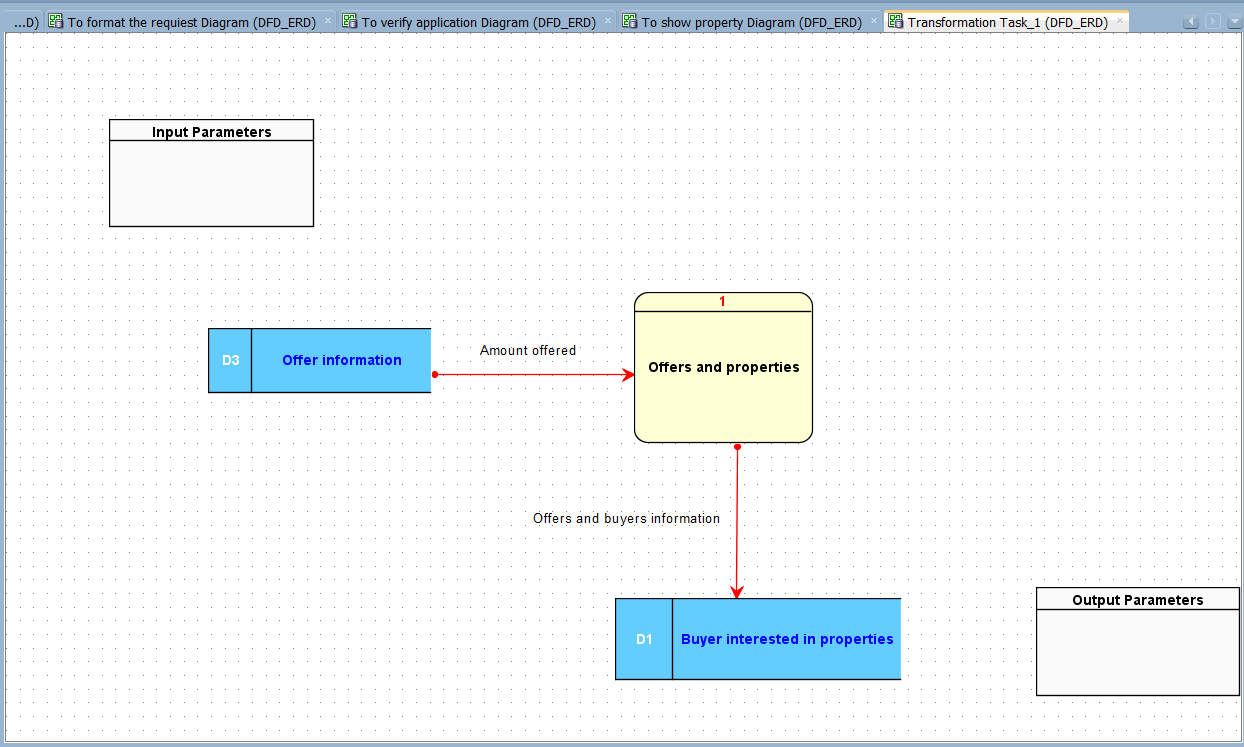
Once the customer (potential buyer) specifications of a property has been taken and processed, the result of this query must be a list of properties that satisfy the customer’s desire. Offers are expected to be placed by the buyer afterwards. The sub process to show property is detailed in Figure 4.

***Figure 4: Sub-process of process 3 (To show property)***



To list the offers is considered a transformation process in which important information is taken from the databases about properties that match the customer specification, and one or more properties are selected to be part of another list in which extra information is added to the property. That is, the buyer information, offer date, and amount he/she offers. Details about this transformation process are shown in Figure 5.

***Figure 5: Transformation process (To list offers)***



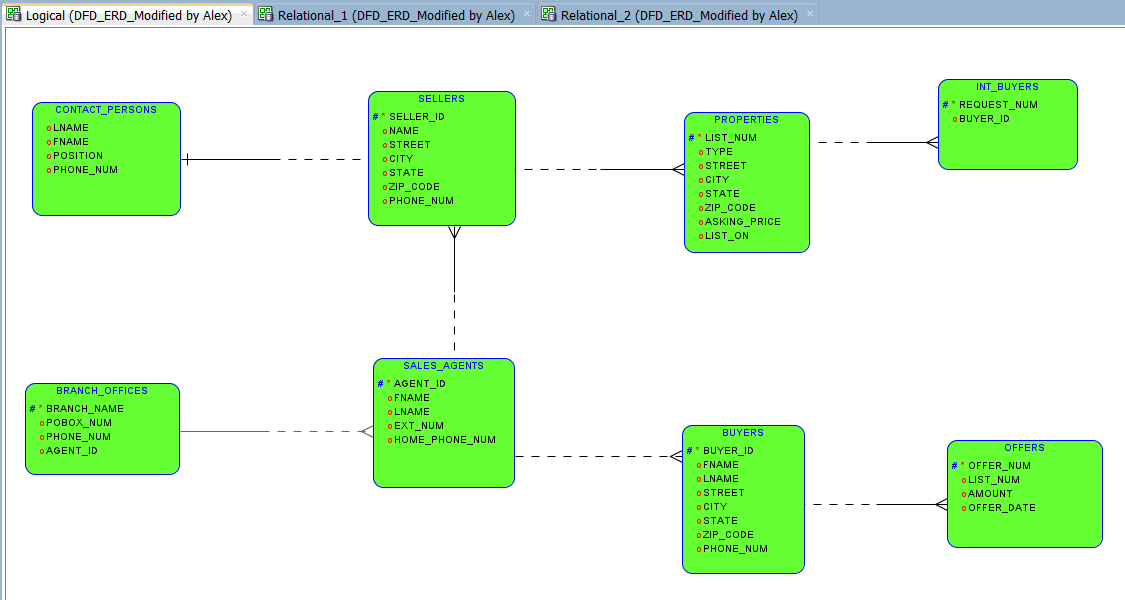
# 2.3 Entity Relationship Diagram (ERD)

It is important to notice that our entity relationship diagram shown in this section is the final and only version that represents the core information for the entire project. This effective and efficient diagram has been achieved through a deep and intense iterative analysis process that involved all steps of the project.

First, we extracted and organized the information that was clearly concentrated in a single entity such as properties, sales agents, buyers and sellers. The scattered information provided in the document was deeply analyzed based on the topic and functionality. Thus, under these criteria it was stored in individual packages and named accordingly to its topic. Four more groups were created: interested buyers, branch offices, offers, and contact persons. Without knowing the most suitable place to locate these entities and its relationship with the entities around, we started the iterative analytical process by locating the entities in different places to evaluate normalization, optimization and the client outcomes represented in the 15 queries, the triggers requested, the 2 forms, and the interfaces with the internal and external customers.

The optimal diagram was tested against minimal controlled redundancy, possesses minimal duplicated attributes and guarantees not only the current requirements stipulated by our client, but also possible new information and interfaces in which he/she might be interested in the future. The ERD is shown in Figure 6.

***Figure 6: Entity Relationship Diagram (ERD)***



# 2.4 Lessons Learned in Part II

This is one of the most critical steps in the process of constructing a new database. It is because the architect must be able to define the bases of the whole structure to later build the specifications and interfaces of top of them. As it was mentioned in class this is an iterative process. We took it so seriously that we spent more than 50% of the whole project to obtain the optimal model.

Through the development of this part we learned the importance of every piece of information, especially when there is no stakeholder around. This atypical situation allowed us to learn more about the Real State business in order to make educated assumptions.

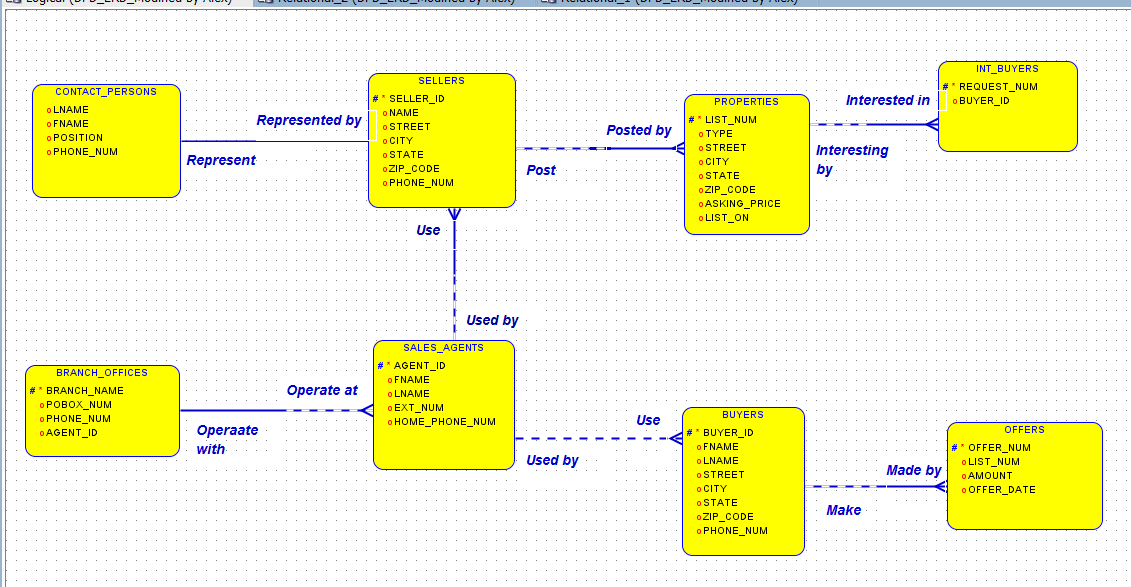
Another important aspect is the fact that we were able to generate our own template or better said systematic approach to work on any new database system. We deeply believe that follow expertise advices is important in the learning process, as well as keeping the rules already established, but what really separate us from the rest of database architects is the implementation of our own methodology. The methodology that provides the comfort and confidence to generate improvements in achieving the most efficient process in the creation of new database systems

**PART 3**

# 3.1 Dependency Diagram

We are representing the high level dependency diagram that is related to the entities interaction that allows fluent communication among all entities into the process. At this point of our approach, each entity is representing a single subject containing information related exclusively to its own. And all primary keys have been assigned, and the non-primary attributes are dependent on the primary key only and no to any other attribute. The aspects of functional dependency will be described in the section 4 on normalization. The high level dependency diagram is shown in Figure 7.

***Figure 7: Dependency Diagram***

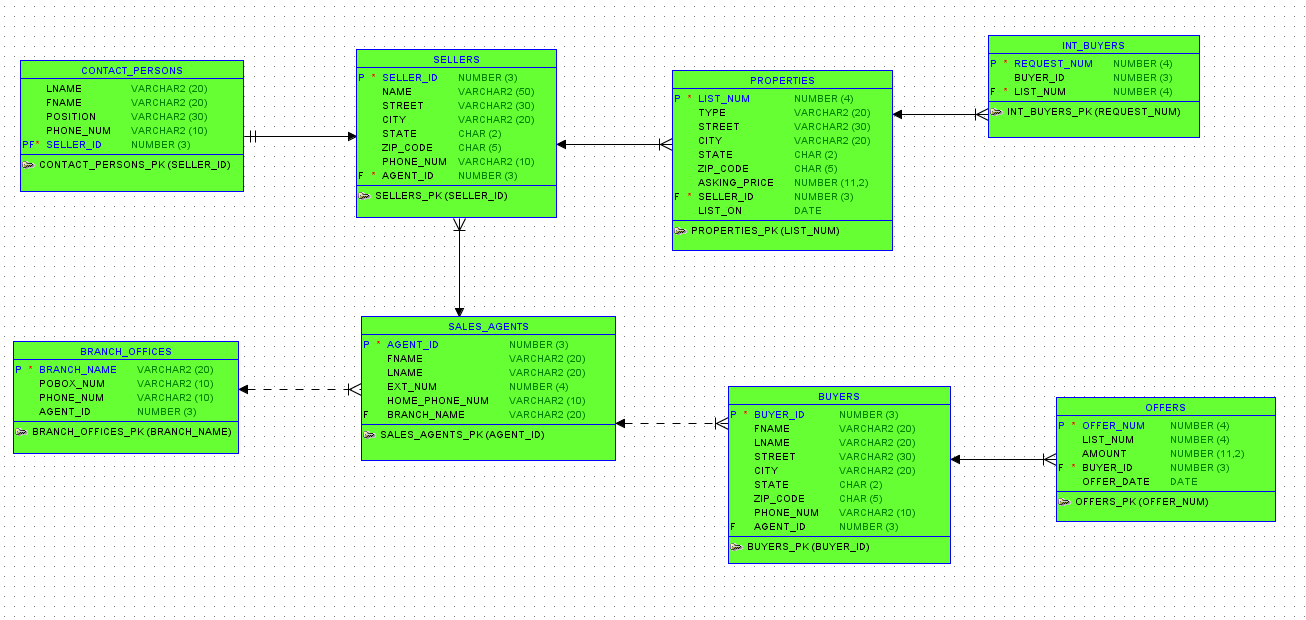


# 3.2 Relational Diagram

Using the powerful tools of the Oracle SQL Developer Data Modeler software the relational diagram is generated. The optimal model is introduced in the software feed only with the primary keys for each entity. It is important to notice entity contact persons, which was intentionally left without primary key, counting on the software capability to produce it and also the required foreign keys for the other entities.

After the diagram is generated, we performed testing with several queries to verity functionality. We checked that foreign keys were correctly placed and that the names automatically generated by the system to the keys were under the stipulated length parameter. The relational diagram is shown in Figure 8.

