2018

DATABASE DESIGN FOR APARTMENT MANAGEMENT SYSTEM

AARYA, HARSHIL, VINYAS

REQUIREMENTS

OVERVIEW

Whether you're moving into your own place for the first time, getting a place to stay in for college, or just making a change in your living situation, renting a new apartment is an exciting time. It's also a potentially nerve-wracking challenge, when you consider the cost of the apartment, moving all your stuff in, unpacking, and getting suitable roommates. The only constant in apartment life, though, is the lease. In the United States, for the most part, apartments are arranged in complexes or buildings, and are individually rented, or leased out. Apartments are also paid for on a month-by-month basis, as opposed to being purchased outright.

Well, it gets very difficult at times to keep a track of people leasing out apartments because there are many apartments within a community and people usually lease out in groups. Therefore, we need a database management system which will have the information of people leasing out the apartments and all the other data pertaining to the apartments and the leasing office like Lease start date, Lease end date, services provided by the leasing office etc.

DEFINING ENTITIES

An entity is any object in the system that we want to model and store information about. Entities are usually recognizable concepts, either concrete or abstract, such as person, places, things, or events which have relevance to the database.

The main function of the leasing office is to assign apartment/flat to a group of people depending on the availability. After assigning the apartment, the leasing period and the rent is finalized upon. The residents then have the access to all the amenities. It is on the maintenance team to look after the apartments after they are rented. The residents if at any point feel something has malfunctioned or is not working, they can raise a complaint to the maintenance department.

RESIDENT STRUCTURE

This entity defines a person or a group of people leasing the apartment. Depending on the sizes of the apartment (1B/1B, 2B/2B or 3B/3B) there is a limit on number of people renting it. Every resident can be uniquely identified by the SSN number.

BANK ACCOUNT

In the United States, usually the rent is payed through cards (debit, credit) because handing over a large amount of cash is not safe and is inconvenient. And the payment is done on a monthly basis,so the residents have an option of linking the card which would automatically deduct the amount at the end of the month or manually pay at the end of the month. Each bank account has a unique account number.

FLAT/APARTMENT

Available in different sizes and varieties which gives residents options ranging from unfurnished to totally furnished apartments. Each apartment is identified uniquely by the apartment number. Depending on the sizes of the apartment there is a limit on the number of people leasing it. For example, for a 1B/1B there is limit of 2 people and for a 2B/2B the limit is 4.

LEASING AGENT

Every resident wanting to lease an apartment is assigned an agent. The agent lays down the set of rules and explains the procedure which is to be followed for renting the apartment. One resident cannot have more than one agent although an agent can have many residents under him/her. A leasing agent can be uniquely identified by an agent id assigned by the leasing office.

