# Project for Database Design

# Phase III. Implementation

Indhumadhi Suryanarayanan ixs150630@.utdallas.edu

Sindhujaa Ranganathan Chandra Babu sxr152530@.utdallas.edu

Suneesha Kudipudi sxk157430@.utdallas.edu

(Week 11-15: Nov.1-Nov.30)

### 0. Pre-Illumination

For clearly describing the implementation of our database, we separate this report into four sections. In Section 1 we normalize the original relational schema into third normal form and changed part of our relational schema because of some requirement from Phase III. We then explained what are changed. In Section 2 we drew a dependency diagram for each relation table one by one. In Section 3 we began our process of building a database in Oracle using SQL statements, which contains three parts. Part one is the creation of database, including tables, all other structures as well as data type and format, Part two is the creation of views corresponding to five distinct requirements from Question d, and Part three is the creation of Queries to satisfy 14 requirements from Question e. Finally, a short summary is given at the end of this report.

# 1. Modified Relational Schema

The table SUBDIVISION has to be normalized to 3NF since it had a transitive dependency of NSME->MANAGER\_ID, MANAGER\_ID->START\_DATE.

Hence it had to be decomposed as follows:

#### **SUBDIVISION:**

<u>NAME</u>	START_YEAR	RANK	ZIP	MANAGER_ID

#### **MANAGER:**

MANAGER_ID	START_DATE

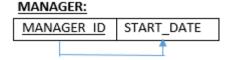
Figure 1: 3NF Normalized tables

Rests of the tables remain the same.

# 2. Dependency Diagram

We now draw a dependency diagram for each tables in the below sections.

# 2.1 Dependency in MANAGER



# 2.2 Dependency in LOT



# 2.3 Dependency in FLOORPLAN



# 2.4 Dependency in INVENTORY



# 2.5 Dependency in SUBDIVISION

#### SUBDIVSION:



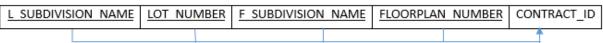
# 2.6 Dependency in TOURS

#### TOURS:

EMP ID	CUSTOM	IER EMAIL	I SUBDIVI	SION NAME	INVEN	TORY ID	TOUR	DATE

# 2.7 Dependency in ASSOCIATES

#### ASSOCIATES:



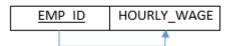
# 2.8 Dependency in EMPLOYEE

### EMPLOYEE:



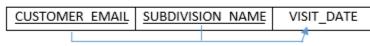
# 2.9 Dependency in SECRETARY

#### SECRETARY:



# 2.10 Dependency in VISITS

#### VISITS:



# 2.11 Dependency in SIGNS

#### SIGNS:



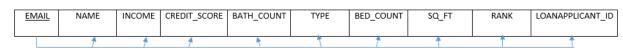
# 2.12 Dependency in CONTRACT

#### CONTRACT:



# 2.13 Dependency in CUSTOMER

#### CUSTOMER:



# 2.14 Dependency in LOAN



### 2.15 Final Results

Figure 2.15 shows the final results for the whole database including the ones who do not have any functional dependencies.

# MANAGER ID



#### LOT:



#### FLOORPLAN:



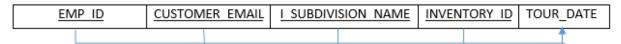
#### INVENTORY:



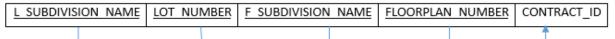
#### SUBDIVSION:



#### TOURS:



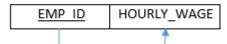
#### ASSOCIATES:



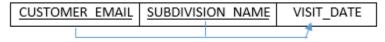
#### EMPLOYEE:



#### SECRETARY:



#### VISITS:



#### SIGNS:



#### CONTRACT:



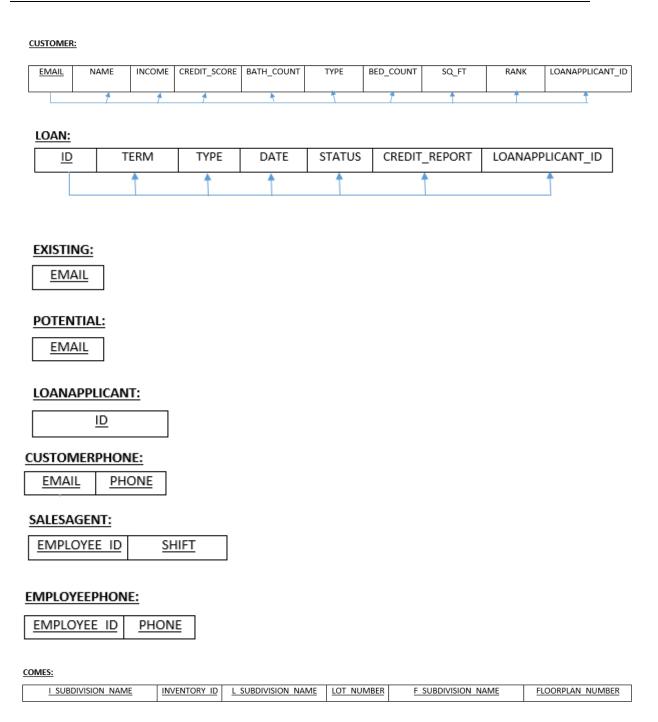


Figure 2.15: Whole Dependency Diagram for Home Builder Company Database

# 3. Implementation of Database

# 3.1 Creation of Database with SQL Statements

After normalizing every relational schema into third normal form and modifying some

details, it is the time to implement our database using SQL languages into Oracle.

#### 3.1.1 Table Creation

Using SQL statement, we created tables as follows:

CREATE TABLE MANAGER (MANAGER\_ID INT NOT NULL, START\_DATE DATE, CONSTRAINT MANAGER\_PK PRIMARY KEY (MANAGER\_ID));

CREATE TABLE SUBDIVISION(NAME VARCHAR(255) NOT NULL, START\_YEAR INT ,RANK INT, ZIP INT, MANAGER\_ID INT);

ALTER TABLE SUBDIVISION ADD CONSTRAINT PK SUB PRIMARY KEY (NAME):

ALTER TABLE SUBDIVISION ADD CONSTRAINT FK\_MANAGER FOREIGN KEY (MANAGER\_ID) REFERENCES MANAGER (MANAGER\_ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE SUBDIVISION ADD CONSTRAINT CK\_MAN UNIQUE(MANAGER\_ID);

CREATE TABLE LOANAPPLICANT (ID INT NOT NULL, CONSTRAINT LOANAPP\_PK PRIMARY KEY (ID));

CREATE TABLE LOT (SUBDIVISION\_NAME VARCHAR(255), LOT\_NUMBER INT UNIQUE, LOT\_SIZE INT, FACING VARCHAR(20) CHECK (FACING IN('NORTH', 'SOUTH', 'EAST', 'WEST')), AVAILABILITY VARCHAR(5) CHECK (AVAILABILITY IN ('YES', 'NO')));

ALTER TABLE LOT ADD CONSTRAINT SUB\_FK FOREIGN KEY (SUBDIVISION\_NAME) REFERENCES SUBDIVISION(NAME) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE LOT ADD CONSTRAINT LOT\_PK PRIMARY KEY (SUBDIVISION\_NAME, LOT\_NUMBER);

CREATE TABLE FLOORPLAN (SUBDIVISION\_NAME VARCHAR(255), FP\_NUMBER NUMBER(4,0) .