***Figure 8: Relational Diagram***



# 3.3 Data Definition Language (DDL)

To generate the data definition language DDL it was used one of the tools of the Oracle SQL Developer Data Modeler software. The DDL code to create the structure of the tables is:

-- Generated by Oracle SQL Developer Data Modeler 4.0.0.833

-- at: 2014-05-03 16:58:06 EDT

-- site: Oracle Database 11g

-- type: Oracle Database 11g

CREATE TABLE BRANCH\_OFFICES

(

BRANCH\_NAME VARCHAR2 (20) NOT NULL ,

POBOX\_NUM VARCHAR2 (10) ,

PHONE\_NUM VARCHAR2 (10) ,

AGENT\_ID NUMBER (3)

) ;

COMMENT ON TABLE BRANCH\_OFFICES

IS

'BRANCH\_OFFICES' ;

COMMENT ON COLUMN BRANCH\_OFFICES.AGENT\_ID

IS

'Manager of the branch office' ;

ALTER TABLE BRANCH\_OFFICES ADD CONSTRAINT BRANCH\_OFFICES\_PK PRIMARY KEY ( BRANCH\_NAME ) ;

CREATE TABLE BUYERS

(

BUYER\_ID NUMBER (3) NOT NULL ,

FNAME VARCHAR2 (20) ,

LNAME VARCHAR2 (20) ,

STREET VARCHAR2 (30) ,

CITY VARCHAR2 (20) ,

STATE CHAR (2) ,

ZIP\_CODE CHAR (5) ,

PHONE\_NUM VARCHAR2 (10) ,

AGENT\_ID NUMBER (3)

) ;

COMMENT ON TABLE BUYERS

IS

'BUYERS' ;

ALTER TABLE BUYERS ADD CONSTRAINT BUYERS\_PK PRIMARY KEY ( BUYER\_ID ) ;

CREATE TABLE CONTACT\_PERSONS

(

LNAME VARCHAR2 (20) ,

FNAME VARCHAR2 (20) ,

POSITION VARCHAR2 (30) ,

PHONE\_NUM VARCHAR2 (10) ,

SELLER\_ID NUMBER (3) NOT NULL

) ;

COMMENT ON TABLE CONTACT\_PERSONS

IS

'CONTACT\_PERSONS' ;

ALTER TABLE CONTACT\_PERSONS ADD CONSTRAINT CONTACT\_PERSONS\_PK PRIMARY KEY ( SELLER\_ID ) ;

CREATE TABLE INT\_BUYERS

(

REQUEST\_NUM NUMBER (4) NOT NULL ,

BUYER\_ID NUMBER (3) ,

LIST\_NUM NUMBER (4) NOT NULL

) ;

COMMENT ON TABLE INT\_BUYERS

IS

'INTERESTED\_BUYERS' ;

ALTER TABLE INT\_BUYERS ADD CONSTRAINT INT\_BUYERS\_PK PRIMARY KEY ( REQUEST\_NUM ) ;

CREATE TABLE OFFERS

(

OFFER\_NUM NUMBER (4) NOT NULL ,

LIST\_NUM NUMBER (4) ,

AMOUNT NUMBER (11,2) ,

BUYER\_ID NUMBER (3) NOT NULL ,

OFFER\_DATE DATE

) ;

COMMENT ON TABLE OFFERS

IS

'OFFERS' ;

ALTER TABLE OFFERS ADD CONSTRAINT OFFERS\_PK PRIMARY KEY ( OFFER\_NUM ) ;

CREATE TABLE PROPERTIES

(

LIST\_NUM NUMBER (4) NOT NULL ,

P\_TYPE VARCHAR2 (20) ,

P\_STREET VARCHAR2 (30) ,

CITY VARCHAR2 (20) ,

STATE CHAR (2) ,

ZIP\_CODE CHAR (5) ,

ASKING\_PRICE NUMBER (11,2) ,

SELLER\_ID NUMBER (3) NOT NULL ,

LIST\_ON DATE

) ;

COMMENT ON TABLE PROPERTIES

IS

'PROPERTIES' ;

ALTER TABLE PROPERTIES ADD CONSTRAINT PROPERTIES\_PK PRIMARY KEY ( LIST\_NUM ) ;

CREATE TABLE SALES\_AGENTS

(

AGENT\_ID NUMBER (3) NOT NULL ,

FNAME VARCHAR2 (20) ,

LNAME VARCHAR2 (20) ,

EXT\_NUM NUMBER (4) ,

HOME\_PHONE\_NUM VARCHAR2 (10) ,

BRANCH\_NAME VARCHAR2 (20)

) ;

COMMENT ON TABLE SALES\_AGENTS

IS

'SALES\_AGENTS' ;

ALTER TABLE SALES\_AGENTS ADD CONSTRAINT SALES\_AGENTS\_PK PRIMARY KEY ( AGENT\_ID ) ;

CREATE TABLE SELLERS

(

SELLER\_ID NUMBER (3) NOT NULL ,

S\_NAME VARCHAR2 (50) ,

STREET VARCHAR2 (30) ,

CITY VARCHAR2 (20) ,

STATE CHAR (2) ,

ZIP\_CODE CHAR (5) ,

PHONE\_NUM VARCHAR2 (10) ,

AGENT\_ID NUMBER (3) NOT NULL

) ;

COMMENT ON TABLE SELLERS

IS

'SELLERS' ;

ALTER TABLE SELLERS ADD CONSTRAINT SELLERS\_PK PRIMARY KEY ( SELLER\_ID ) ;

ALTER TABLE BUYERS ADD CONSTRAINT BUYERS\_SALES\_AGENTS\_FK FOREIGN KEY ( AGENT\_ID ) REFERENCES SALES\_AGENTS ( AGENT\_ID ) ;

ALTER TABLE CONTACT\_PERSONS ADD CONSTRAINT CONTACT\_PERSONS\_SELLERS\_FK FOREIGN KEY ( SELLER\_ID ) REFERENCES SELLERS ( SELLER\_ID ) ;

ALTER TABLE INT\_BUYERS ADD CONSTRAINT INT\_BUYERS\_PROPERTIES\_FK FOREIGN KEY ( LIST\_NUM ) REFERENCES PROPERTIES ( LIST\_NUM ) ;

ALTER TABLE OFFERS ADD CONSTRAINT OFFERS\_BUYERS\_FK FOREIGN KEY ( BUYER\_ID ) REFERENCES BUYERS ( BUYER\_ID ) ;

ALTER TABLE PROPERTIES ADD CONSTRAINT PROPERTIES\_SELLERS\_FK FOREIGN KEY ( SELLER\_ID ) REFERENCES SELLERS ( SELLER\_ID ) ;

ALTER TABLE SALES\_AGENTS ADD CONSTRAINT SALES\_AGENTS\_BRANCH\_OFFICES\_FK FOREIGN KEY ( BRANCH\_NAME ) REFERENCES BRANCH\_OFFICES ( BRANCH\_NAME ) ;

ALTER TABLE SELLERS ADD CONSTRAINT SELLERS\_SALES\_AGENTS\_FK FOREIGN KEY ( AGENT\_ID ) REFERENCES SALES\_AGENTS ( AGENT\_ID ) ;

-- Oracle SQL Developer Data Modeler Summary Report:

--

-- CREATE TABLE 8

-- CREATE INDEX 0

-- ALTER TABLE 15

-- CREATE VIEW 0

-- CREATE PACKAGE 0

-- CREATE PACKAGE BODY 0

-- CREATE PROCEDURE 0

-- CREATE FUNCTION 0

-- CREATE TRIGGER 0

-- ALTER TRIGGER 0

-- CREATE COLLECTION TYPE 0

-- CREATE STRUCTURED TYPE 0

-- CREATE STRUCTURED TYPE BODY 0

-- CREATE CLUSTER 0

-- CREATE CONTEXT 0

-- CREATE DATABASE 0

-- CREATE DIMENSION 0

-- CREATE DIRECTORY 0

-- CREATE DISK GROUP 0

-- CREATE ROLE 0

-- CREATE ROLLBACK SEGMENT 0

-- CREATE SEQUENCE 0

-- CREATE MATERIALIZED VIEW 0

-- CREATE SYNONYM 0

-- CREATE TABLESPACE 0

-- CREATE USER 0

--

-- DROP TABLESPACE 0

-- DROP DATABASE 0

--

-- REDACTION POLICY 0

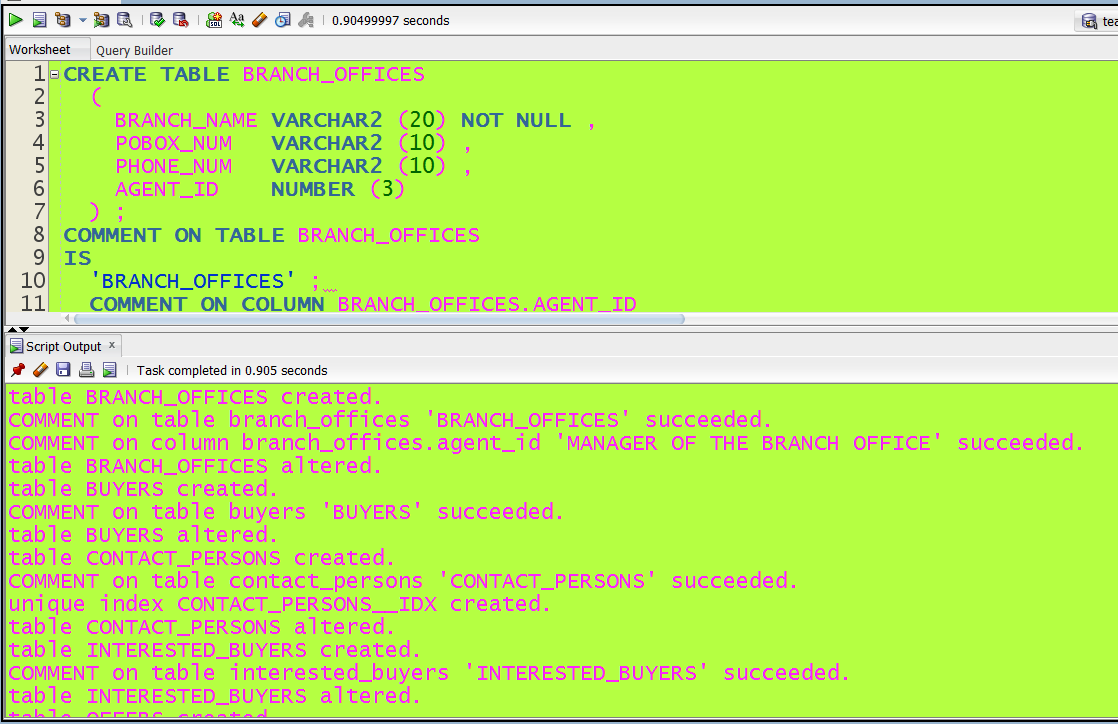
--

-- ERRORS 0

-- WARNINGS 0

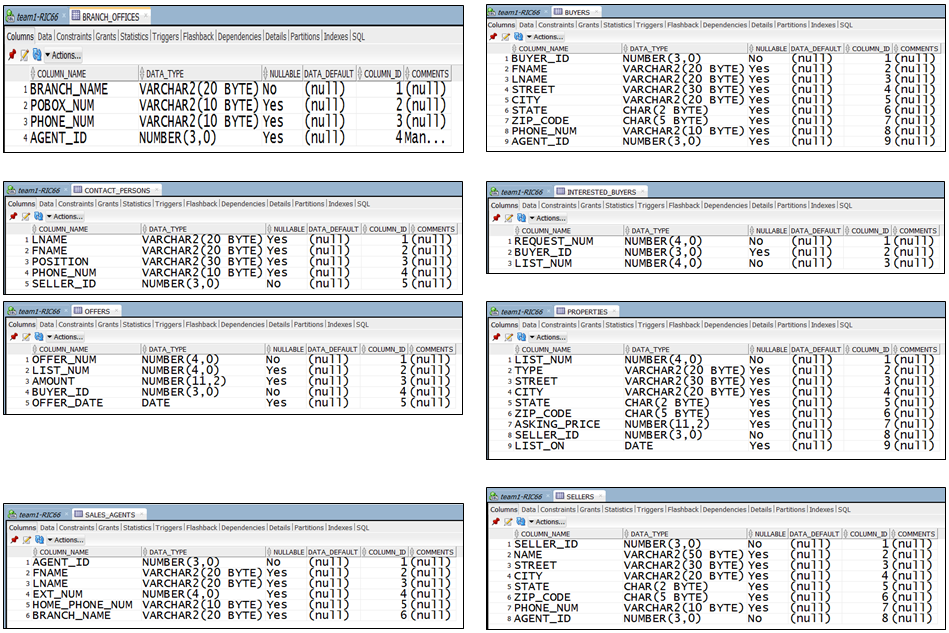
The working code at the Oracle developer software is showed in Figure 9. This process was used to create the tables.

***Figure 9: Generation of Tables***



And the tables created with this code are showed in Figure 10

***Figure 10: Tables created***



# 3.4 Concepts learned in part 3

In this section, we have practiced creating logical model (Entity Relation Diagram), Relational Model, and generating DDL. We used the Oracle Data Modeler to formalize the entities and their attributes and create relationships among them based on our analysis of the project. Then, we validated the created ER to ensure that the relationships between the entities are logical and normalized. Then, we generate the Relational Model using Oracle Data Modeler. We validated the created Relational Model to ensure that each table has primary key and each attribute has assigned a data type. We also validated the relationships among tables. Finally, we generated the DDL (Data Definition Language) in order to create our Database. As can be seen in Figure 10, we created tables using Oracle Data Developer, and the next step is to insert data into the Database.

This is an important part of the project because the Logical Data Model and Entity Relational Diagram (ERD) are converted to a relational diagram and model that includes the tables and schema so we can form the basis of the project’s relational database.

We know that Dependency Diagram and Relational Diagram are all based on ERD in part 2, the difference is that Dependency Diagram can give the relationship between each entities, and Relational Diagram can show us the primary keys, foreign keys and attributes’ data types. But both of them can give us a clear perspective about the whole project. Moreover, they serve as solid preparation for the following Data Definition Language that we are going to program.

After we make sure Dependency Diagram and Relational Diagram were matched to each other, the DDL can easily create the initial database schema and tables based on these two diagrams. DDL is a syntax which is similar to a computer programming language for defining data structures, especially database schemas. SQL uses a collection of imperative verbs whose effect is to modify the schema of the database by adding, changing, or deleting definitions of tables or other objects.

Therefore, we learned how to translate the ERD to Relational Diagram and Dependency Diagram with tables that adhere to integrity constraints based on primary and foreign keys, and unique constraint requirements. And then we wrote DDL script to create the database schema and tables.

**PART 4**

# 4.1 Optimization and Normalization

Database optimization is done by assigning appropriate keys to the various tables in the database. One way of optimization is assigning Primary keys, foreign keys and optimizing natural joins. In our database the following are some examples of processes of normalization and optimization used:

* We have defined the primary keys for all the tables. For example, the SELLER\_ID in the sellers table can be used to uniquely identify each record in the SELLERS table, so we have defined it as the primary key for the table. Similarly we have assigned primary keys to all tables in our database.
* The CONTACT\_PERSONS and SELLERS table are dependent on each other as every contact person represents a seller. To represent this relation we have defined the foreign key constraint in the CONTACT\_PERSONS table by defining the SELLER\_ID as a foreign key which references the primary key of the SELLERS table.
* The SELLERS and PROPERTIES tables are dependent on each other as a property can only be posted by a seller. Thus we have represented this relation in our database by defining a foreign key in the SELLERS table which references SELLER\_ID in the sellers table.
* Similarly foreign key constraints have been defined for all the tables that are dependent to each other, thereby optimizing the database.

Normalization is the process of organizing the fields and tables in a database to minimize redundancy and make sure that the data dependencies make sense. Normalization has been done on all the tables in the database by using the decomposition mechanism. We have normalized most of our tables unto the third normal form. This is because any further normalization would tend to just increase our work rather than reducing it and making it easy. Here is a brief description of how we define whether the table is in a particular Normal Form:

1NF: A table is said to be in 1NF if all the attributes are related, the table has a primary key and finally, the table does not have duplicate sets of values. 2NF: A table is said to be in 2NF if it satisfies all the conditions of 1NF and there are no partial dependencies of any attributes with the primary key. 3NF: A table is said to be in 3NF if it satisfies the conditions of 2NF and non-primary key fields are dependent on the primary key. BCNF: A table is said to be in BCNF if, for each A🡪B, A is either a key or contains a key.