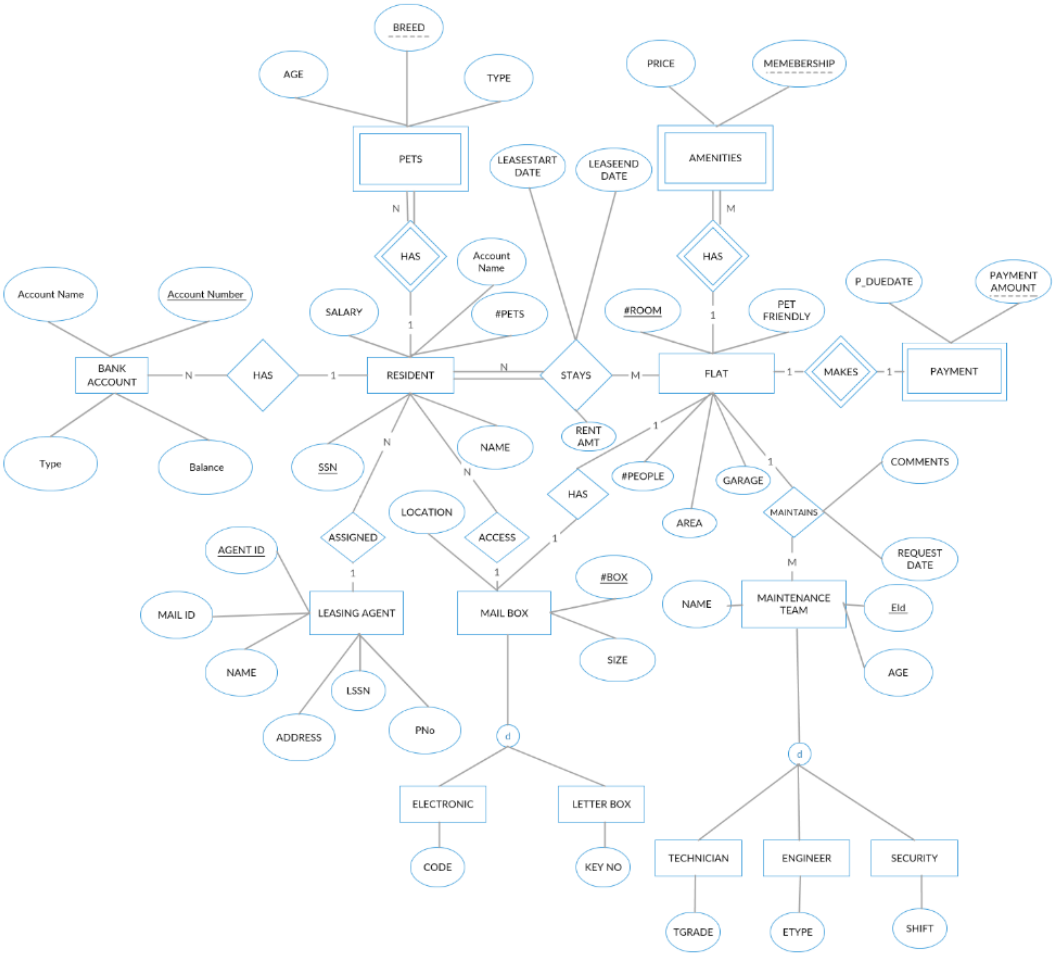
MAINTENANCE TEAM

Every successful organization has an effective maintenance team behind the scenes, tweaking and repairing the tools and facilities we all depend on. The maintenance team can be further classified into 3 categories:

- Technician
- Engineer
- Security

Depending on the type of the problem that arises one of the above mentioned team is consulted by the residents and problem is solved. Each person associated with the maintenance team has a unique id.

Modeling of Requirements as ER-Diagram:



Mapping of ERD into relational schema

1. RESIDENT

SSN	Res_Name	Lagent_ID	Mbox_No	Salary	# Pets	Profession	Email	Phone

- Primary Key: SSN
- Foreign Key: Lagent_ID REFERENCES LEASING AGENT(Agent_ID)
  Mbox_No REFERENCES MAILBOX(Box_No)

2. ACCOUNT DETAILS

AccountNumber	BSSN	AccountType

- Primary Key: AccountNumber
- Foreign Key: BSSN REFERENCES RESIDENT(SSN)

3. PETS

RSSN	PetName	Age	PetType

- Primary Key: RSSN and PetName together make up the primary key since PETS is a weak entity.
- Foreign Key: RSSN REFERENCES RESIDENT (SSN)

4. STAYS

RSSN	RoomNo	LeaseStartDate	LeaseEndDate

- Primary Key: RSSN and RoomNo together make up the primary key.
- Foreign Key: RSSN REFERENCES RESIDENT (SSN)

5.PAYMENT

RoomNo	RentAmt	DueDate	RentPaid

- Primary Key: RoomNo and RentAmt together make up the Primary key.
- Foreign Key: RoomNo REFERENCES FLAT (RoomNo)

6. LEASING AGENT

AgentID	AgentName	PhoneNumber	Email

- Primary Key: AgentID
- Foreign Key: No Foreign Key

7. FLAT

RoomNo	TotalResidents	Area

- Primary Key: RoomNo
- Foreign Key: No foreign key

8. AMENITIES

ARoomNo	Membership	Price

- Primary Key: ARoomNo and Membership together make up the primary key.
- Foreign Key: ARoomNo REFERENCES FLAT (RoomNo)

9. MAIL BOX

BoxNo	MLocation	BoxSize

- Primary Key: BoxNo

10. ELECTRONIC BOX

EboxNo	Code

- Primary Key: No Primary Key
- Foreign Key: EboxNo REFRECNECES MAILBOX (BoxNo)