FLOOR\_COUNT INT CHECK(FLOOR\_COUNT>=1 AND FLOOR\_COUNT<=3), BED\_COUNT INT CHECK(BED\_COUNT>=1 AND BED\_COUNT<=5), BATH\_COUNT INT CHECK(BATH\_COUNT>=1 AND BATH\_COUNT<=5), NAME VARCHAR(255), PRICE INT, SQ\_FT NUMBER(4,0), GARAGE\_COUNT INT CHECK(GARAGE\_COUNT>=1 AND GARAGE COUNT<=3));

ALTER TABLE FLOORPLAN ADD CONSTRAINT FP\_FK FOREIGN KEY (SUBDIVISION\_NAME) REFERENCES SUBDIVISION(NAME) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE FLOORPLAN ADD CONSTRAINT FP\_PK PRIMARY KEY (SUBDIVISION\_NAME, FP\_NUMBER);

CREATE TABLE INVENTORY(SUBDIVISION\_NAME VARCHAR(255), ID INT, PRICE INT, MOVE\_IN\_DATE DATE);

ALTER TABLE INVENTORY ADD CONSTRAINT INVENT\_FK FOREIGN KEY (SUBDIVISION\_NAME) REFERENCES SUBDIVISION(NAME) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE INVENTORY ADD CONSTRAINT IN\_PK PRIMARY KEY (SUBDIVISION NAME, ID);

CREATE TABLE COME( I\_SUBDIVISION\_NAME VARCHAR(255), INVENTORY\_ID INT, L\_SUBDIVISION\_NAME VARCHAR(255), LOT\_NUMBER INT, F\_SUBDIVISION\_NAME VARCHAR(255), FLOORPLAN\_NUMBER INT);

ALTER TABLE COME ADD CONSTRAINT IN\_COME\_FK FOREIGN KEY (I\_SUBDIVISION\_NAME,INVENTORY\_ID) REFERENCES INVENTORY(SUBDIVISION\_NAME,ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE COME ADD CONSTRAINT L\_COME\_FK FOREIGN KEY (L\_SUBDIVISION\_NAME,LOT\_NUMBER) REFERENCES LOT(SUBDIVISION\_NAME,LOT\_NUMBER) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE COME ADD CONSTRAINT F\_COME\_FK FOREIGN KEY (F\_SUBDIVISION\_NAME,FLOORPLAN\_NUMBER) REFERENCES FLOORPLAN(SUBDIVISION\_NAME,FP\_NUMBER) ON DELETE CASCADE INITIALLY DEFERRED:

ALTER TABLE COME ADD CONSTRAINT COME\_PK PRIMARY KEY ( I\_SUBDIVISION\_NAME, INVENTORY\_ID, L\_SUBDIVISION\_NAME, LOT\_NUMBER, F\_SUBDIVISION\_NAME, FLOORPLAN\_NUMBER);

CREATE TABLE EMPLOYEE (ID INT, FNAME VARCHAR(255), LNAME VARCHAR(255), INT CHECK(YEARS OF EXPERIENCE>=0 YEARS OF EXPERIENCE AND YEARS OF EXPERIENCE<=55), E TYPE VARCHAR(20) CHECK IN (E TYPE ('SECRETARY','SALES\_AGENT')), **SALARY** DECIMAL(10,2), SUBDIVISION\_NAME VARCHAR(255), LOANAPPLICANT\_ID INT);

ALTER TABLE EMPLOYEE ADD CONSTRAINT EM\_SUB\_FK FOREIGN KEY (SUBDIVISION\_NAME) REFERENCES SUBDIVISION(NAME) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE EMPLOYEE ADD CONSTRAINT EM\_LA\_FK FOREIGN KEY (LOANAPPLICANT\_ID) REFERENCES LOANAPPLICANT(ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE EMPLOYEE ADD CONSTRAINT EMPLOY PK PRIMARY KEY (ID);

CREATE TABLE EMPLOYEE (ID INT, FNAME VARCHAR(255), LNAME VARCHAR(255), YEARS OF EXPERIENCE INT CHECK(YEARS\_OF\_EXPERIENCE>=0 **AND** YEARS\_OF\_EXPERIENCE<=55), E\_TYPE VARCHAR(20) **CHECK**  $(E_TYPE)$ IN ('SECRETARY', 'SALES\_AGENT')), SUBDIVISION\_NAME **SALARY** DECIMAL(10,2), VARCHAR(255), LOANAPPLICANT ID INT);

CREATE TABLE CONTRACT (ID INT , SIGN\_DATE DATE, SALE\_PRICE INT ,EMPLOYEE\_ID INT);

ALTER TABLE CONTRACT ADD CONSTRAINT CON\_FK FOREIGN KEY (EMPLOYEE\_ID) REFERENCES EMPLOYEE(ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE CONTRACT ADD CONSTRAINT CON\_PK PRIMARY KEY (ID);

CREATE TABLE ASSOCIATES(L\_SUBDIVISION\_NAME VARCHAR(255), LOT\_NUMBER INT,

F\_SUBDIVISION\_NAME VARCHAR(255), FLOORPLAN\_NUMBER INT, CONTRACT\_ID INT);

ALTER TABLE ASSOCIATES ADD CONSTRAINT L\_AS\_FK FOREIGN KEY (L\_SUBDIVISION\_NAME,LOT\_NUMBER) REFERENCES LOT(SUBDIVISION NAME,LOT NUMBER) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE ASSOCIATES ADD CONSTRAINT F\_AS\_FK FOREIGN KEY (F\_SUBDIVISION\_NAME,FLOORPLAN\_NUMBER) REFERENCES FLOORPLAN(SUBDIVISION\_NAME,FP\_NUMBER) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE ASSOCIATES ADD CONSTRAINT CON\_AS\_FK FOREIGN KEY (CONTRACT\_ID) REFERENCES CONTRACT(ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE ASSOCIATES ADD CONSTRAINT AS\_PK PRIMARY KEY ( L\_SUBDIVISION\_NAME, LOT\_NUMBER, F\_SUBDIVISION\_NAME, FLOORPLAN\_NUMBER);