Based on the above rules we can classify the tables in our database as follows:

2NF: SELLERS, PROPERTIES, BUYERS are in the second normal form as they only satisfy the rules of 2NF. The do not satisfy the condition for third normal form that there should not be any transitive dependencies among the attributes.

3NF: The SALES\_AGENTS table is in the third normal form because it satisfies all the conditions of 2NF and also it does not have any transitive dependencies.

BCNF: The OFFERS, BRANCH\_OFFICES and CONTACT\_PERSONS tables are in BCNF as they satisfy the rule that for all functional dependencies (A🡪B) possible, A is a primary key.

The reason why all the tables were not normalized to BCNF was because any further decomposition of the tables in order to normalize them would have made it very hectic and confusing to manage the data.

# 4.2 Database Tables Population

The code to insert data into the tables is:

insert into BRANCH\_OFFICES values ('Williamsburg','PO Box 122','433-0009',10);

insert into BRANCH\_OFFICES values ('Elkton','PO Box 333','401-2222',30);

insert into BRANCH\_OFFICES values ('Bridgewater','PO Box 566','322-0909',50);

insert into SALES\_AGENTS values (10,'Jane','Smith',111,'433-1010','Williamsburg');

insert into SALES\_AGENTS values (20,'Ellen','Davis',222,'434-2000','Williamsburg');

insert into SALES\_AGENTS values (80,'Thomas','Pill',434,'403-7781','Williamsburg');

insert into SALES\_AGENTS values (30,'Frank','Ramey',333,'401-7777','Elkton');

insert into SALES\_AGENTS values (40,'Paul','Jones',444,'401-8010','Elkton');

insert into SALES\_AGENTS values (50,'Alfred','Baker',555,'668-3011','Bridgewater');

insert into SALES\_AGENTS values (60,'Mollie','Brown',666,'568-9021',null);

insert into SALES\_AGENTS values (70,'Sarah','Parket',777,'401-8765',null);

insert into BUYERS values (500,'Gene','Guall','12 Oak La.','Elkton','VA','31102','401-5522',10);

insert into BUYERS values (510,'Bob','Smith','367 Divot','Elkton','VA','31102','401-2222',null);

insert into BUYERS values (520,'Ina','Ingle','266 Snowbird','Williamsburg','VA','22801','434-9999',60);

insert into BUYERS values (530,'Jill','Jenkins','1966 Westover','Williamsburg','VA','22801','434-0000',10);

insert into BUYERS values (540,'Kim','Kemper','399 Market','Bridgewater','VA','22900','434-1000',20);

insert into BUYERS values (550,'Jane','Johnston','456 Lakeview','Williamsburg','VA','22801','433-5555',70);

insert into BUYERS values (560,'Mike','Miller','872 Arrowhead','Williamsburg','VA','22801','434-3000',20);

insert into BUYERS values (570,'Nelson','Jones','126 E. Market','Williamsburg','VA','22801','433-4567',30);

insert into BUYERS values (580,'Polly','Paul','411 Duke','Bridgewater','VA','22900','434-1234',null);

insert into BUYERS values (590,'Bob','Smith','196 Phillips','Williamsburg','VA','22801','434-1122',60);

insert into BUYERS values (600,'Tom','Turk','888 Market','Bridgewater','VA','22900','434-3344',70);

insert into SELLERS values (1,'Al Able','500 Maple','Elkton','VA','31102','401-1111',10);

insert into SELLERS values (2,'Bob Bodkin','410 South','Williamsburg','VA','22801','433-2222',10);

insert into SELLERS values (3,'Dominion National Bank','314 N. Main','Williamsburg','VA','22801','433-3333',20);

insert into SELLERS values(4,'Bob Ellis','300 Westover','Williamsburg','VA','22801','433-5555',30);

insert into SELLERS values (5,'Jill Turner','105 Market','Bridgewater','VA','22900','322-6789',30);

insert into SELLERS values (6,'Paula Jones','902 Park','Bridgewater','VA','22900','322-9011',60);

insert into SELLERS values (7,'Tammy Miller','230 Mt. Clinton Pk','Williamsburg','VA','22801','568-0000',10);

insert into SELLERS values (8,'First National Bank','109 E. Main','Bridgewater','VA','22900','322-4444',20);

insert into CONTACT\_PERSONS values ('Dike','Robert','Vice President','433-9922',3);

insert into CONTACT\_PERSONS values ('Lamb','Laura','Loan Officer','433-3434', 8);

insert into properties values (101, 'Single Family', '104 South St.' , 'Williamsburg', 'VA', '22801', 210000,1,'05-DEC-03');

insert into properties values (102, 'Building Lot', '200 E. Main,', 'Elkton', 'VA', '31102', 35000,2,'08-FEB-04');

insert into properties values (103, 'Single Family', '300 W. Market', 'Williamsburg', 'VA', '22801', 200000,3,'05-JAN-04');

insert into properties values (104, 'Single Family', '400 N. Maple', 'Elkton', 'VA', '31102', 175500,8,'16-MAR-04');

insert into properties values (105, 'Condo', '500 E. Oak', 'Elkton' ,'VA' ,'31102', 305600,8,'05-DEC-03');

insert into properties values (106, 'Building Lot', '600 N. Dogwood', 'Williamsburg', 'VA' ,'22801', 32000,3,'15-FEB-04');

insert into properties values (107, 'Single Family', '700 S. Crescent' , 'Bridgewater' ,'VA' ,'22900', 225000,4,'03-APR-04');

insert into properties values (108, 'Condo', '800 Lakewood Dr', 'Elkton', 'VA','31102', 235000,8,'08-NOV-03');

insert into properties values (109, 'Single Family', '419 Maple' , 'Williamsburg', 'VA', '22801' ,125500,3,'07-MAR-04');

insert into properties values (110, 'Townhouse', '902 Park', 'Bridgewater', 'VA', '22900', 119000,6,'25-JAN-04');

insert into properties values (111, 'Townhouse', '677 Market', 'Elkton', 'VA', '31102', 115500,5,'12-MAR-04');

insert into properties values (112, 'Single Family', '230 Mt. Clinton Pike', 'Williamsburg', 'VA', '22801', 135500,7,'08-FEB-04');

insert into INT\_BUYERS values (1,500,101);

insert into INT\_BUYERS values (2,500,102);

insert into INT\_BUYERS values (3,500,103);

insert into INT\_BUYERS values (4,530,101);

insert into INT\_BUYERS values (5,530,102);

insert into INT\_BUYERS values (6,530,106);

insert into INT\_BUYERS values (7,520,101);

insert into INT\_BUYERS values (8,520,106);

insert into INT\_BUYERS values (9,520,107);

insert into INT\_BUYERS values (10,570,106);

insert into INT\_BUYERS values (11,570,111);

insert into INT\_BUYERS values (12,540,102);

insert into INT\_BUYERS values (13,540,107);

insert into INT\_BUYERS values (14,540,111);

insert into INT\_BUYERS values (15,560,102);

insert into INT\_BUYERS values (16,590,111);

insert into INT\_BUYERS values (17,590,106);

insert into offers values (1,101,190000,530,'04-APR-04');

insert into offers values (2,102,32000,510,'11-APR-04');

insert into offers values (3,103,155000,540,'13-APR-04');

insert into offers values (4,101,201000,540,'12-APR-04');

insert into offers values (5,108,225000,530,'11-APR-04');

insert into offers values (6,106,29000,550,'04-APR-04');

insert into offers values (7,109,115550,550,'17-APR-04');

insert into offers values (8,101,195000,550,'06-APR-04');

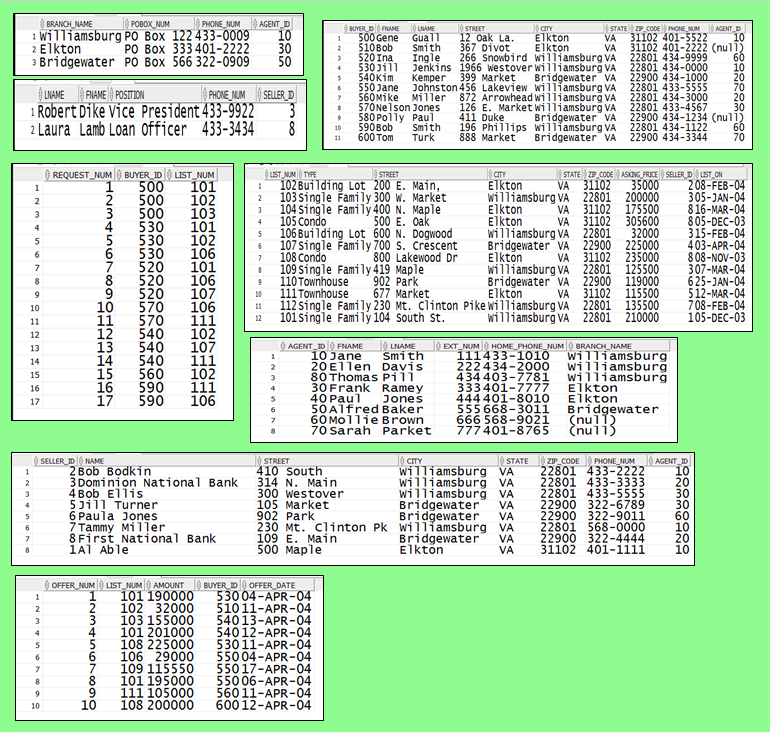
insert into offers values (9,111,105000,560,'11-APR-04');

insert into offers values (10,108,200000,600,'12-APR-04');

commit;

The tables with data included are showed in Figure 11

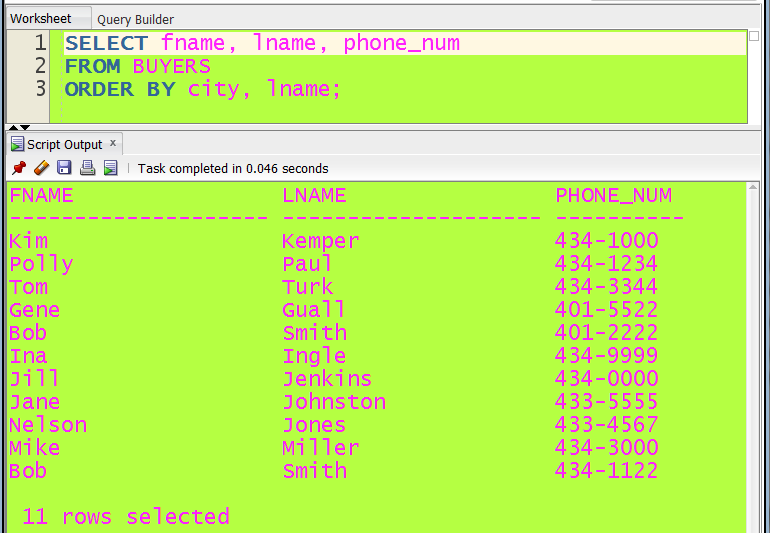
***Figure 11: Tables and Data Included***



# 4.3 Solution of Queries and Reports

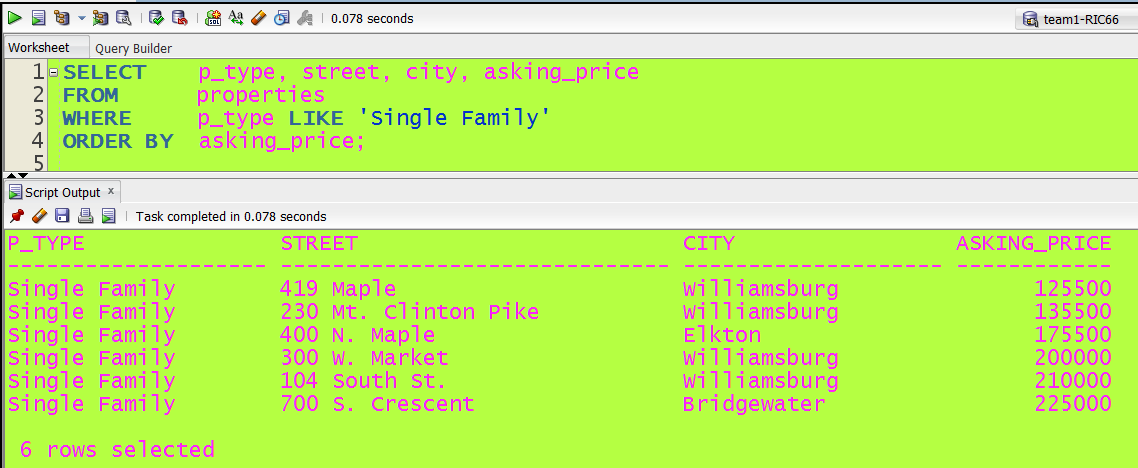
Query1: List the First Name, Last Name, and Phone of all Buyers. Order the list by City. Within City, order the list by Last Name.

***Figure 12: Solution Query 1***



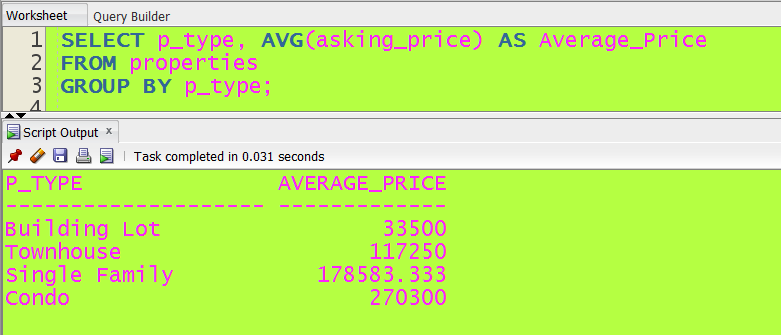
Query2: List the Property type, Street, City, and Asking price of all Single Family properties in the database. Order the properties by Asking Price.