11. LETTERBOX

LBoxNo	KeyNo

- Primary Key: No primary key.
- Foreign Key: LBoxNo REFERENCES MAILBOX (BoxNo)

12. MAINTENANCE TEAM

EID	MName	Age	EmailID

- Primary Key: EID

13. TECHNICIAN

TeID	TGrade

- Foreign key: TeID REFERENCES MAINTENANCE TEAM(EID)

14. ENGINEER

EeID	Specialization

- Foreign Key: EeID REFERENCES MAINTENANCE TEAM(EID)

15. GARDENER

GeID	GType

- Foreign Key: Foreign key: GeID REFERENCES MAINTENANCE TEAM(EID)

16. MAINTAINS

RId	MeID	MSSN	MRoomNo	RequestDate	Comments

- Primary Key: RID, MeID, MSSN, MRoomNo and RequestDate together make up the primary key.
- Foreign Key: MeID REFERENCES MAINTENANCETEAM (EID),
  MSSN REFERENCES RESIDENT (SSN),
  MRoomNo REFERENCES FLAT (RoomNo)

SQL STATEMENTS TO CREATE RELATIONS IN DB AND ADD CONSTARINTS

We used MySQL instance for this project and the queries correspond to MySQL standard.

```
DROP TABLE AccountDetails;
 CREATE TABLE AccountDetails(
  BSSN integer,
AccountNumber  integer,
AccountType  varchar(40),
  primary key (AccountNumber)
);

insert into AccountDetails values(432093120,2334412345,'Checking');
insert into AccountDetails values(432093121,2334412346,'Savings');
insert into AccountDetails values(432093122,2334412347,'Checking');
```

```
insert into AccountDetails values(432093123,2334412348,'Checking');

insert into AccountDetails values(432093124,2334412349,'Savings');


select * from AccountDetails;


DROP TABLE Pets;

 CREATE TABLE Pets(

  RSSN integer,

Ptype  varchar(20),

  Age integer NOT NULL,

PetName  varchar(40),

  primary key (RSSN,PetName)

);


insert into Pets values(432093120,'Dog',4,'scooby');

insert into Pets values(432093121,'Dog',6,'tuffy');

insert into Pets values(432093122,'Cat',7,'tommy');

insert into Pets values(432093123,'Dog',8,'gilbert');

insert into Pets values(432093123,'Dog',8,'simon');

insert into Pets values(432093124,'Cat',9,'gipsy');


select * from Pets;
```

```
DROP TABLE LeasingAgent;
 CREATE TABLE LeasingAgent(
AgentID integer,
AgentName varchar(40),
  Email  varchar(40) NOT NULL,
pNumber  Integer NOT NULL,
  primary key (AgentID)
);

insert into LeasingAgent values(73,'jason','jason@gmail.com',1234556346);
insert into LeasingAgent values(20,'mark','mark@gmail.com',6356435345);
insert into LeasingAgent values(93,'ross','ross@gmail.com',5345354567);
insert into LeasingAgent values(09,'jeff','jeff@gmail.com',8465653222);
insert into LeasingAgent values(32,'ted','ted@gmail.com',9243325325);

select * from LeasingAgent;

DROP TABLE Resident;
CREATE TABLE Resident(
  SSN integer,
ResidentName varchar(30),
LAgentID integer,
```

```
MboxNo integer,
  Salary integer,
TotalPets integer DEFAULT 0,
  Profession varchar(40),
  Email  varchar(40) NOT NULL,
pNumber  varchar(20) NOT NULL,
  primary key (SSN)
);

insert into Resident
values(432093120,'Zoey',73,24,50000,DEFAULT,'Professor','Zoey@gmail.com',235325666);
insert into Resident
values(432093127,'zack',73,24,60000,DEFAULT,'Professor','zack@gmail.com',235325766);
insert into Resident
values(432093121,'Chandler',20,69,60000,DEFAULT,'Banker','Chandler@gmail.com',4664344534);