CREATE TABLE CUSTOMER (EMAIL VARCHAR(255) ,NAME VARCHAR(255) , INCOME CREDIT SCORE CHECK(CREDIT SCORE>=300 DECIMAL(10,2), INT AND CREDIT SCORE<=850), BATH COUNT INT CHECK(BATH COUNT>=1 AND BATH\_COUNT<=5),TYPE VARCHAR(20) CHECK(TYPE IN ('POTENTIAL', 'EXISTING')), BED\_COUNT INT CHECK(BED\_COUNT>=1 **AND** BED COUNT<=5),SQ FT NUMBER(4,0),RANK INT ,LOANAPPLICANT ID INT);

ALTER TABLE CUSTOMER ADD CONSTRAINT CUS\_LA\_FK FOREIGN KEY (LOANAPPLICANT\_ID) REFERENCES LOANAPPLICANT(ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE CUSTOMER ADD CONSTRAINT CUS PK PRIMARY KEY (EMAIL);

CREATE TABLE TOURS (EMP\_ID INT,CUSTOMER\_EMAIL VARCHAR(255), I\_SUBDIVISION\_NAME VARCHAR(255), INVENTORY\_ID INT, TOUR\_DATE DATE);

ALTER TABLE TOURS ADD CONSTRAINT TOURS\_EMP\_FK FOREIGN KEY (EMP\_ID) REFERENCES EMPLOYEE(ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE TOURS ADD CONSTRAINT TOURS\_CUS\_FK FOREIGN KEY (CUSTOMER\_EMAIL) REFERENCES CUSTOMER(EMAIL) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE TOURS ADD CONSTRAINT TOUR\_IN\_FK FOREIGN KEY (I\_SUBDIVISION\_NAME,INVENTORY\_ID) REFERENCES INVENTORY(SUBDIVISION\_NAME,ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE TOURS ADD CONSTRAINT TOURS\_PK PRIMARY KEY (EMP\_ID,CUSTOMER\_EMAIL,I\_SUBDIVISION\_NAME,INVENTORY\_ID);

CREATE TABLE EMPLOYEEPHONE (EMPLOYEE ID INT, PHONE VARCHAR(13));

ALTER TABLE EMPLOYEEPHONE ADD CONSTRAINT PHONE\_EMP\_FK FOREIGN KEY (EMPLOYEE\_ID) REFERENCES EMPLOYEE(ID) ON DELETE CASCADE INITIALLY DEFERRED:

ALTER TABLE EMPLOYEEPHONE ADD CONSTRAINT EM\_PH\_PK PRIMARY KEY

(EMPLOYEE\_ID,PHONE);

Alter table EMPLOYEEPHONE add constraint "PHONENO\_CHECK" CHECK (REGEXP\_LIKE(PHONE, \( [0-9]{3}\)[0-9]{3}\\-[0-9]{4}'));

CREATE TABLE SECRETARY (EMPLOYEE\_ID INT, HOURLY\_WAGE DECIMAL(5,2));

ALTER TABLE SECRETARY ADD CONSTRAINT SEC\_EMP\_FK FOREIGN KEY (EMPLOYEE\_ID) REFERENCES EMPLOYEE(ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE SECRETARY ADD CONSTRAINT SEC\_PK PRIMARY KEY (EMPLOYEE\_ID);

CREATE TABLE SALES\_AGENT (EMPLOYEE\_ID INT, SALES\_SHIFT VARCHAR(3));

ALTER TABLE SALES\_AGENT ADD CONSTRAINT NE CHECK(SALES\_SHIFT IN ('monday', 'tuesday', 'wednesday', 'thrusday', 'saturday', 'saturday');

ALTER TABLE SALES\_AGENT MODIFY (SALES\_SHIFT VARCHAR(10));

ALTER TABLE SALES\_AGENT ADD CONSTRAINT SALES\_EMP\_FK FOREIGN KEY (EMPLOYEE\_ID) REFERENCES EMPLOYEE(ID) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE SALES\_AGENT ADD CONSTRAINT SALES\_PK PRIMARY KEY (EMPLOYEE\_ID, SALES\_SHIFT);

CREATE TABLE VISITS (CUSTOMER\_EMAIL VARCHAR(255), SUBDIVISION\_NAME VARCHAR(255), VISIT\_DATE DATE);

ALTER TABLE VISITS ADD CONSTRAINT VISITS\_CUS\_FK FOREIGN KEY (CUSTOMER\_EMAIL) REFERENCES CUSTOMER(EMAIL) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE VISITS ADD CONSTRAINT VISIT\_SUB\_FK FOREIGN KEY (SUBDIVISION\_NAME) REFERENCES SUBDIVISION(NAME) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE VISITS ADD CONSTRAINT VIS\_PK PRIMARY KEY (CUSTOMER\_EMAIL,SUBDIVISION\_NAME);

CREATE TABLE EXISTING( EMAIL VARCHAR(255), CONSTRAINT EX\_FK FOREIGN KEY (EMAIL) REFERENCES CUSTOMER(EMAIL) ON DELETE CASCADE INITIALLY DEFERRED, CONSTRAINT EX\_PK PRIMARY KEY(EMAIL));

CREATE TABLE POTENTIAL( EMAIL VARCHAR(255), CONSTRAINT PO\_FK FOREIGN KEY (EMAIL) REFERENCES CUSTOMER(EMAIL) ON DELETE CASCADE INITIALLY DEFERRED, CONSTRAINT PO\_PK PRIMARY KEY(EMAIL));

CREATE TABLE SIGNS (EXISTING\_CUSTOMER\_EMAIL VARCHAR(255), CONTRACT\_ID INT , SUBDIVISION\_NAME VARCHAR(255));

ALTER TABLE SIGNS ADD CONSTRAINT SIGN\_EX\_FK FOREIGN KEY (EXISTING\_CUSTOMER\_EMAIL) REFERENCES EXISTING(EMAIL) ON DELETE CASCADE

#### **INITIALLY DEFERRED;**

ALTER TABLE SIGNS ADD CONSTRAINT SIGN\_CON\_FK FOREIGN KEY (CONTRACT\_ID) REFERENCES CONTRACT(ID) ON DELETE CASCADE INITIALLY DEFERRED:

ALTER TABLE SIGNS ADD CONSTRAINT SIGN\_SUB\_FK FOREIGN KEY (SUBDIVISION\_NAME) REFERENCES SUBDIVISION(NAME) ON DELETE CASCADE INITIALLY DEFERRED;

ALTER TABLE SIGNS ADD CONSTRAINT SIGN\_PK PRIMARY KEY(EXISTING\_CUSTOMER\_EMAIL, CONTRACT\_ID, SUBDIVISION\_NAME);

CREATE TABLE CUSTOMERPHONE (EMAIL VARCHAR(255), PHONE VARCHAR(13), CONSTRAINT PHONE\_CUS\_FK FOREIGN KEY (EMAIL) REFERENCES CUSTOMER(EMAIL) ON DELETE CASCADE INITIALLY DEFERRED, CONSTRAINT CUS\_PH\_PK PRIMARY KEY (EMAIL, PHONE));

Alter table CUSTOMERPHONE add constraint "PHONE\_CHECK" CHECK (REGEXP\_LIKE(PHONE, '\([0-9]{3}\)[0-9]{3}\\-[0-9]{4}'));

CREATE TABLE LOAN( ID INT , TERM INT , TYPE VARCHAR(255), GRANT\_DATE DATE, STATUS VARCHAR(20) CHECK(STATUS IN ('APPROVED', 'DISAPPROVED') ),LOANAPPLICANT\_ID INT);

ALTER TABLE LOAN ADD CONSTRAINT LOAN\_LA\_FK FOREIGN KEY (LOANAPPLICANT\_ID) REFERENCES LOANAPPLICANT(ID) ON DELETE CASCADE INITIALLY DEFERRED:

ALTER TABLE LOAN ADD CONSTRAINT LOAN\_PK PRIMARY KEY(ID);

ALTER TABLE MANAGER ADD CONSTRAINT FK\_EMPLOYEE FOREIGN KEY (MANAGER\_ID) REFERENCES EMPLOYEE (ID) ON DELETE CASCADE INITIALLY DEFERRED;

#### ALTER TABLE CUSTOMER ADD DOB DATE:

alter table CUSTOMER add constraint "EMAILFORMAT\_CHK" check ( REGEXP\_LIKE(EMAIL, '[a-zA-Z0-9.\_%-]+@[a-zA-Z0-9.\_%-]+\.[a-zA-Z]{2,4}'));

alter table EXISTING add constraint "EXISTINGEMAIL\_CHK" check ( REGEXP\_LIKE(EMAIL, '[a-zA-Z0-9.\_%-]+\.[a-zA-Z]{2,4}'));

alter table CUSTOMERPHONE add constraint "CUSTOMERPHONE\_EMAIL\_CHK" check (REGEXP\_LIKE(EMAIL, '[a-zA-Z0-9.\_%-]+@[a-zA-Z0-9.\_%-]+\.[a-zA-Z]{2,4}'));

alter table POTENTIAL add constraint "POTENTIAL\_EMAIL\_CHK" check (REGEXP\_LIKE(EMAIL, '[a-zA-Z0-9.\_%-]+@[a-zA-Z0-9.\_%-]+\.[a-zA-Z]{2,4}'));

alter table VISITS add constraint "VISITS\_EMAIL\_CHK" check ( REGEXP\_LIKE(CUSTOMER\_EMAIL, '[a-zA-Z0-9.\_%-]+@[a-zA-Z0-9.\_%-]+\.[a-zA-Z]{2,4}'));

alter table SIGNS add constraint "SIGNS\_EMAIL\_CHK" check ( REGEXP\_LIKE(EXISTING\_CUSTOMER\_EMAIL, '[a-zA-Z0-9.\_%-]+@[a-zA-Z0-9.\_%-]+\.[a-zA-Z]{2,4}'));

#### 3.1.3 A Database State

We insert some values into the database in order to test our SQL create view and query statement. Here we just give one example of insertions as follows:

#### INSERTION OF TABLE EMPLOYEE

INSERT INTO EMPLOYEE("ID", "FNAME", "LNAME", "YEARS\_OF\_EXPERIENCE", "E\_TYPE", "SALARY") VALUES (1, 'Indhu', 'Surya', 5, 'SALES\_AGENT', 25000);

#### **EMPLOYEE**

	∯ID		<b>\$ LNAME</b>	\$\text{\$\text{YEARS_OF_EXPERIENCE}}\$		SALARY
1	1	Indhu	Surya	5	SALES_AGENT	25000 5
2	2	Sindhuja	R	15	SALES_AGENT	25345
3	3	Suneesha	Kudipudi	25	SALES_AGENT	35000 5
4	4	John	Smith	35	SALES_AGENT	25000 5
5	5	Mark	Twain	45	SALES_AGENT	25000 5
6	6	Sneha	Vishwa	50	SALES_AGENT	25000 5
7	7	Anisha	Chikerur	30	SALES_AGENT	25000 5
8	8	Sumithra	Shenoy	20	SALES_AGENT	25000 5
9	9	Swetha	Krishnakumar	5	SECRETARY	55500 5
10	10	Sathvi	Anish	5	SECRETARY	33000 5

#### **INSERTION OF TABLE MANAGER**

 $INSERT\ INTO\ MANAGER(MANAGER\_ID,\ START\_DATE)\ VALUES(1,\ TO\_DATE('06051992','MMDDYYYY'));$ 

#### **MANAGER**

		\$ START_DATE
1	1	05-JUN-92
2	2	05-JUL-92
3	3	14-AUG-92
4	4	05-DEC-92
5	5	20-SEP-92
6	6	10-JAN-00
7	7	15-FEB-02
8	8	06-JUN-15

#### **INSERTION OF TABLE SUBDIVISION**

INSERT INTO SUBDIVISION(NAME, START\_YEAR, RANK, ZIP, MANAGER\_ID) VALUES('SD1', 1993, 5, 75252, 2);

	<b>♦ NAME</b>		∯ RANK	<b>∜ ZIP</b>	
1	SD1	1993	5	75252	2
2	SD2	1993	5	75252	1
3	SD3	2015	5	75252	(null)
4	SD4	1993	7	75252	3
5	SD5	1993	8	75252	4
6	SD6	2015	2	75252	5
7	SD7	1993	1	75252	6
8	SD8	1993	11	75252	7
9	SD9	2015	10	75252	8

#### **INSERTION OF TABLE CUSTOMER**

INSERT INTO CUSTOMER (EMAIL , NAME , INCOME, CREDIT\_SCORE, BATH\_COUNT, TYPE, BED\_COUNT, SQ\_FT, RANK, DOB) VALUES ('ABC@GMAIL.COM' , 'CUST1' , 25000, 350, 1, 'EXISTING', 2, 2500, 5, TO\_DATE('06051992','MMDDYYYY') );