***Figure 13: Solution Query 2***



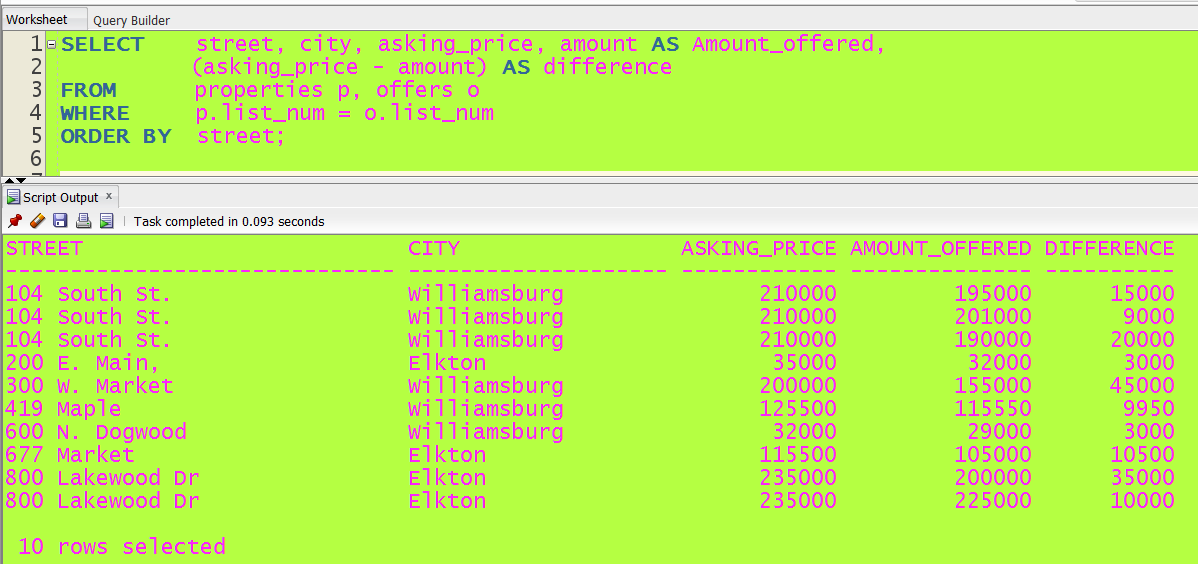
Query3: List each property type and the average price for the property type.

***Figure 14: Solution Query 3***



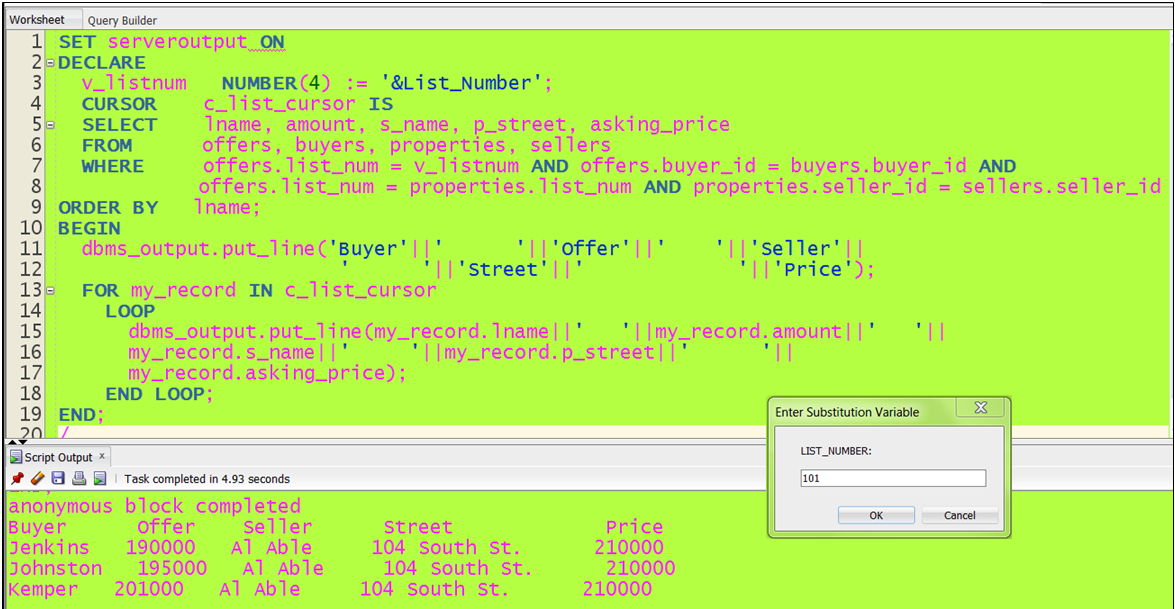
Query4: List each property in the database (Street, City, and Asking Price), the offers on the house Offer amount), and the difference between the Asking Price and the Offer Amount (Asking-Offer); name the difference column DIFFERENCE. Do not list houses that have no offers. Order the list by Street address.

***Figure 15: Solution Query 4***



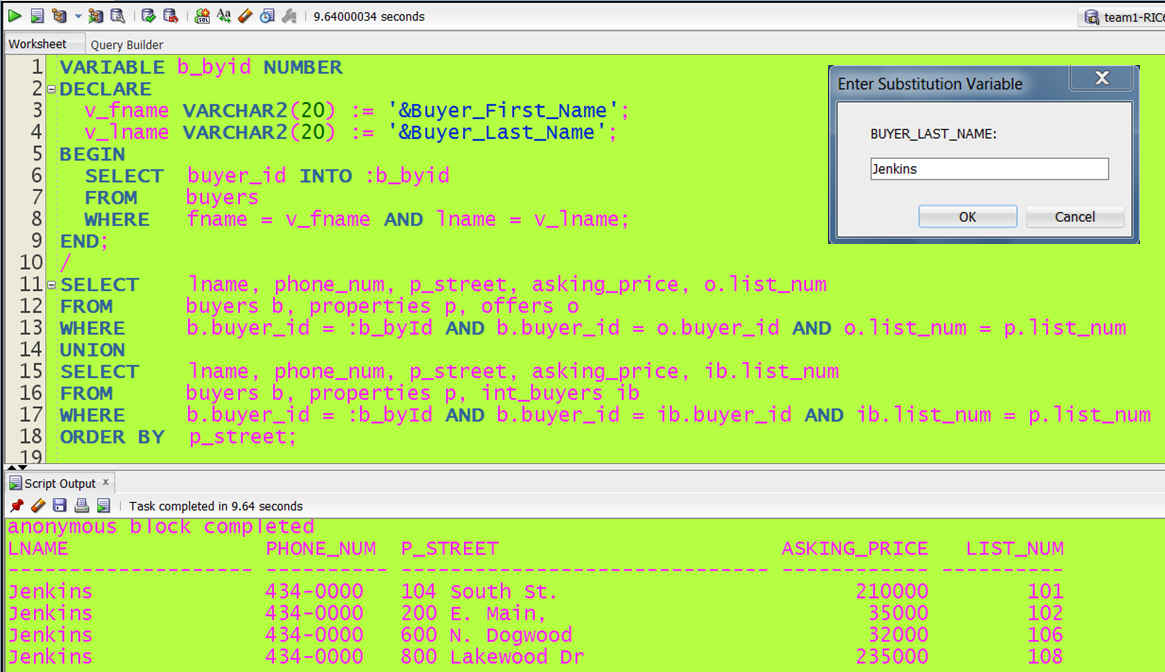
Query5: Allow a user to enter a **Listing Number**. List the Buyer’s name, offer amount, Seller’s name, house street address and asking price for any offer on the property. Order the list by buyer’s last name. Use Listing 101 to test your query.

***Figure 16: Solution Query 5***



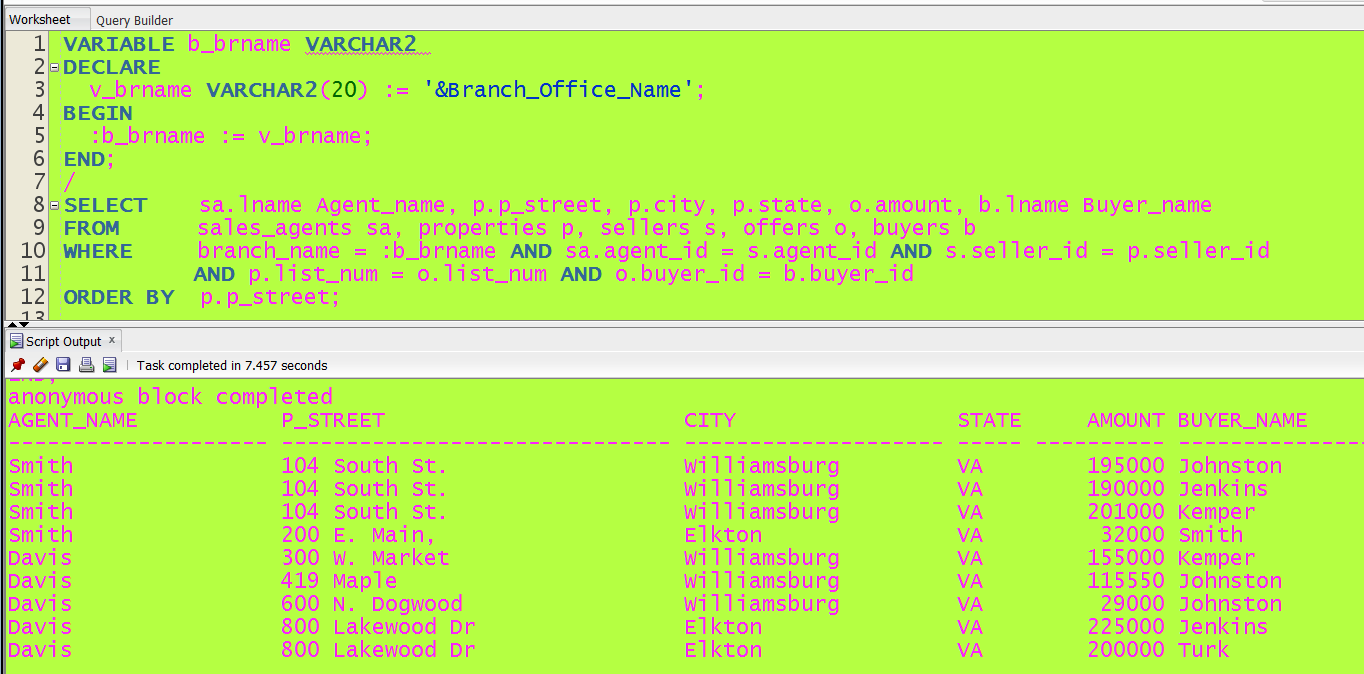
Query6: Allow a user to enter a **buyer’s name** (First and Last); using the buyers name, list the buyers name and phone number, house street and asking price of all houses that the buyer may be interested. Include houses that the buyer has placed an offer on **and** houses they are interested in but have not placed an offer on. Order the list by Street address. Test your query using Jill Jenkins.

***Figure 17: Solution Query 6***

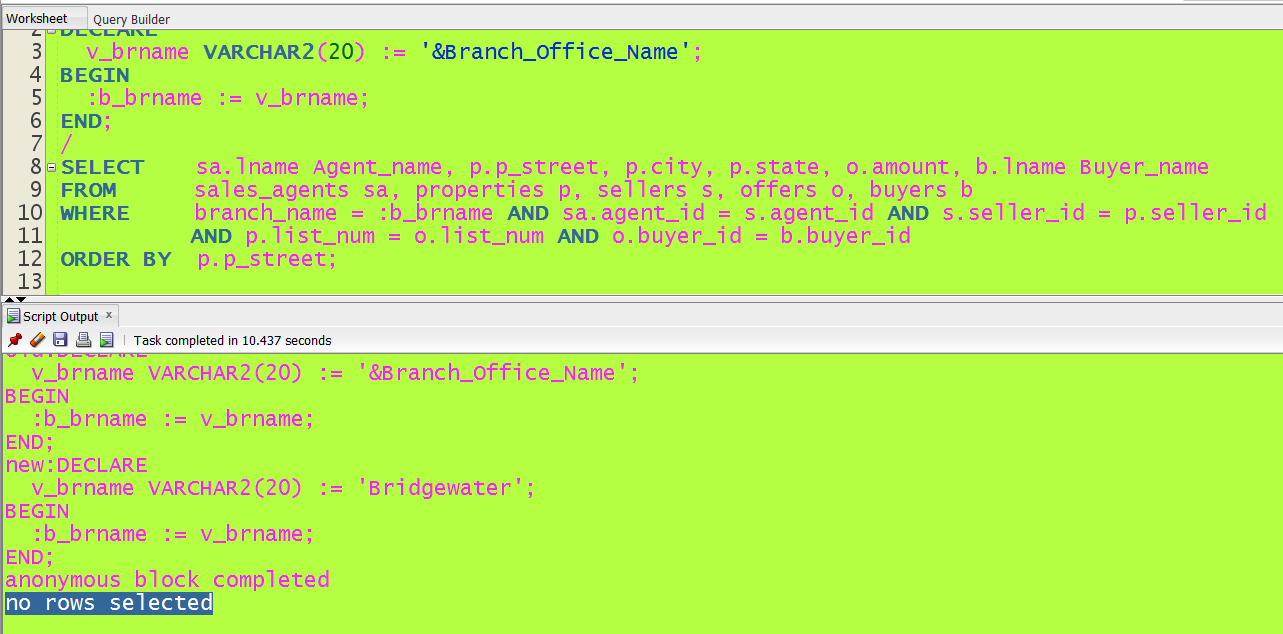


Query7: Allow a user to enter a branch name. List all the agents for the branch office, the amount of any offer placed on a house listed by any agent in the branch, the address of the house (Street, City, State) the offer was placed on and the name of the buyer that placed the offer. Order the list by Street address. Test your query with the Williamsburg branch and the Bridgewater branch. (Turn in both listings).