insert into Resident
values(432093122,'Phoebe',93,10,40000,1,'Clerk','Phoebe@gmail.com',4776542754);
insert into Resident
values(432093123,'Rachel',93,47,80000,2,'Manager','Rachel@gmail.com',8654374265);
insert into Resident
values(432093124,'Monica',32,70,70000,1,'Analyst','Monica@gmail.com',6538542256);

insert into Resident
values(43209345,'Monidog',32,70,70000,7,'Analyst','Monica@gmail.com',6538542256);
```

```
select * from Resident;


delete from Resident where SSN='43209345';


DROP TABLE Stays;
 CREATE TABLE Stays(
  RSSN integer,
RoomNo  integer,
LeaseStartDate  Date NOT NULL,
LeaseEndDate  Date NOT NULL,
  primary key (RSSN,RoomNo)
);


insert into Stays values(432093120,1524,'20-may-15','12-june-20');
insert into Stays values(432093120,1627,'11-oct-16','12-jul-20');
insert into Stays values(432093121,2226,'11-dec-17','12-feb-19');
insert into Stays values(432093123,403,'11-aug-17','12-sep-19');
insert into Stays values(432093124,003,'11-nov-18','12-oct-20');


insert into Stays values(432093124,1524,'11-nov-20','12-oct-18');
delete from Stays where RSSN='432093124' and RoomNo=1524;
```

```
select * from Stays;


DROP TABLE Payment;
 CREATE TABLE Payment(
RoomNo integer,
RentAmount integer,
DueDate Date NOT NULL,
Rentpaid varchar(10) DEFAULT 'NO',
  primary key (RoomNo,RentAmount)
);


insert into Payment values(1524,1500,'11-dec-18',DEFAULT);
insert into Payment values(1627,1400,'11-dec-18','yes');
insert into Payment values(2226,1300,'11-dec-18',DEFAULT);
insert into Payment values(403,1800,'11-dec-18','yes');
insert into Payment values(003,1700,'11-dec-18',DEFAULT);
insert into payment values(104,1700,'19-oct-18',DEFAULT);
insert into payment values(6604,2700,'1-oct-18',DEFAULT);


select * from Payment where RentPaid='NO';


DROP TABLE Flat;
```

```
 CREATE TABLE Flat(
RoomNo integer,
TotalResidents integer DEFAULT 0,
  Area integer,
  primary key (RoomNo)
);


insert into Flat values(1524,4,1200);
insert into Flat values(1627,2,1400);
insert into Flat values(2226,3,1700);
insert into Flat values(403,5,1400);
insert into Flat values(003,6,2000);
insert into Flat values(004,DEFAULT,1000);
insert into Flat values(104,DEFAULT,1200);
insert into Flat values(6604,DEFAULT,1500);
insert into Flat values(1004,DEFAULT,1400);


select * from Flat;


DROP TABLE Amenities;
 CREATE TABLE Amenities(
ARoomNo integer,
  Membership varchar(40) default 'silver',
```

```
  Price integer DEFAULT 50,
  primary key (ARoomNo,Membership)
);

insert into Amenities values(1524,'gold',100);
insert into Amenities values(1524,'platinum',150);
insert into Amenities values(403,default,40);
insert into Amenities values(1627,'silver',70);
insert into Amenities values(3,'platinum',40);
insert into Amenities values(4,'gold',70);

select * from Amenities;

DROP TABLE MailBox;
 CREATE TABLE MailBox(
BoxNo integer,
MLocation varchar(30),
BSize varchar(20),
  primary key (BoxNo)
);
```

```
insert into MailBox values(10,'A-block','small');

insert into MailBox values(24,'A-block','medium');

insert into MailBox values(35,'B-block','large');