#### **CUSTOMER**

		NAME			⊕ BATH_COUNT		BED_COUNT		∯ RANK	UOANAPPLICANT_ID	∯ DOB
1	ABC@GMAIL.COM	CUST1	25000	350	1	EXISTING	2	2500	5	(null)	05-JUN-92
2	EFG@GMAIL.COM	CUST3	45000	500	1	EXISTING	3	4500	2	10003	05-NOV-92
3	IJK@YAHOO.COM	CUST 4	25000	300	1	EXISTING	1	2500	3	10004	14-NOV-92
4	LMN@GMAIL.COM	CUST5	235000	400	1	EXISTING	2	3500	2	(null)	13-NOV-00
5	OPQ@GMAIL.COM	CUST 6	24000	450	2	EXISTING	2	4500	1	(null)	10-JAN-00
6	RST@GMAIL.COM	CUST7	252000	650	2	EXISTING	3	5500	1	(null)	28-NOV-83
7	UVW@GMAIL.COM	CUST8	28000	350	1	EXISTING	1	2500	3	(null)	28-NOV-83
8	CDE@GMAIL.COM	CUST2	35000	550	2	EXISTING	2	3500	4	(null)	20-SEP-92
9	XYZ@GMAIL.COM	CUST9	235000	750	1	POTENTIAL	2	3500	2	(null)	13-NOV-00
10	BCD@GMAIL.COM	CUST10	24000	750	2	POTENTIAL	2	4500	1	(null)	10-JAN-00
11	ZXY@GMAIL.COM	CUST11	252000	650	2	POTENTIAL	3	5500	1	(null)	28-NOV-83

#### **INSERTION OF TABLE LOT**

 $INSERT\ INTO\ LOT(SUBDIVISION\_NAME,\ LOT\_NUMBER,\ LOT\_SIZE,\ FACING\ ,\ AVAILABILITY\ )\\ VALUES('SD1', 1001, 500, 'NORTH', 'YES');$ 

#### **LOT**

	\$\text{\$\text{SUBDIVISION_NAME}}	\$LOT_NUMBER	\$LOT_SIZE		
1	SD1	1001	500	NORTH	YES
2	SD2	1002	100	SOUTH	NO
3	SD3	1003	200	EAST	YES
4	SD4	1004	300	WEST	YES
5	SD1	1005	500	SOUTH	NO
6	SD2	1006	200	NORTH	YES
7	SD5	1007	100	NORTH	NO

#### **INSERTION OF TABLE FLOORPLAN**

INSERT INTO FLOORPLAN(SUBDIVISION\_NAME,FP\_NUMBER , FLOOR\_COUNT , BED\_COUNT , BATH\_COUNT , NAME , PRICE , SQ\_FT ,GARAGE\_COUNT) VALUES ('SD1', 101 , 1 , 3 , 2 , 'FP1' , 500 , 3625 ,1);

#### **FLOORPLAN**

	\$\text{\$\text{SUBDIVISION_NAME}}	\$ FP_NUMBER		BED_COUNT	BATH_COUNT	NAME		\$Q_FT	GARAGE_COUNT
1	SD1	101	1	3	2	FP1	500	3625	2
2	SD2	102	2	4	2	FP1	600	4625	2
3	SD3	103	3	2	2	FP2	700	5625	2
4	SD4	104	3	5	3	FP2	800	6625	3
5	SD5	105	2	4	2	FP1	600	4625	2
6	SD2	106	2	2	1	FP3	600	3625	1
7	SD1	107	1	1	1	FP4	500	2625	1

#### **INSERTION OF TABLE INVENTORY**

 $INSERT\ INTO\ INVENTORY(\ SUBDIVISION\_NAME\ ,\ ID\ ,PRICE,\ MOVE\_IN\_DATE\ )\ VALUES('SD1',\ 10000,\ 56,\ TO\_DATE('05091992',\ 'MMDDYYYY')\ );$ 

#### **INVENTORY**

	\$ SUBDIVISION_NAME	∯ ID		MOVE_IN_DATE
1	SD1	10000	56	09-MAY-92
2	SD2	10001	5634	04-JUN-00
3	SD3	10002	5623	03-JUL-01
4	SD1	10003	5456	06-FEB-15
5	SD5	10004	5456	07-APR-15
6	SD3	10005	3454	(null)
7	SD2	10006	3456	08-NOV-15
8	SD4	10007	5236	09-DEC-14

#### INSERTION OF TABLE CONTRACT

INSERT INTO CONTRACT(ID, SIGN\_DATE , SALE\_PRICE, EMPLOYEE\_ID) VALUES(1,TO\_DATE('11282014','MMDDYYYY'),5000,1);

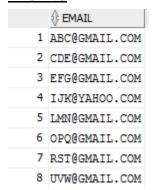
#### **CONTRACT**

	∯ ID	\$ SIGN_DATE	\$ SALE_PRICE	\$ EMPLOYEE_ID
1	1	28-NOV-14	5000	1
2	2	24-NOV-14	4000	2
3	3	02-SEP-00	3000	3
4	4	04-APR-92	5000	1
5	5	04-FEB-12	3000	2
6	6	03-NOV-13	4000	1
7	7	15-NOV-14	2000	5
8	8	18-NOV-15	4000	5

#### INSERTION OF TABLE EXISTING

INSERT INTO EXISTING(EMAIL) VALUES ('ABC@GMAIL.COM');

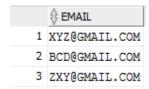
#### **EXISTING**



#### **INSERTION OF TABLE POTENTIAL**

INSERT INTO POTENTIAL(EMAIL) VALUES ('XYZ@GMAIL.COM');

#### **POTENTIAL**



#### INSERTION OF TABLE LOANAPPLICANT

INSERT INTO LOANAPPLICANT(ID) VALUES(10001);

#### **LOANAPPLICANT**



#### **INSERTION OF TABLE LOAN**

INSERT INTO LOAN(ID, TERM, TYPE, GRANT\_DATE, STATUS, LOANAPPLICANT\_ID) VALUES(101,7,'FIXED RATE', TO\_DATE('11282014','MMDDYYYY'), 'APPROVED', 10001);