***Figure 18: Solution Query 7 (Brach Office: Williamsburg)***



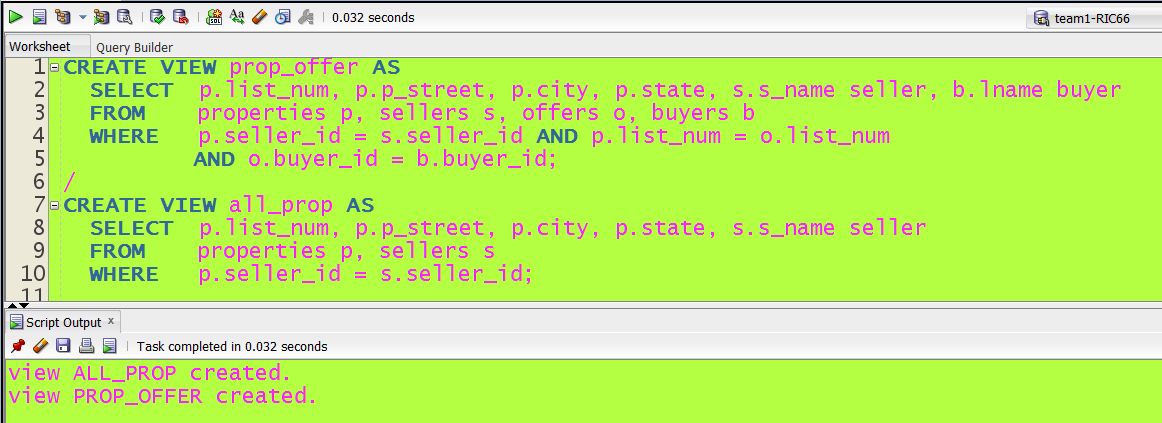
***Figure 19: Solution Query 7 (Brach Office: Bridgewater)***



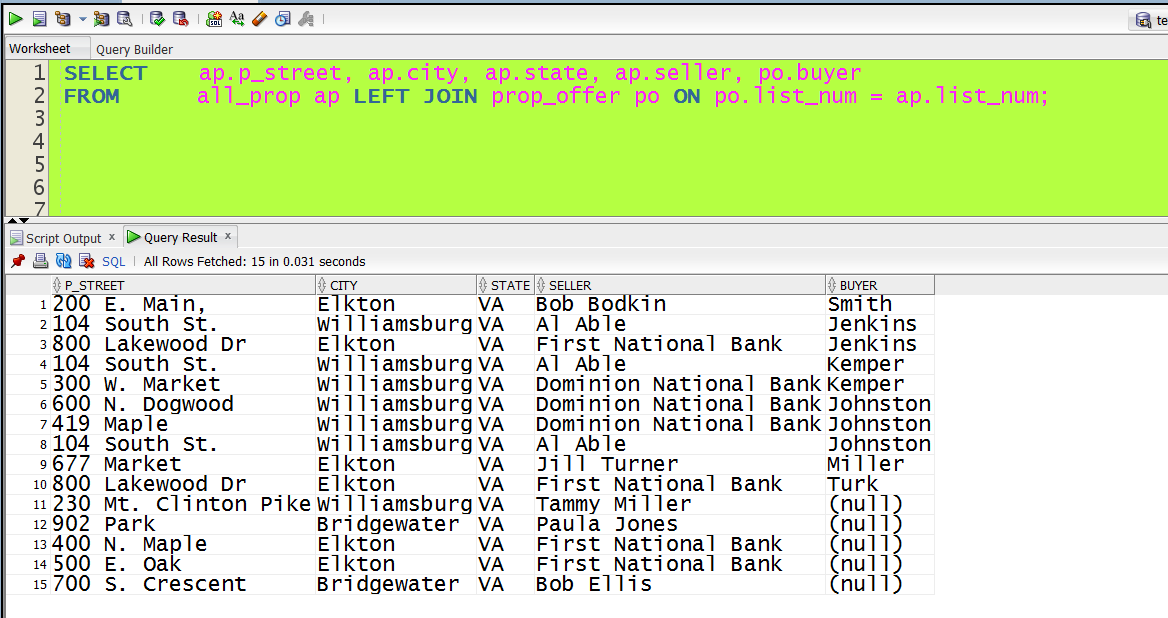
Only agent at Bridgewater is Baker and he does not have any seller; therefore, no properties

Query8: List all house addresses (Street, City, and State), the seller's name, and the name of any buyer that has an offer on the house. **Include the house address and seller's name of houses that do not have offers placed on them**.

***Figure 20: Solution Query 8, step 1 (creation of views)***

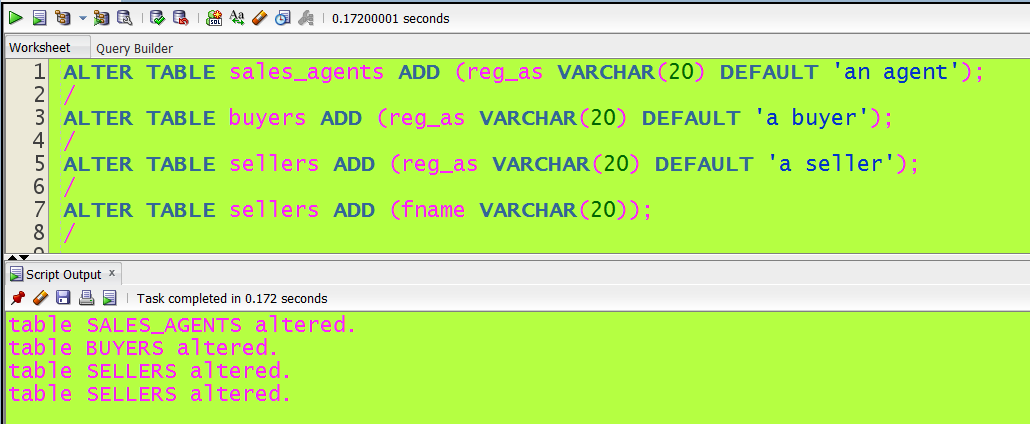


***Figure 21: Solution Query 8, step 2 (Final answer using previous views)***

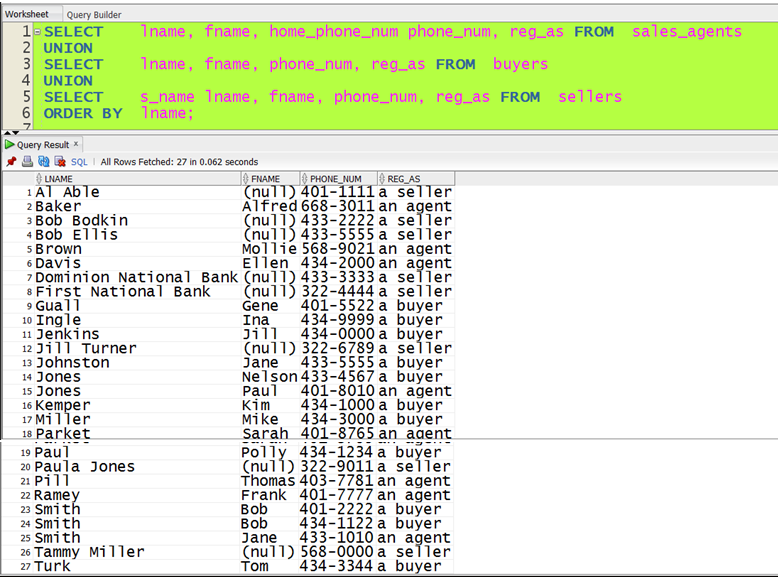


Query9: List the First Name, Last Name, and phone number of all Agents, Buyers, and Sellers. Indicate whether the person is an Agent or a Buyer. Order the list by last name.

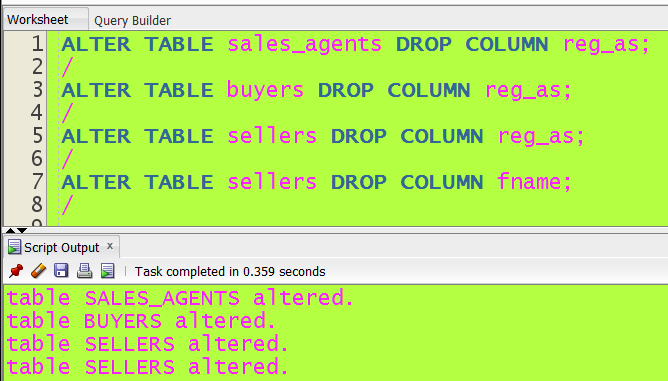
***Figure 22: Solution Query 9, step 1 (Altering tables)***



***Figure 23: Solution Query 9, step 2 (Selection and Union final answer)***

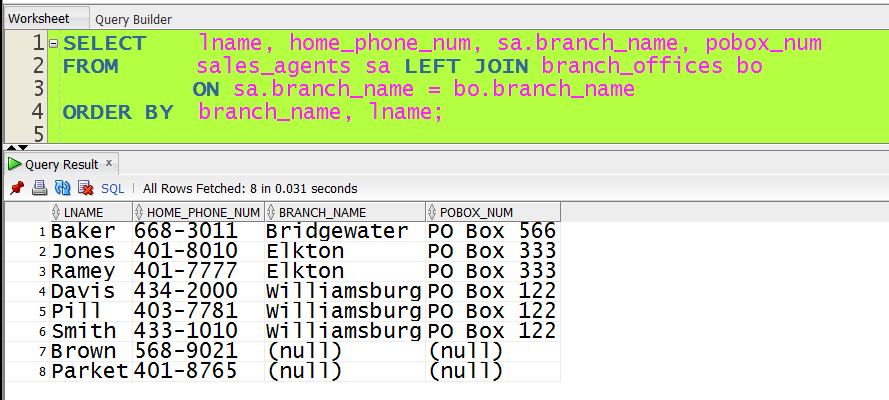


***Figure 24: Solution Query 9, step 3 (Returning to original state)***



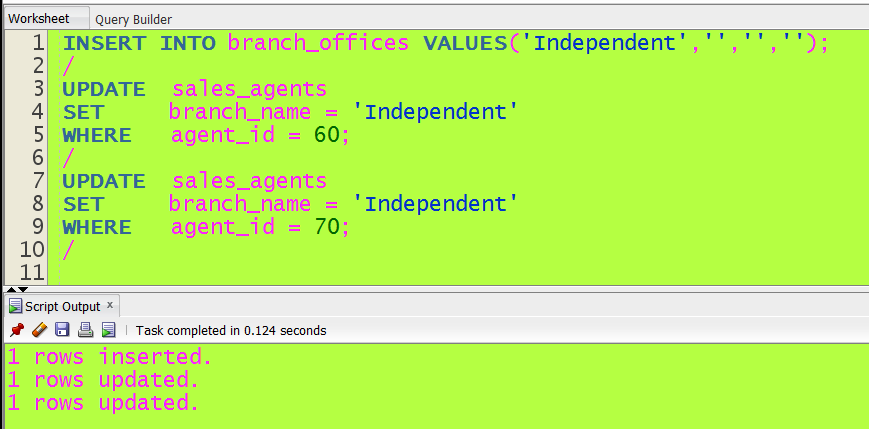
Query10: List the Name and Phone number of all agents along with the Branch Office name and the P.O. Box of the branch office the agent works for. Include all agents even if they do not work for a branch (they are independent). Order the list by Agent last name within Branch Office name.

***Figure 25: Solution Query 10***

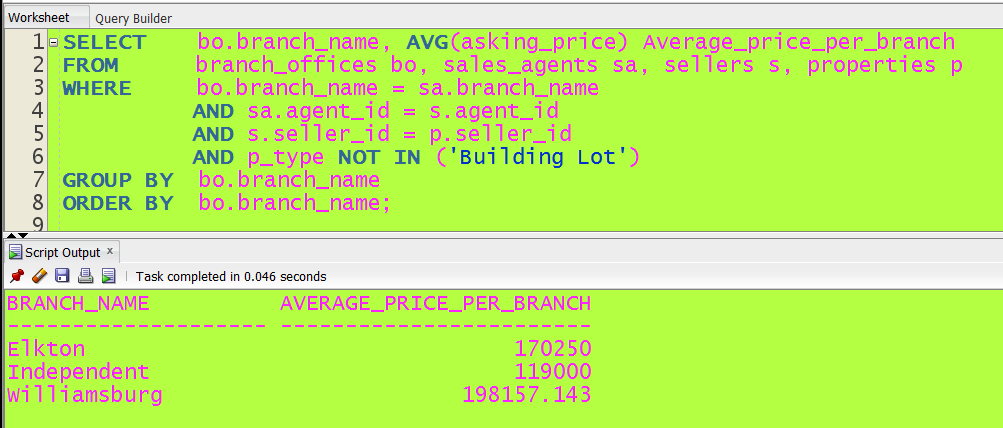


Query11: List the Branch Name of all branches and the average asking price of properties listed by agents in the branch. Do not include properties that are Building Lots in the average but do include the average of properties that are listed by agents that do not work for a branch office.

***Figure 26: Solution Query 11 step 1(Inserting and updating)***

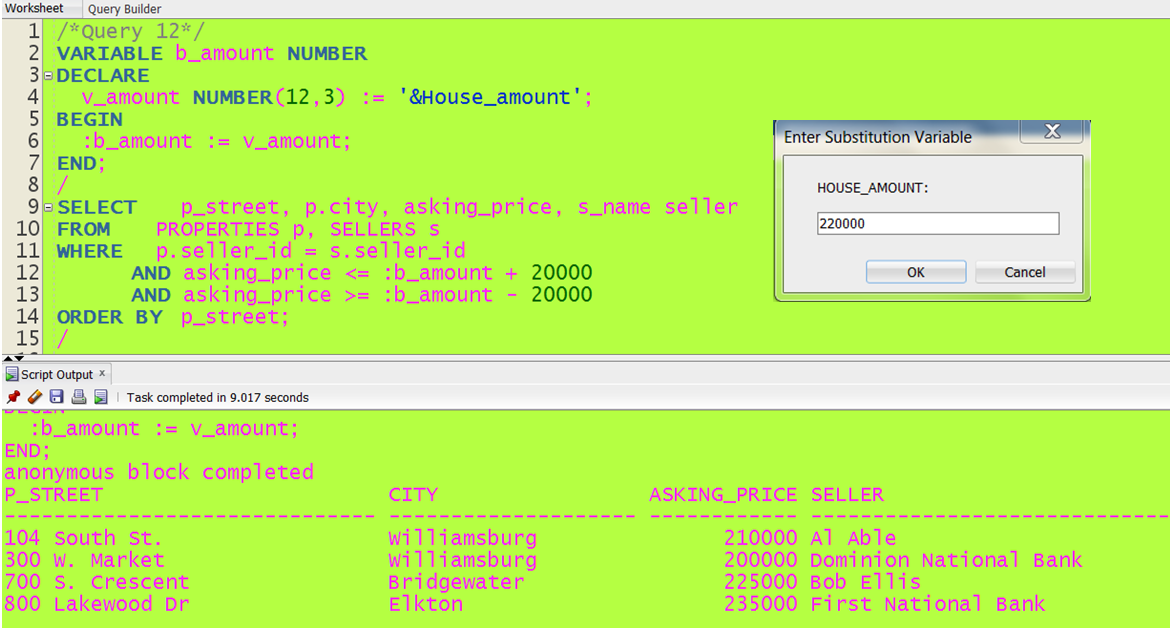


***Figure 27: Solution Query 11 step 2(Selecting and final answer)***



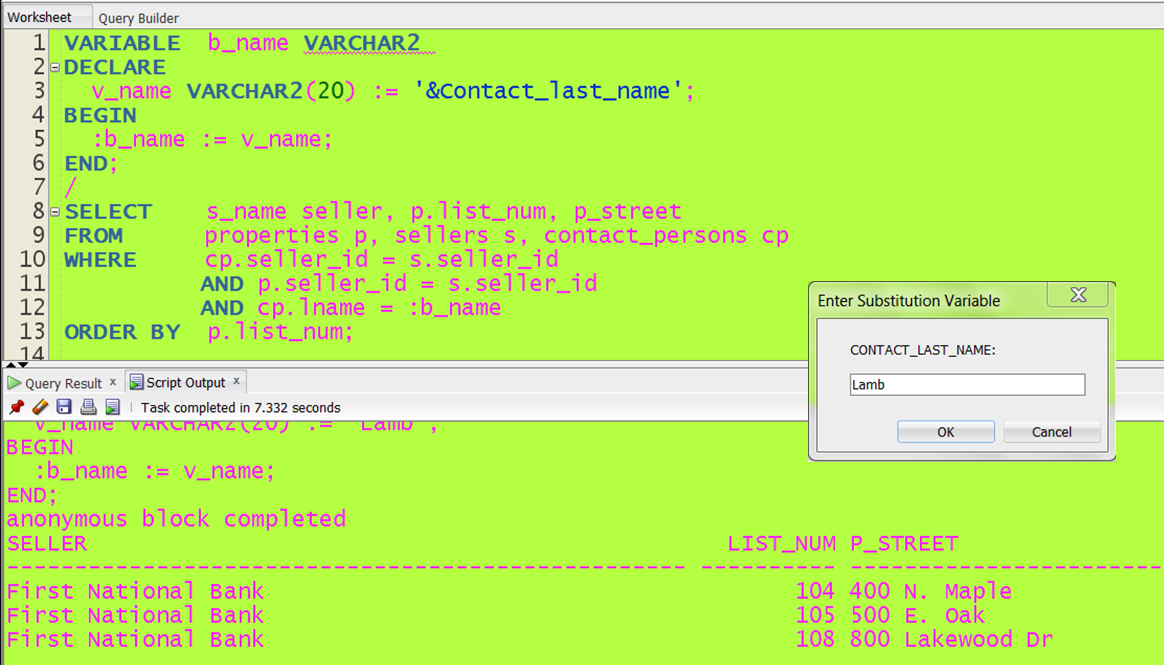
Query12: Allow a user to enter an Amount. List the Street, City, Seller Name and Asking Price of any house that is priced within $20,000 of the price entered (20,000 higher or lower). Order the list by Street. Test your query use 220,000.