insert into MailBox values(47,'B-block','large');

insert into MailBox values(58,'C-block','small');

insert into MailBox values(69,'C-block','medium');

insert into MailBox values(70,'C-block','small');


select * from MailBox;


DROP TABLE ElectronicBox;

 CREATE TABLE ElectronicBox(

EBoxNo integer,

  Code integer,

  primary key (EBoxNo)

);


insert into ElectronicBox values(10,1467);

insert into ElectronicBox values(35,4678);

insert into ElectronicBox values(58,0986);

insert into ElectronicBox values(70,4673);


select * from ElectronicBox;
```

```
DROP TABLE LetterBox;
 CREATE TABLE LetterBox(
LBoxNo integer,
KeyNo integer,
  primary key (LBoxNo)
);

insert into LetterBox values(24,1524);
insert into LetterBox values(47,403);
insert into LetterBox values(69,2226);

select * from LetterBox;

DROP TABLE  MaintenanceTeam;
 CREATE TABLE MaintenanceTeam(
EId integer,
MName varchar(30) NOT NULL,
  Age integer,
EmailId varchar(40),
  primary key (EId)
);
```

```
insert into MaintenanceTeam values(1000,'Joe Root',28,'root@gmail.com');

insert into MaintenanceTeam values(1001,'Virat Kohli',30,'kohli@gmail.com');

insert into MaintenanceTeam values(1002,'Steven Smith',29,'ssmith@gmail.com');

insert into MaintenanceTeam values(1003,'Kane Williamson',29,'kw123@gmail.com');

insert into MaintenanceTeam values(1004,'AB Devilliers',34,'abd@gmail.com');

insert into MaintenanceTeam values(1005,'David Warner',33,'warnerd@gmail.com');


select * from MaintenanceTeam;


DROP TABLE Technician;

 CREATE TABLE Technician(

TEId integer,

TGrade varchar(20),

  primary key (TEId)

);



insert into Technician values(1000,'Intermediate');

insert into Technician values(1005,'Professional');


select * from Technician;


DROP TABLE Engineer;
```

```
 CREATE TABLE Engineer(
EEId integer,
Specilization varchar(20),
  primary key (EEId)
);

insert into Engineer values(1001,'Plumber');
insert into Engineer values(1002,'Carpenter');

select * from Engineer;

DROP TABLE Gardener;
 CREATE TABLE Gardener(
GEId integer,
Gtype varchar(10),
  primary key (GEId)
);

insert into Gardener values(1003,'watering');
insert into Gardener values(1004,'trimming');

select * from Gardener;
```

```
DROP TABLE Maintains;

 CREATE TABLE Maintains(

RequestID integer,

MEId integer,

  MSSN integer,

MRoomNo integer,

RequestDate Date DEFAULT (SYSDATE),

  Comments varchar(100),

  primary key (RequestID,MEId,MSSN,MRoomNo)

);



insert into Maintains values(1,1000,432093120,1524,DEFAULT,'Fan Not working');

insert into Maintains values(2,1001,432093123,403,'19-nov-18','Sink pipe broken');

insert into Maintains values(3,1002,432093120,1524,DEFAULT,'Wooden door broken');


select * from Maintains;


ALTER TABLE ElectronicBox ADD CONSTRAINT ak1 FOREIGN KEY(EBoxNo) REFERENCES MailBox(BoxNo)
ON DELETE CASCADE;

ALTER TABLE LetterBox ADD CONSTRAINT ak2 FOREIGN KEY(LBoxNo) REFERENCES MailBox(BoxNo) ON
DELETE CASCADE;

ALTER TABLE Resident ADD CONSTRAINT ak3 FOREIGN KEY(LAgentID) REFERENCES
LeasingAgent(AgentID) ON DELETE SET NULL;
```