#### **LOAN**

	∯ID	∯ TERM	<b>∜ TYPE</b>				
1	101	7	FIXED	RATE	28-NOV-14	APPROVED	10001
2	102	15	ARM		02-SEP-00	DISAPPROVED	10002
3	103	7	FIXED	RATE	28-NOV-15	APPROVED	10003
4	104	4	ARM		28-SEP-15	APPROVED	10004
5	105	5	FIXED	RATE	28-OCT-00	APPROVED	10002

### INSERTION OF TABLE SECRETARY

INSERT INTO SECRETARY(EMPLOYEE\_ID, HOURLY\_WAGE) VALUES(9,30);

#### **SECRETARY**

		♦ HOURLY_WAGE
1	9	30
2	10	35

#### **INSERTION OF TABLE SALES\_AGENT**

 $INSERT\ INTO\ SALES\_AGENT(EMPLOYEE\_ID,\ SALES\_SHIFT)\ VALUES(1, 'monday');$ 

#### SALES AGENT

		\$ SALES_SHIFT
1	1	monday
2	2	tuesday
3	4	monday
4	5	monday
5	6	wednesday
6	7	monday
7	1	friday
8	2	saturday
9	3	sunday
10	4	wednesday
11	1	thrusday
12	3	wednesday

#### **INSERTION OF TABLE CUSTOMERPHONE**

 $INSERT\ INTO\ CUSTOMERPHONE (EMAIL, PHONE)\ VALUES ('ABC@GMAIL.COM', '(999)999-9999');$ 

#### **CUSTOMERPHONE**



#### INSERTION OF TABLE EMPLOYEE\_PHONE

INSERT INTO EMPLOYEEPHONE(EMPLOYEE\_ID,PHONE) VALUES(1,'(999)999-9999');

#### **EMPLOYEEPHONE**

	\$ EMPLOYEE_ID	
1	1	(999) 999-9999
2	2	(999) 999-9999
3	3	(999) 999-9999
4	4	(999) 999-9999
5	7	(999) 999-9999
6	10	(999) 999-9999
7	1	(456) 999-9999
8	1	(753) 999-9999
9	2	(432)999-9999
10	3	(865) 999-9999
11	6	(437) 999-9999

#### **INSERTION OF TABLE VISITS**

 $INSERT INTO VISITS (CUSTOMER\_EMAIL, SUBDIVISION\_NAME, VISIT\_DATE) \\ VALUES ('ABC@GMAIL.COM', 'SD1', TO\_DATE ('05092000', 'MMDDYYYY')); \\$ 

#### **VISITS**

	Λ	Λ	Λ
1	ABC@GMAIL.COM	SD1	09-MAY-00
2	ABC@GMAIL.COM	SD2	09-JUN-01
3	ABC@GMAIL.COM	SD3	09-NOV-14
4	ABC@GMAIL.COM	SD4	09-DEC-15
5	CDE@GMAIL.COM	SD1	09-SEP-15
6	OPQ@GMAIL.COM	SD1	09-AUG-11
7	CDE@GMAIL.COM	SD3	09-JUL-00
8	UVW@GMAIL.COM	SD4	09-JUN-15
9	UVW@GMAIL.COM	SD5	09-APR-14
10	EFG@GMAIL.COM	SD2	09-MAY-93
11	EFG@GMAIL.COM	SD6	09-MAR-92
12	IJK@YAHOO.COM	SD2	09-MAY-92

#### **INSERTION OF TABLE SIGNS**

 $INSERT INTO SIGNS (EXISTING\_CUSTOMER\_EMAIL,CONTRACT\_ID, SUBDIVISION\_NAME) \\ VALUES ('ABC@GMAIL.COM', 1, 'SD1');$ 

#### **SIGNS**

	\$\text{EXISTING_CUSTOMER_EMAIL}		\$ SUBDIVISION_NAME
1	ABC@GMAIL.COM	1	SD1
2	ABC@GMAIL.COM	2	SD4
3	ABC@GMAIL.COM	3	SD3
4	CDE@GMAIL.COM	4	SD2
5	EFG@GMAIL.COM	5	SD5
6	EFG@GMAIL.COM	6	SD4
7	OPQ@GMAIL.COM	7	SD3
8	OPQ@GMAIL.COM	8	SD2

#### **INSERTION OF TABLE ASSOCIATES**

 $INSERT \quad INTO \quad ASSOCIATES (L\_SUBDIVISION\_NAME, \quad LOT\_NUMBER, \quad F\_SUBDIVISION\_NAME, \\ FLOORPLAN\_NUMBER, CONTRACT\_ID) \ VALUES ('SD1', 1001, 'SD1', 101, 1); \\$ 

#### **ASSOCIATES**

	\$L_SUBDIVISION_NAME	\$LOT_NUMBER			CONTRACT_ID
1	SD1	1001	SD1	101	1
2	SD1	1001	SD2	102	2
3	SD1	1005	SD1	101	3
4	SD1	1005	SD2	102	4
5	SD2	1002	SD1	101	5
6	SD2	1006	SD2	102	6
7	SD4	1004	SD2	106	8
8	SD5	1007	SD2	106	1
9	SD3	1003	SD2	106	7

#### **INSERTION OF TABLE TOURS**

INSERT INTO TOURS(EMP\_ID, CUSTOMER\_EMAIL, I\_SUBDIVISION\_NAME, INVENTORY\_ID, TOUR\_DATE) VALUES(1, 'ABC@GMAIL.COM', 'SD1', 10000, TO\_DATE('05092000', 'MMDDYYYY'));

#### **TOURS**

				\$ INVENTORY_ID	TOUR_DATE
1	1	ABC@GMAIL.COM	SD1	10000	09-MAY-00
2	3	ABC@GMAIL.COM	SD1	10000	29-DEC-04
3	2	ABC@GMAIL.COM	SD1	10000	19-NOV-03
4	1	IJK@YAHOO.COM	SD4	10007	09-AUG-05
5	3	IJK@YAHOO.COM	SD4	10007	05-JUL-07
6	2	LMN@GMAIL.COM	SD5	10004	03-MAY-08
7	4	IJK@YAHOO.COM	SD5	10004	02-MAR-06
8	3	RST@GMAIL.COM	SD3	10005	01-MAR-15
9	2	RST@GMAIL.COM	SD2	10001	18-APR-14

#### **INSERTION OF TABLE COME**

INSERT INTO COME(I\_SUBDIVISION\_NAME, INVENTORY\_ID, L\_SUBDIVISION\_NAME, LOT\_NUMBER, F\_SUBDIVISION\_NAME, FLOORPLAN\_NUMBER) VALUES('SD1', 10000, 'SD1', 1001, 'SD3', 103);