***Figure 28: Solution Query 12***



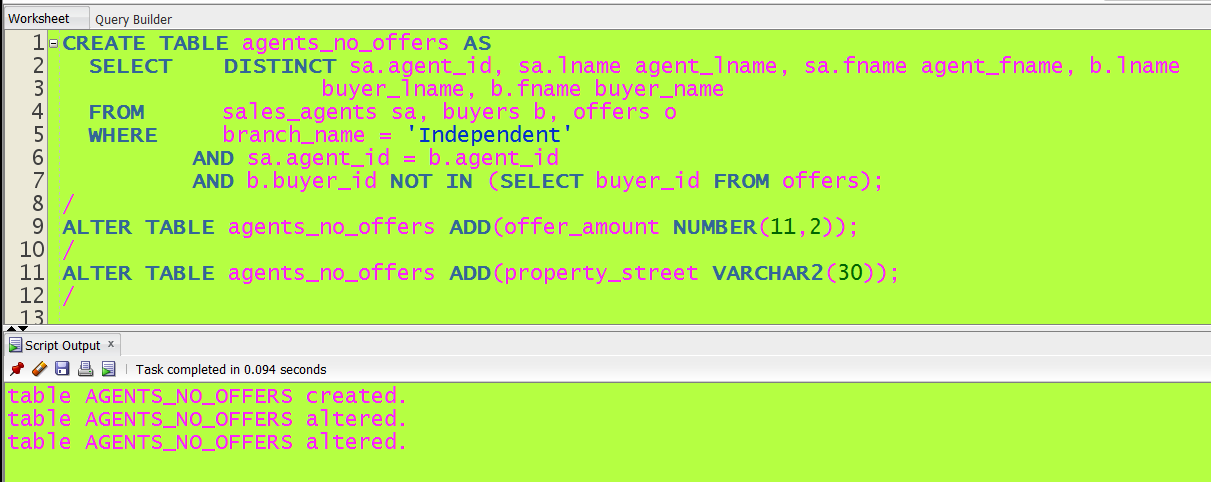
Query13: Allow a user to enter a Contact person’s name. List the seller name, listing number, and street address of all houses that Contact person is responsible for. Order the list by ‘Listing’ number.

***Figure 29: Solution Query 13***

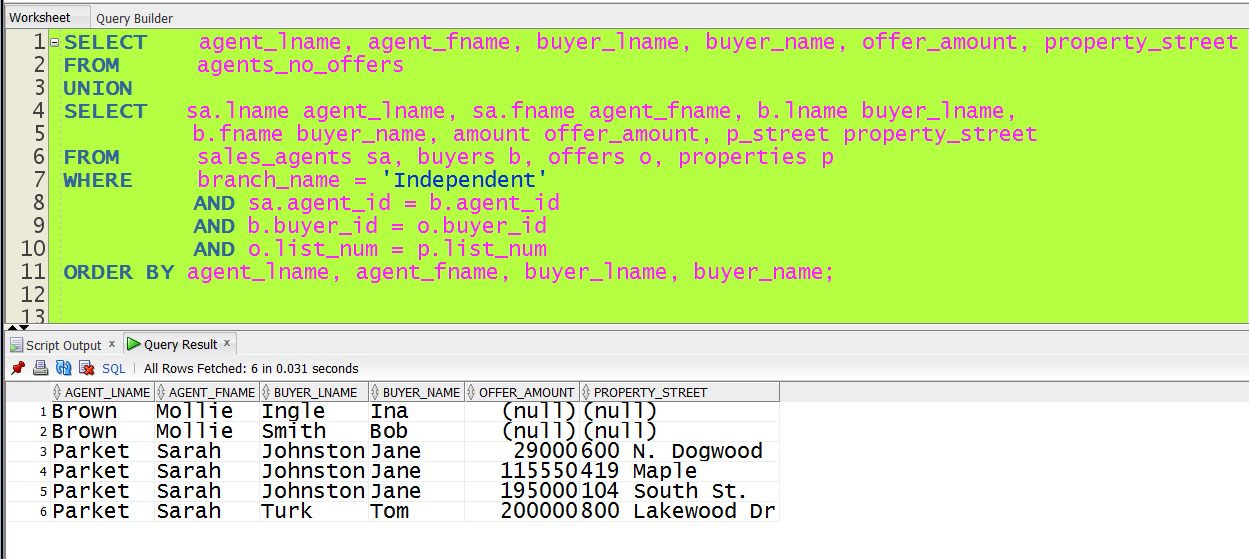


Query14: List the name of all independent agents (those not working for an agency), Buyer Name the agent represents, street address and offer price for all offers placed by the buyer. Include the name of all independent agents even if they do not have offers placed by a buyer or if they do not represent any buyers. Order the list by Buyer Name (Last, First) within Agent Name (Last, First).

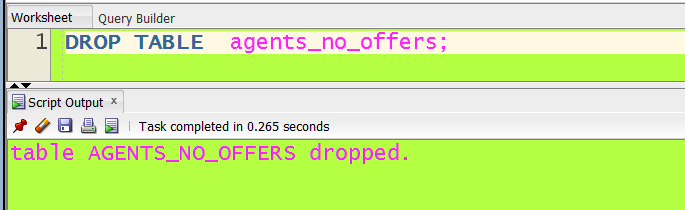
***Figure 30: Solution Query 14, step 1 (Creating a new table)***



***Figure 31: Solution Query 14, step 2(Selection and Union final answer)***

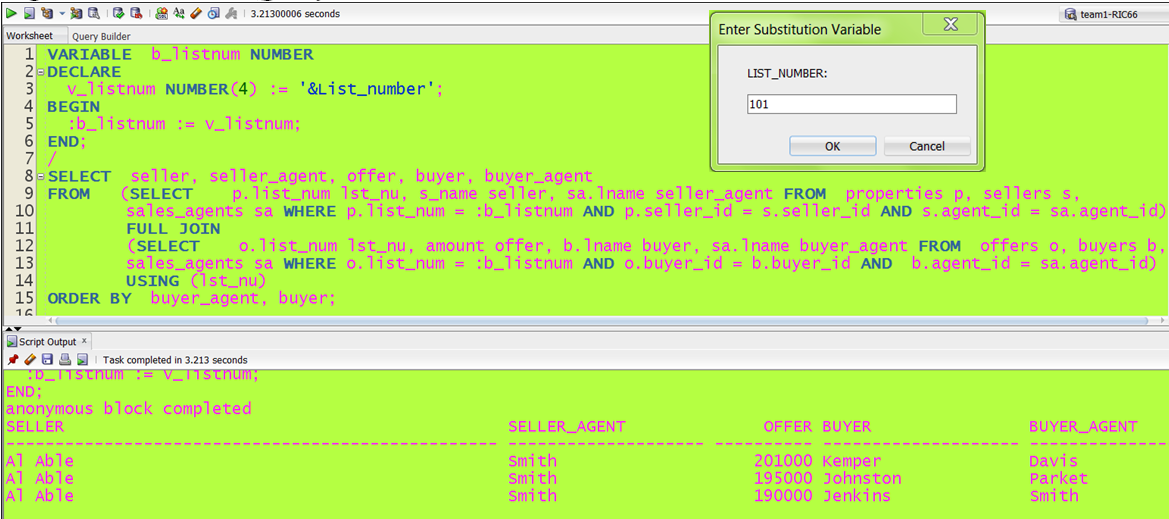


***Figure 32: Solution Query 14, step 3(Returning to original state)***



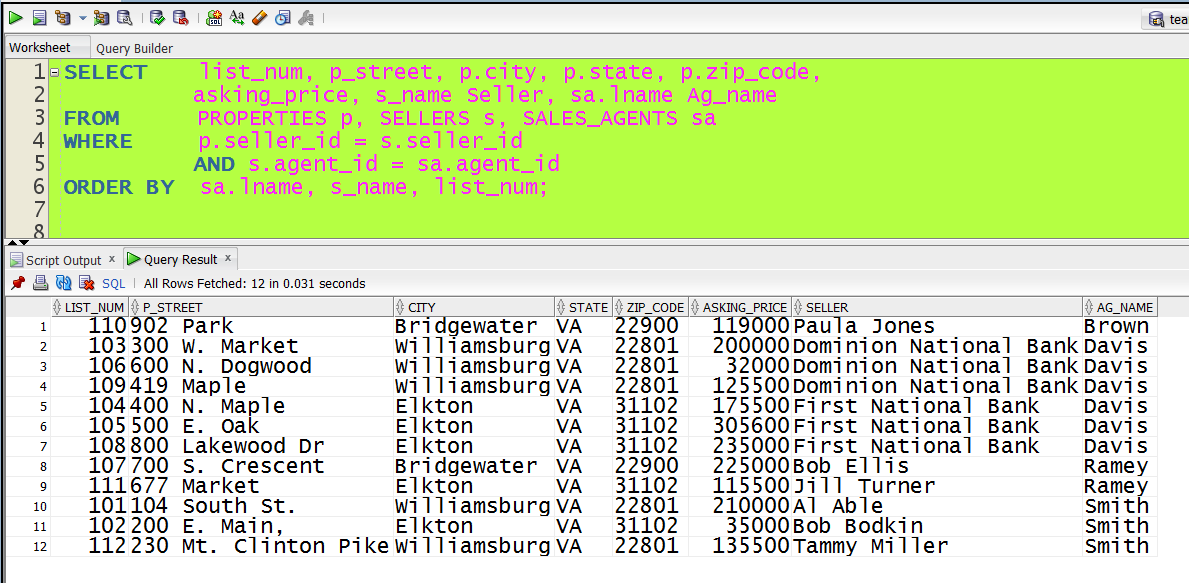
Query15: Allow a user to enter a listing number. List the name of the seller and the name of the seller’s agent, the name of the buyer and the name of the buyer’s agent, and the amount of the offer. Order the list by Buyers name within buyer’s Agent name. Test your query by entering listing number 101

***Figure 33: Solution Query 15***



**Report1:** Create a Report that all properties (Street, City, State, Zip) Asking price, Seller’s Name, and Agent’s Name. Order the report by List Id within Seller Name within Agent Name.

***Figure 34: Report 1***

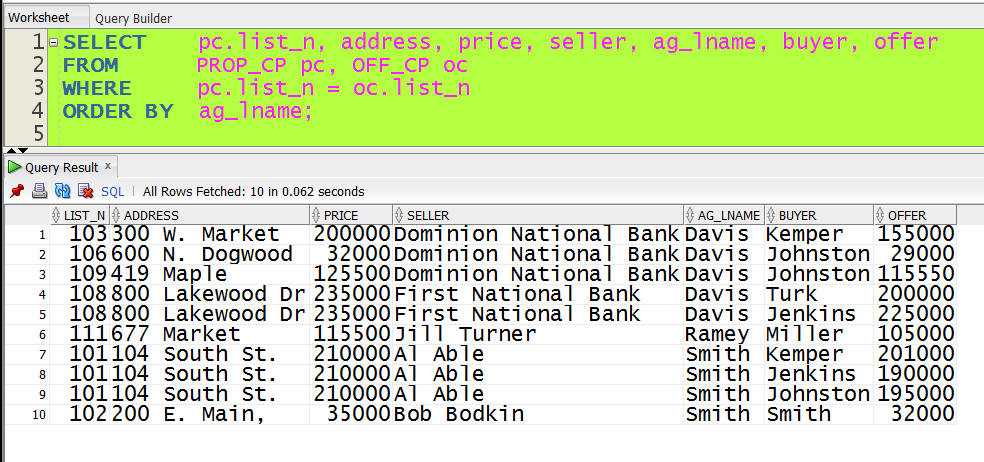


**Report2:** Create a Report that All properties (Street, Asking Price, and Picture), Seller’s Name and Buyers name and Offer amount. Order the list by Agent Name. Count the number of properties for each agent, the total of the Asking Prices for that agent, and the total of the offers for that agent.

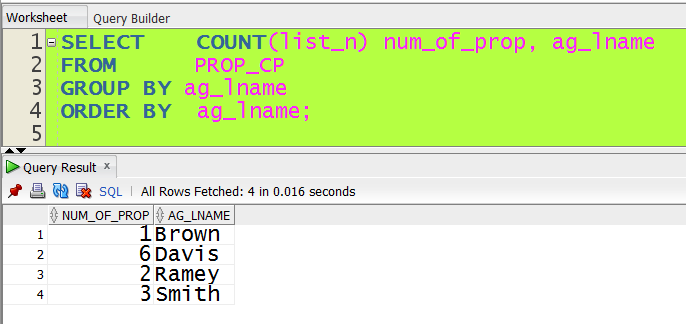
***Figure 35: Report 2, Step1 (Creation of Views)***



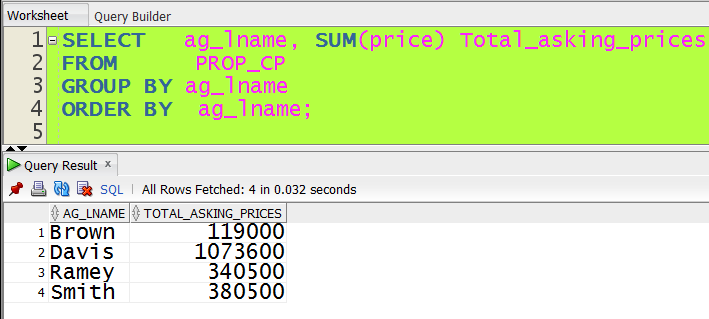
***Figure 36: Report 2, Step 2(Results)***



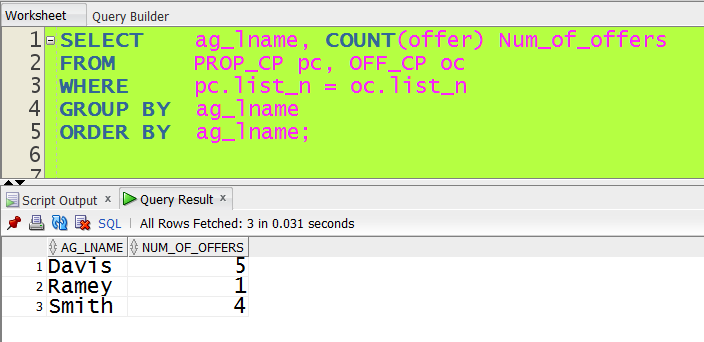
***Figure 37: Report 2, Step 3(Properties per Agent)***



***Figure 38: Report 2, Step 3 (Sum of Asking Prices per Agent)***



***Figure 39: Report 2, Step 4 (Total Offers per Agent)***



# 4.4 Lessons Learned in Part IV

In this section we have learned the processes of normalization and optimization to analyzed the relations among attributes in each entity and establish improvements in their structure. As we mentioned in previous sections, every time that in our iterative approach we generate a new arrangement of entities and relationships, normalization and optimization were the following obligated steps. After models were generated and improved, we weighted each alternative to select the optimal model. The criteria used were model efficiency and accessibility to information requited by the stakeholders.