ALTER TABLE Resident ADD CONSTRAINT ak4 FOREIGN KEY(MboxNo) REFERENCES MailBox(BoxNo) ON DELETE SET NULL;

ALTER TABLE Pets ADD CONSTRAINT ak5 FOREIGN KEY(RSSN) REFERENCES Resident(SSN) ON DELETE CASCADE;

ALTER TABLE AccountDetails ADD CONSTRAINT ak6 FOREIGN KEY(BSSN) REFERENCES Resident(SSN) ON DELETE CASCADE;

ALTER TABLE Stays ADD CONSTRAINT ak7 FOREIGN KEY(RSSN) REFERENCES Resident(SSN) ON DELETE CASCADE;

ALTER TABLE Stays ADD CONSTRAINT ak8 FOREIGN KEY(RoomNo) REFERENCES Flat(RoomNo) ON DELETE CASCADE;

ALTER TABLE Payment ADD CONSTRAINT ak9 FOREIGN KEY(RoomNo) REFERENCES Flat(RoomNo) ON DELETE CASCADE;

ALTER TABLE Amenities ADD CONSTRAINT ak10 FOREIGN KEY(ARoomNo) REFERENCES Flat(RoomNo) ON DELETE CASCADE;

ALTER TABLE Technician ADD CONSTRAINT ak11 FOREIGN KEY(TEId) REFERENCES MaintenanceTeam(EId) ON DELETE CASCADE;

ALTER TABLE Engineer ADD CONSTRAINT ak12 FOREIGN KEY(EEId) REFERENCES MaintenanceTeam(EId) ON DELETE CASCADE;

ALTER TABLE Gardener ADD CONSTRAINT ak13 FOREIGN KEY(GEId) REFERENCES MaintenanceTeam(EId) ON DELETE CASCADE;

ALTER TABLE Maintains ADD CONSTRAINT ak14 FOREIGN KEY(MSSN) REFERENCES Resident(SSN)ON DELETE CASCADE;

ALTER TABLE Maintains ADD CONSTRAINT ak15 FOREIGN KEY(MEId) REFERENCES MaintenanceTeam(EId)ON DELETE CASCADE;

ALTER TABLE Maintains ADD CONSTRAINT ak16 FOREIGN KEY(MRoomNo) REFERENCES Flat(RoomNo)ON DELETE CASCADE;

Normalization of Relational Schema

The following Functional Dependencies exists in the relational schema –

1.RESIDENT{SSN-> Res_Name, Mbox_No, Salary, #Pets, Profession, Email, Phone}

2.ACCOUNT DETAILS{ AccountNumber -> AccountType}

3.PETS{ RSSN, Petname ->Age, petType}

4.STAYS{ RSSN, RoomNo -> LeaseStartDate, LeaseEndDate}

5.PAYMENT{RoomNo, RentAmt ->DueDate, RentPaid}

6.LeasingAgent{AgentID-> AgentName, PhoneNumber, Email}

7.FLAT{ RoomNo-> TotalResidents, Area}

8.AMENITIES{ARoomNo, Membership ->price}

9.MAILBOX{ BoxNo -> MLocation, BoxSize}

10.ELECTRONICBOX{ EBoxNo -> code}

11.LETTERBOX { LBoxNo -> KeyNo}

12.MAINTENANCETEAM{ EID-.> MName, Age, EmailID}

13.TECHNICIAN{TeID->TGRADE}

14.ENGINEER{ EeID -> Specialization}

15.GARDENER{GeID-> GType}

16.MAINTAINS{ Rid, MeID, MSSN, MRoomNo -> RequestDate, Comments}

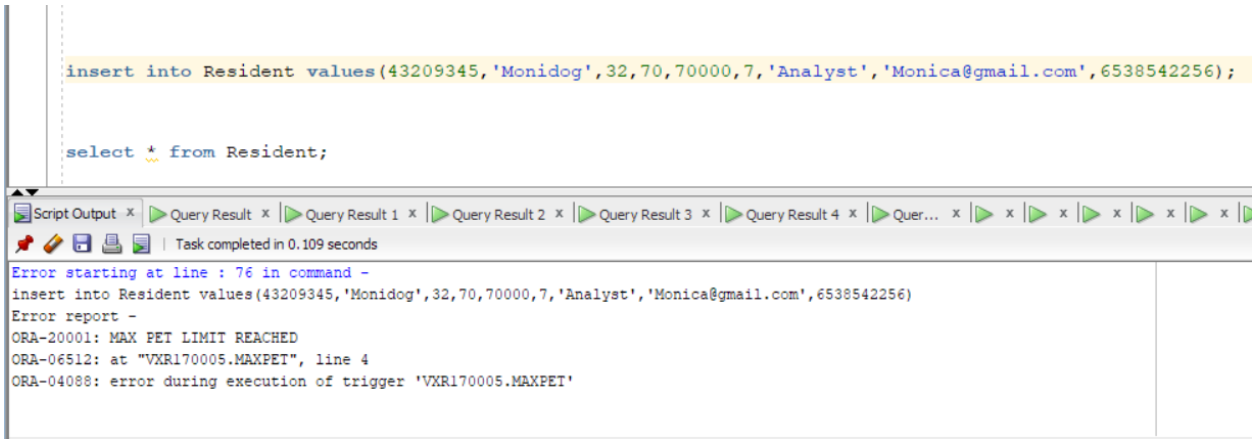
The above functional dependencies cause the schema to be in third normal form.