#### **COME**

		\$ INVENTORY_ID				
1	SD1	10000	SD1	1001	SD3	103
2	SD1	10000	SD1	1005	SD4	104
3	SD2	10001	SD2	1002	SD3	103
4	SD2	10001	SD2	1006	SD1	107
5	SD3	10002	SD3	1003	SD2	102
6	SD3	10002	SD2	1002	SD1	101
7	SD3	10005	SD3	1003	SD3	103
8	SD3	10005	SD2	1002	SD1	107
9	SD5	10004	SD1	1005	SD3	103
10	SD5	10004	SD3	1003	SD4	104

#### UPDATE OF TABLE EMPLOYEE

UPDATE EMPLOYEE SET SUBDIVISION\_NAME='SD1', LOANAPPLICANT\_ID='10001' WHERE ID=1; UPDATE EMPLOYEE SET SUBDIVISION\_NAME='SD1' WHERE ID=2;

#### **EMPLOYEE**

	∯ID	<b>♦ FNAME</b>			E_TYPE			
1	1	Indhu	Surya	5	SALES_AGENT	25000	SD1	10001
2	2	Sindhuja	R	15	SALES_AGENT	25345	SD1	(null)
3	3	Suneesha	Kudipudi	25	SALES_AGENT	35000	SD2	10002
4	4	John	Smith	35	SALES_AGENT	25000	SD1	(null)
5	5	Mark	Twain	45	SALES_AGENT	25000	SD7	10005
6	6	Sneha	Vishwa	50	SALES_AGENT	25000	SD2	(null)
7	7	Anisha	Chikerur	30	SALES_AGENT	25000	SD3	(null)
8	8	Sumithra	Shenoy	20	SALES_AGENT	25000	SD3	(null)
9	9	Swetha	Krishnakumar	5	SECRETARY	55500	SD5	(null)
10	10	Sathvi	Anish	5	SECRETARY	33000	SD6	(null)

Till now we finished the process of creating tables and database states.

# 3.2 Creation of Views (Answer for Question d)

Use the Create View statement to create the following views:

1. School rank: show the school rank of each subdivision along with the average base price of floor plans the subdivision provides.

 $\label{lem:condition} Create \ view \ school\_rank \ as \ select \ s.name \ as \ subdivisionname, \ rank, \ avg(f.price) \ as \ avg\_base\_price \ from \ subdivision \ s \ join \ floorplan \ f \ on \ s.name = f.subdivision\_name \ group \ by \ s.name, \ rank;$ 

**SELECT \* FROM SCHOOL\_RANK;** 

		<b>♦ RANK</b>	\$ AVG_BASE_PRICE
1	SD3	5	700
2	SD4	7	800
3	SD2	5	600
4	SD1	5	500
5	SD5	8	600

2. Promising customers: show the name, age and email address of potential customers with credit score over 740.

Create view promising\_customers as select name, floor(months\_between(SYSDATE, DOB) /12) as age, p.email from potential p join customer c on p.email = c.email where credit\_score > 740;

#### **SELECT \* FROM PROMISING\_CUSTOMERS;**

	NAME		
1	CUST9	15	XYZ@GMAIL.COM
2	CUST10	15	BCD@GMAIL.COM

3. Inventory homes: show the information of unsold inventory homes that have been completed or will be completed by the end of 2015 for each subdivision along with the manager name of the subdivision.

Create view inventory\_homes as select i.subdivision\_name, concat(e.fname, e.lname) as manager, i.id as inventory\_id, price, move\_in\_date from inventory I join subdivision s on i.subdivision\_name = s.name join employee e on s.manager\_id = e.id where move\_in\_date is null group by i.subdivision\_name, concat(e.fname, e.lname), i.id, price, move\_in\_date;

#### **SELECT \* FROM INVENTORY\_HOMES;**

	\$ SUBDIVISION_NAME	MANAGER	\$ INVENTORY_ID	♦ PRICE	
1	SD3	SwethaKrishnakumar	10005	3454	(null)

4. Large floor plans: show the information of floor plans over 4000 square feet for each subdivision.

Create view Large\_floor\_plans as Select subdivision\_name, fp\_number, floor\_count, bed\_count, bath\_count, name, price, sq\_ft, garage\_count from floorplan where sq\_ft > 4000 group by subdivision\_name, fp\_number, floor\_count, bed\_count, bath\_count, name, price, sq\_ft, garage\_count;

SELECT \* FROM LARGE\_FLOOR\_PLANS;

	\$ SUBDIVISION_NAME	\$ FP_NUMBER			⊕ BATH_COUNT	<b>∜ NAME</b>		SQ_FT	
1	SD4	104	3	5	3	FP2	800	6625	3
2	SD5	105	2	4	2	FP1	600	4625	2
3	SD3	103	3	2	2	FP2	700	5625	2
4	SD2	102	2	4	2	FP1	600	4625	2

5. Sales record: for each subdivision, show the name of each sales agent and the number of contracts she prepared in year 2014.

Create view Sales\_Record as Select subdivision\_name, e.id as employee\_id, concat(fname, lname) as name, count(c.id) as number\_of\_contracts from employee e join contract c on e.id = c.employee\_id where to\_char(sign\_date,'yyyy') = '2014' group by subdivision\_name, e.id, concat(fname, lname);

SELECT \* FROM SALES\_RECORD;

	\$ SUBDIVISION_NAME	\$ EMPLOYEE_ID	NAME	NUMBER_OF_CONTRACTS	
1	SD1	1	IndhuSurya	1	
2	SD1	2	SindhujaR	1	
3	SD7	5	MarkTwain	1	

6. Deal discount: for each subdivision, show the information of the contracts signed for inventory homes with a discount in year 2015. Here contract with discount means that the sales price showed in the contract is lower than the list price of the inventory home.

Create view Deal\_discount as select sign.subdivision\_name, c.id as contracted, sign\_date, sale\_price, employee\_id from contract c join signs sign on c.id = sign.contract\_id join associates a on sign.contract\_id = a.contract\_id join come c1 on (a.l\_subdivision\_name = c1.l\_subdivision\_name and a.lot\_number = c1.lot\_number and a.f\_subdivision\_name = c1.f\_subdivision\_name and a.floorplan\_number = c1.floorplan\_number) join inventory i on (c1.i\_subdivision\_name = i.subdivision\_name and c1.inventory\_id = i.id) where c.sale\_price < i.price and sign\_date >= to\_date('01/JAN/2015', 'dd/mon/yyyy') group by sign.subdivision\_name, c.id, sign\_date, sale\_price, employee\_id;

#### SELECT \* FROM DEAL\_DISCOUNT;

	\$\text{\$\text{SUBDIVISION_NAME}\$}			\$ SALE_PRICE	
1	SD2	8	18-NOV-15	4000	5

# 3.3 Creation of SQL Queries (Answer for Question f)

Now we give out the SQL Queries for each of 10 questions listed in Question e as follows:

1. Retrieve the school rank of each subdivision in decreasing order of the average base price of floor plans the subdivision provides.