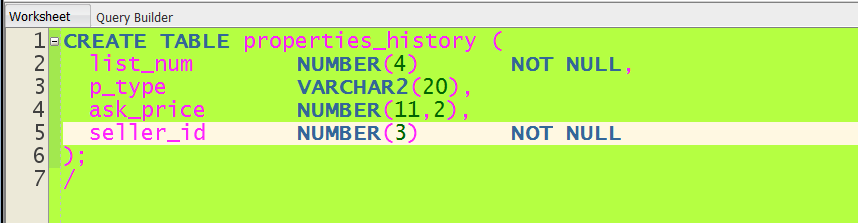
We also learned how to insert information systematically into the entities and how to resolve the queries and reports. By resolving all the queries we assured that the model was working correctly and opened our imagination to interfaces ideas.

**PART 5**

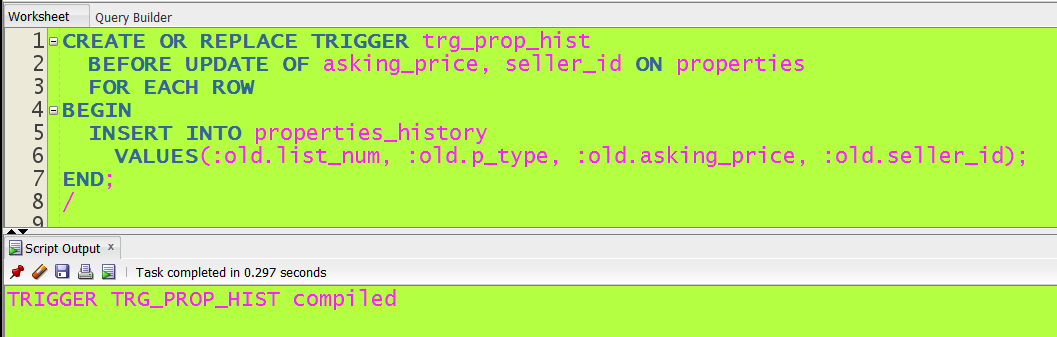
# 5.1 History Triggers

1- ) Write a Database Triggers to write into a history table the rows from your most critical table before an update; this will be a row level trigger that will save the rows that are being updated before the update occurs. For example, for a table EMPLOYEES, you add a table EMPLOYEES\_HIST that has the same schema as the table EMPLOYEES an contains the row that is about to be changed (the one contained in the “:old” record in a before update row level trigger).

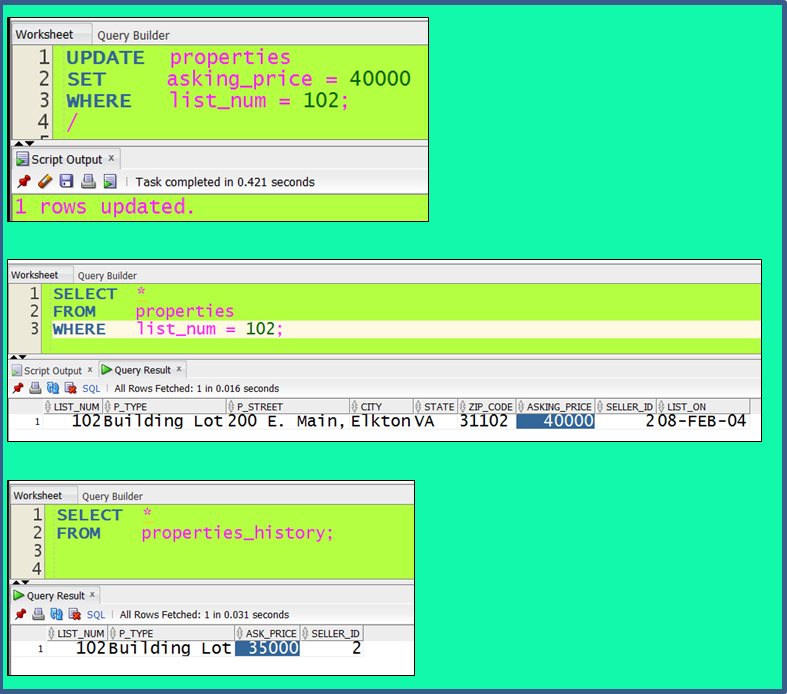
***Figure 40: Solution Trigger 1, step 1(Creation of history table)***



***Figure 41: Solution Trigger 1, step 2(Creation of trigger)***



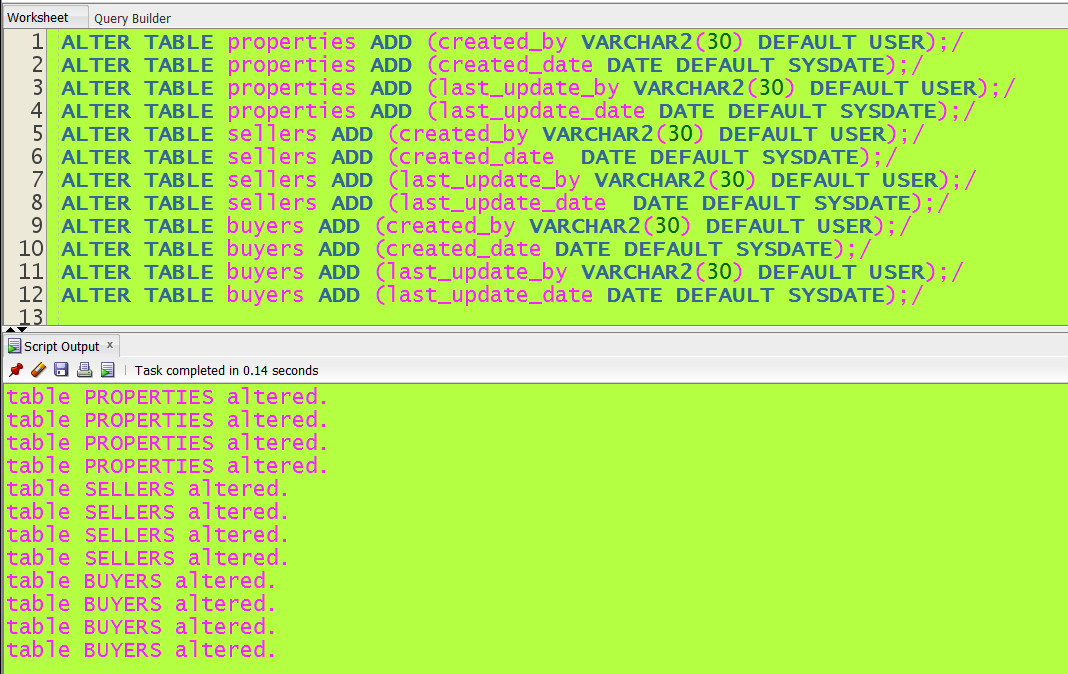
***Figure 42: Solution Trigger 1, step 3(Testing Triggers)***



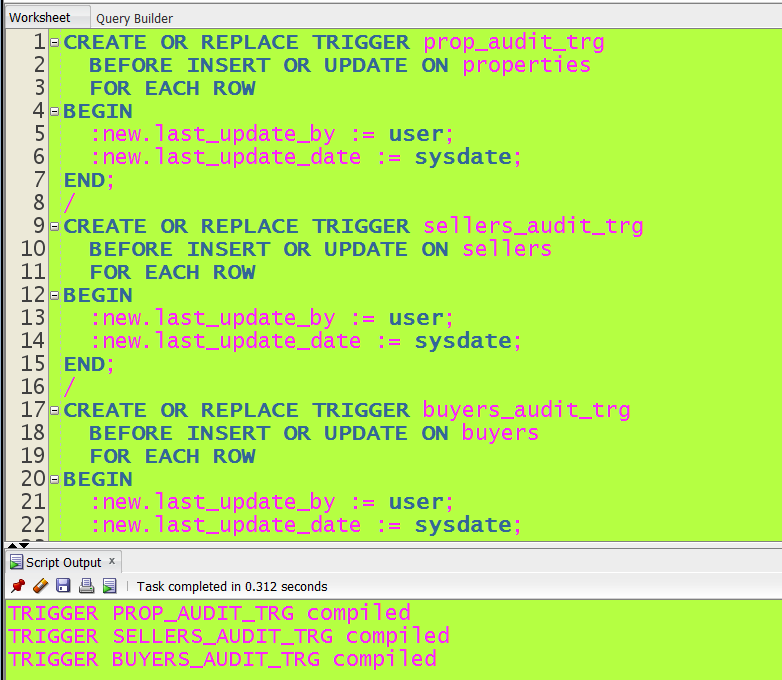
# 5.2 Auditing Triggers

1- ) Add audit columns to at least three tables in your database schema; the new columns will be CREATED\_DATE, CREATED\_BY, UPDATED\_DATE, UPDATED\_BY. The values for these three columns need to be generated by a ROW LEVEL TRIGGER after all INSERTS and UPDATES.

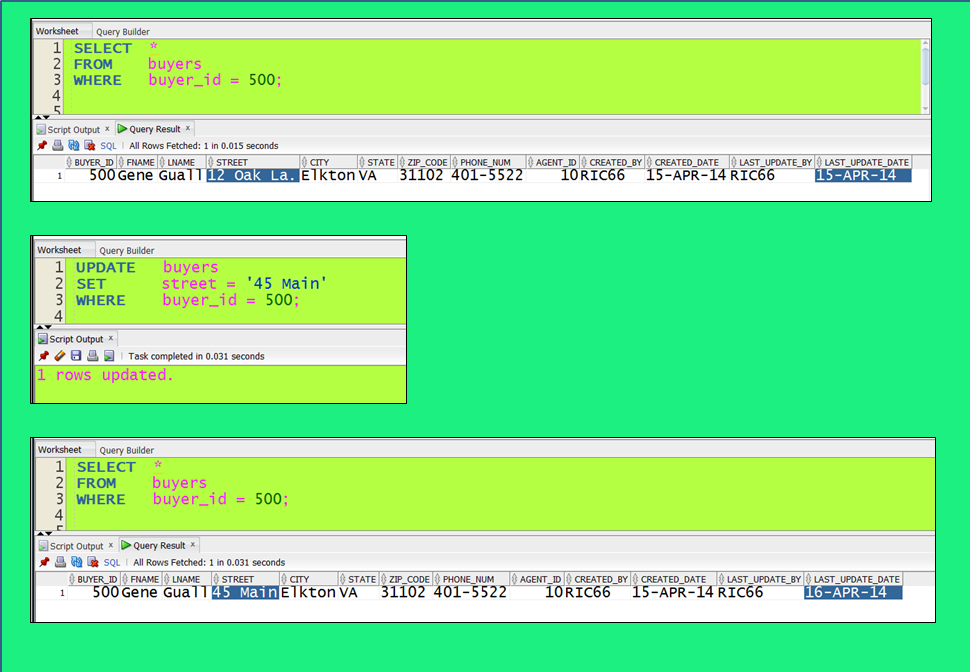
***Figure 43: Solution Trigger, step 1(Creation of columns)***



***Figure 44: Solution Trigger, step 2(Creation of Triggers)***



***Figure 45: Solution Trigger, step 3(Testing Triggers)***



# 5.3 Prototypes

In order to represent all the forms and interfaces, we have designed and developed two different prototypes: a website PHP based prototype and a Java-GUI prototype

# 5.3.1 Website Prototype

This prototype was constructed using PHP-MySQL languages hosted in a commercial website that belongs to one of the members of our group. The address where anyone can test its functionality is:

<http://www.blesolutions.com/wp-content/themes/bleSite.1.4.2/dbhome.php>

At this link the customer interface is shown in which there is a radio button selector to choose among properties, branch offices or sales agents. Once one of them is displayed, it comes automatically with an independent filter designed exclusively for each table where the customer can introduce different values to obtain selected information about each table.

At the main menu level “Agents” and “Managers” are activated to display two more interfaces. Agents for the agents working area and Managers for the system administration facility, the first one displays a joined table from properties, sales agents and sellers and two filters to facilitate agents to find information quickly. The system administration facility allows the managers or authorized personnel to edit, delete or add more records to the tables. The Figure 40 shows the website going through the 3D-testing phase using Firefox tools.

***Figure 46: Website Prototype (Testing Phase)***

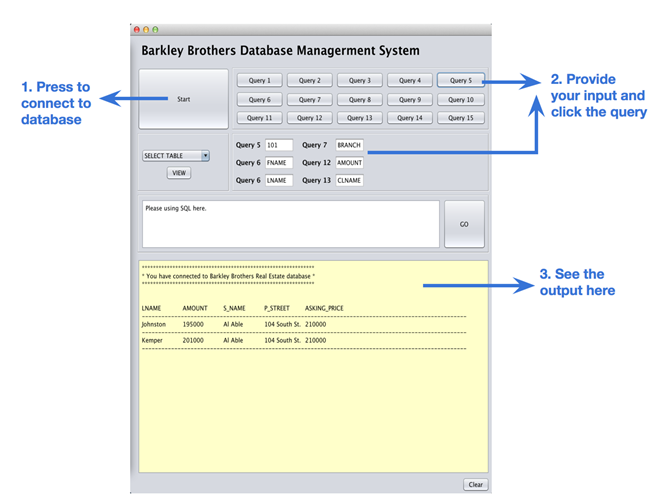


# 5.3.2 Java-GUI Prototype

This user interface is developed on Java Swing and Oracle JDBC drive. It intends to help the employees accessing the data and managing the database more easily and efficiently. All those most common used functions are all combined together on this prototype. Using this prototype, user can connect to the database by simply press the Start button. Other functions include:

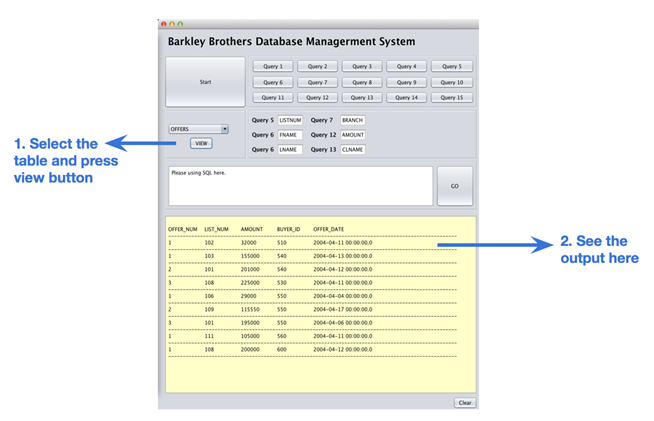
1) Fast execute the build in Queries. The user could get rid of typing SQL query for most of the time.