PL/SQL – Triggers

The following triggers are used to implement various requirements –

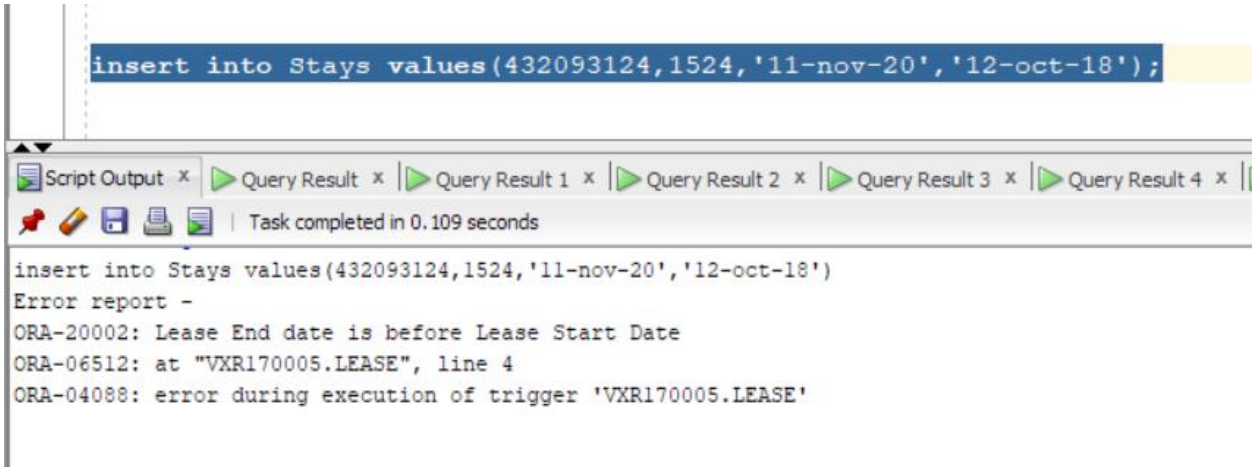
1) Trigger created to check if the no.of pets has reached the limit or not.

```
CREATE OR REPLACE TRIGGER maxpet
BEFORE INSERT OR UPDATE OF TotalPets ON Resident
FOR EACH ROW
DECLARE
BEGIN
IF (:new.TotalPets > 4) THEN
RAISE_APPLICATION_ERROR(-20001,'MAX PET LIMIT REACHED');
END IF;
END;
```



2) Trigger to check if the lease end date is before lease start date.

```
CREATE OR REPLACE TRIGGER Lease
  BEFORE INSERT OR UPDATE OF LeaseStartDate ON Stays
  FOR EACH row
  DECLARE
  BEGIN
    IF :new.LeaseEndDate < :new.LeaseStartDate THEN
      raise_application_error(-20002, ('Lease End date is before Lease Start Date'));
    END IF;
  END;
```



PL/SQL- Procedures

1) Procedure to check the rent payment.

```
CREATE OR REPLACE PROCEDURE p1 AS
 CURSOR c1 IS
 SELECT payment.RoomNo,payment.duedate,payment.rentpaid,payment.rentamount FROM Payment WHERE rentpaid = 'NO';
 Dd c1%ROWTYPE;
BEGIN
 OPEN c1;
LOOP
FETCH c1 INTO Dd;
EXIT WHEN (c1%NOTFOUND);?
DBMS_OUTPUT.PUT_LINE('Due Date '||Dd.RoomNo);
END LOOP;
CLOSE c1;
END;

EXECUTE p1;
```

2) Procedure to insert tuples into MailBox Table.

```
CREATE PROCEDURE
 amail( thismail IN MailBox.BoxNo%TYPE) AS
BEGIN
INSERT INTO MailBox VALUES(thismail,'C-block','small');
END;

BEGIN
amail(143);
amail(854);
END;
SELECT * FROM MailBox;
```