Select \* from school\_rank order by avg\_base\_price desc;

		∯ RANK	AVG_BASE_PRICE	
1	SD4	7	800	
2	SD3	5	700	
3	SD5	8	600	
4	SD2	5	600	
5	SD1	5	500	

2. For each subdivision, retrieve the number of unsold inventory homes that have been completed or will be ready by the end of year 2015.

Select subdivision\_name, count(inventory\_id) as unsold\_inventory from inventory\_homes group by subdivision\_name;

	\$ SUBDIVISION_NAME	UNSOLD_INVENTORY	
1	SD3	1	

3. For each subdivision, retrieve information of the sales agents who prepared all the contracts with floor plans over 4000 square feet signed in year 2015

Select lfp.subdivision\_name, c.employee\_id, c.id from Large\_floor\_plans lfp join associates a on (lfp.subdivision\_name = a.f\_subdivision\_name and lfp.fp\_number = a.floorplan\_number) join contract c on a.contract\_id = c.id where c.employee\_id in (select employee\_id from SALES\_AGENT) group by lfp.subdivision\_name, c.employee\_id, c.id;

	\$\text{\$\text{SUBDIVISION_NAME}\$}	\$ EMPLOYEE_ID	∯ ID
1	SD2	1	6
2	SD2	1	4
3	SD2	2	2

4. For each subdivision, retrieve the information of the sales agent who prepared the highest number of contracts in year 2014.

 $Select * from employee e join sales\_record sr on \underline{e.id} = sr.employee\_id where sr.number\_of\_contracts = (select max(number\_of\_contracts) from sales\_record);$ 

ID () FNAME			E_TYPE	SALARY	\$ SUBDIVISION_N		\$ SUBDIVISION		NAME	NUMBER_OF_CONTRACTS
1 Indhu	Surya	5	SALES_AGENT	25000	SD1	10001	SD1	:	l IndhuSurya	1
2 Sindhuja	R	15	SALES_AGENT	25345	SD1	(null)	SD1		2 SindhujaR	1
5 Mark	Twain	45	SALES_AGENT	25000	SD7	10005	SD7		MarkTwain	1

5. Retrieve the information of the inventory home that has not been sold for the longest time since its completion, along with the name of the manager who is in charge of that subdivision.

Select \* from (select \* from inventory homes order by move in date) where rownum = 1;

	\$\text{\$\subdivision_name}\$	MANAGER		<b>\$</b> ▼	MOVE_IN_DATE
1	SD3	SwethaKrishnakumar	10005	3454	(null)

6. Retrieve the information of potential customers who have been visiting subdivisions of the company for more than one month and the price range of the inventory homes they have toured lies between \$300,000 to \$400,000.

Select c.email, c.name, c.income, c.credit\_score, i.price from CUSTOMER c join potential p on c.email = p.email join tours t on p.email = t.customer\_email join inventory i on (t.i\_subdivision\_name = i.subdivision\_name and t.inventory\_id = i.id) where i.price between 300000 and 400000;

		NAME			PRICE
1	XYZ@GMAIL.COM	CUST9	235000	750	350000
2	BCD@GMAIL.COM	CUST10	24000	750	350000

7. For each subdivision, retrieve the information of each customer whose home loan has not been approved over one month after he/she signed his/her contract.

Select sign.subdivision\_name, sign.existing\_customer\_email, cust.name, c.id as contract\_id, c.sign\_date, sale\_price, status from contract c join signs sign on c.id = sign.contract\_id join customer cust on sign.existing\_customer\_email = cust.email join loan on cust.loanapplicant\_id = loan.loanapplicant\_id where (status is null or status = 'DISAPPROVED') and months\_between(sysdate, sign\_date) > 1 group by sign.subdivision\_name, sign.existing\_customer\_email, cust.name, c.id, c.sign\_date, sale\_price, status;

4	♦ SUBDIVISION	⊕ EXISTING_CUSTOMER_EMAIL	⊕ NAME	⊕ CONTRACT_ID		\$ SALE_PRICE		
1 5	SD5	EFG@GMAIL.COM	CUST3	5	04-FEB-12	3000	(null)	
2 5	5D4	EFG@GMAIL.COM	CUST3	6	03-NOV-13	4000	(null)	

8. Retrieve the name, age and email address of potential customers with credit score over 740 who visited a subdivision in August this year but have not visited any subdivision since September 1st.

 $Select * from promising\_customers pc, visits v where pc.email = v.customer\_email and upper(to\_char(visit\_date, 'Month')) = 'AUGUST' and not exists (select 1 from visits vs where vs.customer\_email = pc.email and visit\_date >= to\_date('01-SEP-2015', 'dd-mon-yy'));$ 

4	NAME	∯ AGE					
---	------	-------	--	--	--	--	--

Retrieve the average discount received by the customers who have signed a contract to purchase an inventory home.

Select avg(price-sale\_price) from contract c join signs sign on c.id = sign.contract\_id join associates a on sign.contract\_id = a.contract\_id join come c1 on (a.l\_subdivision\_name = c1.l\_subdivision\_name and a.lot\_number = c1.lot\_number and a.f\_subdivision\_name = c1.f\_subdivision\_name and a.floorplan\_number = c1.floorplan\_number) join inventory i on (c1.i\_subdivision\_name = i.subdivision\_name and c1.inventory\_id = i.id) where c.sale\_price < i.price;



10. Retrieve information of the potential customers whose house preferences can be matched by one or more inventory homes that will be ready to move in by the end of year 2015

select customer.email, customer.name from inventory i join come c on (c.i\_subdivision\_name = i.subdivision\_name and c.inventory\_id = i.id) join floorplan fp on (c.f\_subdivision\_name = fp.subdivision\_name and c.floorplan\_number = fp.fp\_number) , customer where customer.bath\_count=fp.bath\_count and customer.bed\_count=fp.bed\_count and customer.sq\_ft=fp.sq\_ft and upper(to\_char(i.move\_in\_date,'Year')) = '2015' and customer.email in (select email from potential);



# 4. Conclusion

In this report we modified the EER diagram and relational schemas for XXX Database according to the requirement of Phase III. We also give dependency diagram for each relational schema in database. Then we created tables for each relational schema and write the SQL statements for the views and queries listed in Question d and Question e.