***Figure 47a: Java-GUI Prototype Output I***



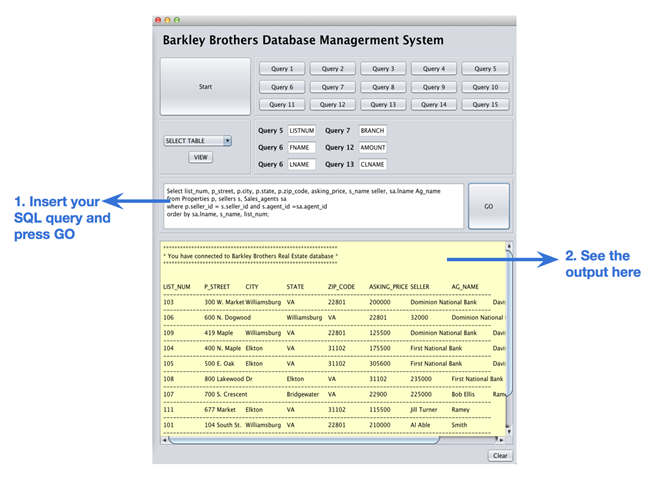
2) Select each table and view all data in that table.

***Figure 47b: Java-GUI Prototype Output II***



3) Insert SQL and fetch data form the database. The user will could have more flexibility and the ability to modify the tables.

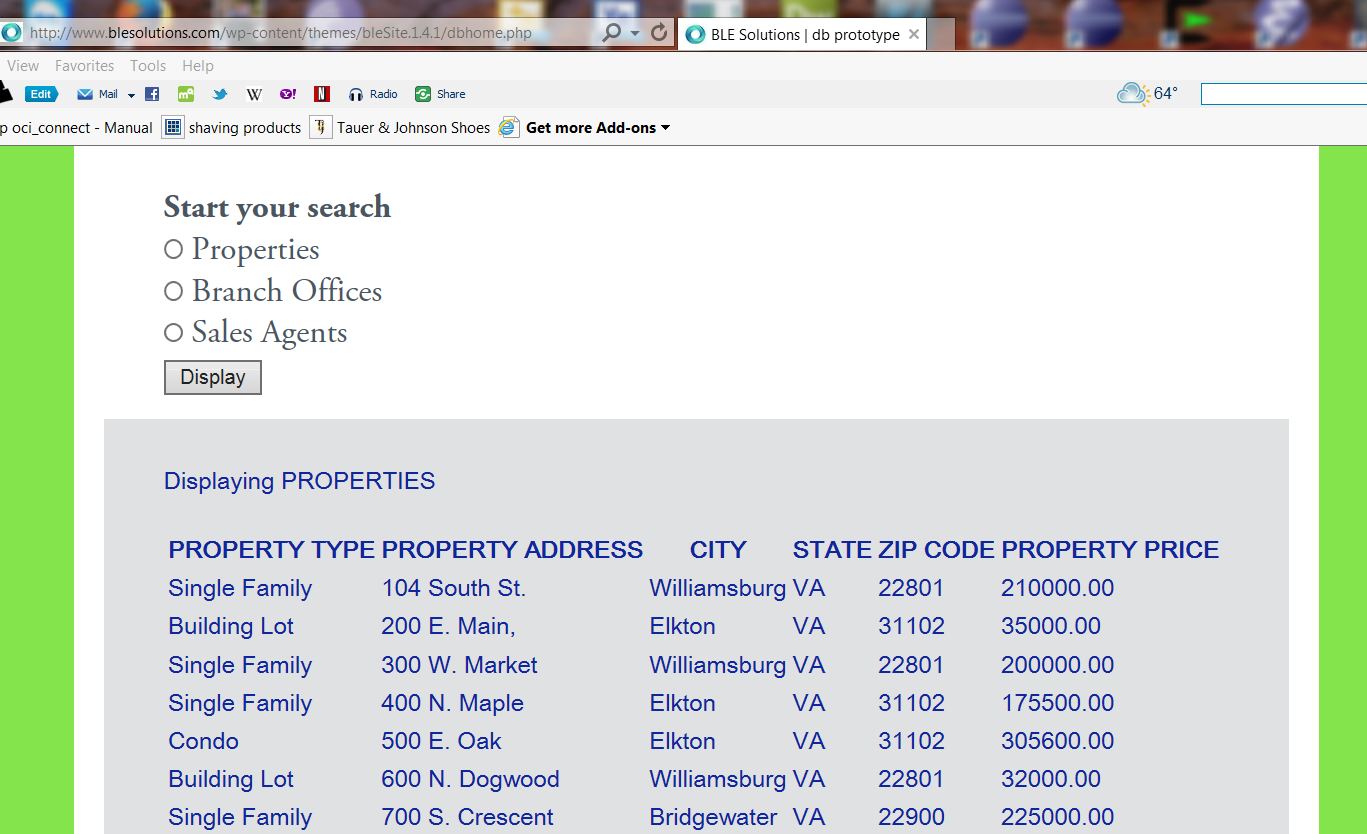
***Figure 47c: Java-GUI Prototype Output III***



# 5.4 Front-end Customers Interface

**Form 1:** Create a form that will allow a user to view all properties in the database.

***Figure 48: Website Display of Form 1 (External Customer Interface)***



This form was created in our website prototype, and the specific form is located at this address:

<http://www.blesolutions.com/wp-content/themes/bleSite.1.4.2/dbhome.php>

The encrypted code that cannot be seen at the website is annexed in the following file, double

Click on it to open the file:

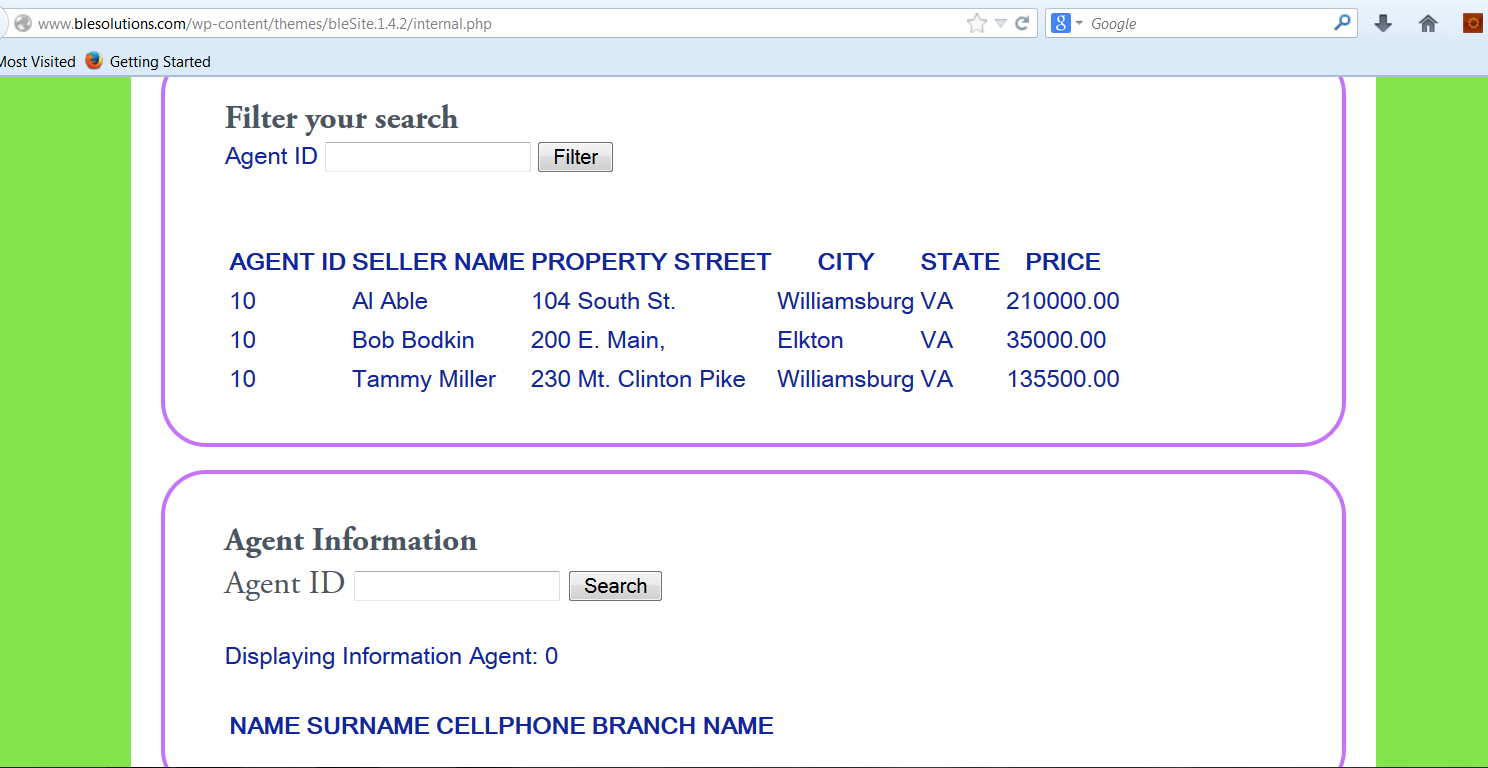


**Form 2:** Create a form that will allow a user to enter an Agents’s Id. The form should display the Agent’s Name (first and Last), the Agent’s Branch if they work for one, and the Agent’s phone number. In a sub-form, list the Street, City, State, Asking Price, and Seller name of any property the Agent is selling. Test the form with agent Ellen Davis to turn in.

This form was created in our website prototype, and the specific form is located at this address:

<http://www.blesolutions.com/wp-content/themes/bleSite.1.4.2/internal.php>

***Figure 49: Form 2(Employees Interface)***



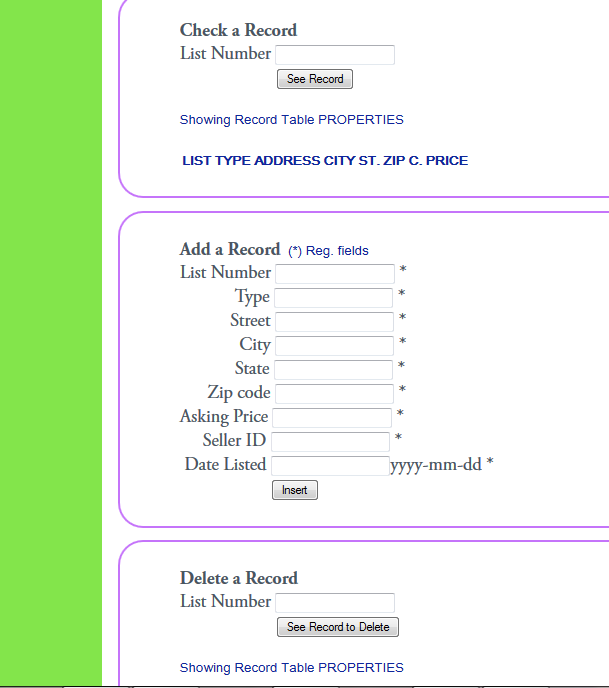
# 5.5 System Administration Facility

1. You must design and implement a system administration facility to add, delete, or update information about Branch Offices, Sellers, Sales Agents, Buyers and Properties.

The system administration facility is constructed into the website prototype, and the specific interface is available at:

<http://www.blesolutions.com/wp-content/themes/bleSite.1.4.2/internalMgr.php>

***Figure 50: System Administration Facility (Website Prototype)***



# 5.5 SQL Scripts

In this section we are collecting all scripts generated within the project to easy the grading of our work. We are separating the scripts in order of execution starting by the construction of tables, followed by population of tables, then the code to resolve 15 queries and two reports and finally construction of triggers (history and auditing). Double click on each icon to open the files.

Tables Structures:



Population of tables:



Queries and Reports:



History trigger and auditing triggers:



# 5.6 Lessons Learned in Part V

In this section we have learned the important concept of automation of business procedures and automatically maintaining data integrity and consistency. By using the triggers in data history and auditing we have practice how to implement these concepts that are very useful in architecting real life databases.

We have also learned how to implement two different prototypes in real life using totally different approaches, but based on the same construction concept. As a result of these two experiments we feel completely loaded to the necessary tools and knowledge to start any database endeavor.

# CONCLUSIONS

Through the development of this project we have designed and implemented an Oracle database using the business rules, Oracle software and information proportioned in class and referenced books. We now clearly understand how important and critical are the management of data for any organization and the implications of good or bad information proportioned to the decision makers. Quality of data which is the result of an efficiently database model is a comprehensive approach to ensure accuracy, validity and timeliness of data that might be used for critical decisions that will determine the future of an organization.

We have extracted the most of this project by creating a template for architecting a systematic approach to construct the optimal database model for any new database construction process. Our approach has to be tune-up while developing the next (probably real life) database projects. What is guarantee in the next endeavor is that the time to develop the new database will be considerable reduced and the template upgraded for better future functionality.

# BIBLIOGRAPHY

Coronel C, Morris S, and Rob Peter, Database Systems: Design, Implementation, and Management, Tenth Edition. USA: Joe Sabatino, 2013.

Ramakrishnan R, Gehrke J, Database Management Systems, Third Edition. New York: McGraw-Hill, 2003.

Reges S, Stepp M, Building Java Programs: A Back to Basics Approach, second Edition. USA: Addison-Wesley, 2011.

Nixon Robin, Learning PHP, MySQL, JavaScript, & CSS: A Step by Step Guide to Creating Dynamic Websites, Second Edition. Cambridge USA: O’Reilly, 2012.

Oracle and/or its affiliates, Oracle Data Modeling and Relational Database Design: Activity Guide, Edition 1.0 September 2010.

Oracle and/or its affiliates, Oracle Data Modeling and Relational Database Design: Volume I Student Guide, Edition 1.0 September 2010.

Oracle and/or its affiliates, Oracle Data Modeling and Relational Database Design: Volume II Student Guide, Edition 1.0 September 2010.

Oracle and/or its affiliates, Oracle Database PL/SQL Fundamentals: Student Guide, Edition 1.1 March 2012.

Oracle and/or its affiliates, Oracle Database SQL Fundamentals I: Activity Guide, Edition 1.1 March 2012.