COMP40725- Introduction to Relational Database and SQL Programming

Project Final Submission

Rosemont Student Accommodation Database

Saransh Agarwal
School of Computer Science & Informatics
University College Dublin
Befield, Dublin 4.
saransh.agarwal@ucdconnect.ie

1 PROJECT DESCRIPTION

Rosemont Student Accommodation offers furnished student flats that consists of single- rooms for groups of students sharing common area and kitchen. Each flat includes a flat number, address and number of single- rooms available. Each single- room in a flat has a room number, fixed monthly charges. The rooms are rented on lease and each lease has a lease number, duration, start and end dates, and student number on it. The student data recorded by the accommodation office contains student number, name (first and last), date of birth, sex, degree studying, nationality and any special needs required. Also, every student is issued an invoice that has a unique invoice number, lease number, payment and student details. Moreover, in case of emergency, details of student's parents/ guardians are also stored that includes their name, relationship, address and a contact number. The flats are maintained by well trained staff that help student settle and provide assistance in cases. Also, an inspection is done on a routine basis to ensure that the flats are well maintained. The inspections are carried out by a staff member and details of which are recorded. Various courses are also organized by the Accommodation office for the well-being of the students which are delivered by external coaches.

2 BUSINESS RULES AND ASSUMPTIONS

- The number of single- rooms in a rented flat accommodation cannot exceed six;
- A member of staff cannot inspect more than 10 flats at the same time;
- The rent due on a room lease is calculated monthly;
- The duration of the lease can be semester wise, or for the summer term;
- Each student is associated with only one course;
- Every student is supposed to have a Guardian or a next- of- kin relationship;

3 ENTITY AND RELATIONSHIPS

3.1 FINAL ENTITIES (WITH ATTRIBUTES)-

- Student (<u>StudentNo</u>, fName, lName, addr, DOB, sex, degree, nationality, spl_req)
- Flat (<u>flatNo</u>, fAddr, no_of_rooms)
- Room (<u>roomNo</u>, rent, flatNo)
- Course (<u>courseNo</u>, c_title, coach)
- Staff (<u>staffID</u>, fName, lName, addr, position)
- Lease (<u>leaseNo</u>, length, StudentNo, roomNo, checkin_date, checkout_date)
- Invoice (<u>invNo</u>, payment, payment_date, payment_type, leaseNo)
- Inspection (<u>flatNo</u>, date_of_insp, remarks, StaffID)
- Guardian (StudentNo, name, g_addr, tel_no)
- Service (serviceNo, staffid)

3.2 FINAL RELATIONSHIPS (WITH CARDINALITY)-

- Every student requests for a lease agreement
  Student- Lease (1:1) requests for;
- Each flat contains number of rooms in it
  Flat- Rooms (1:m) contains;

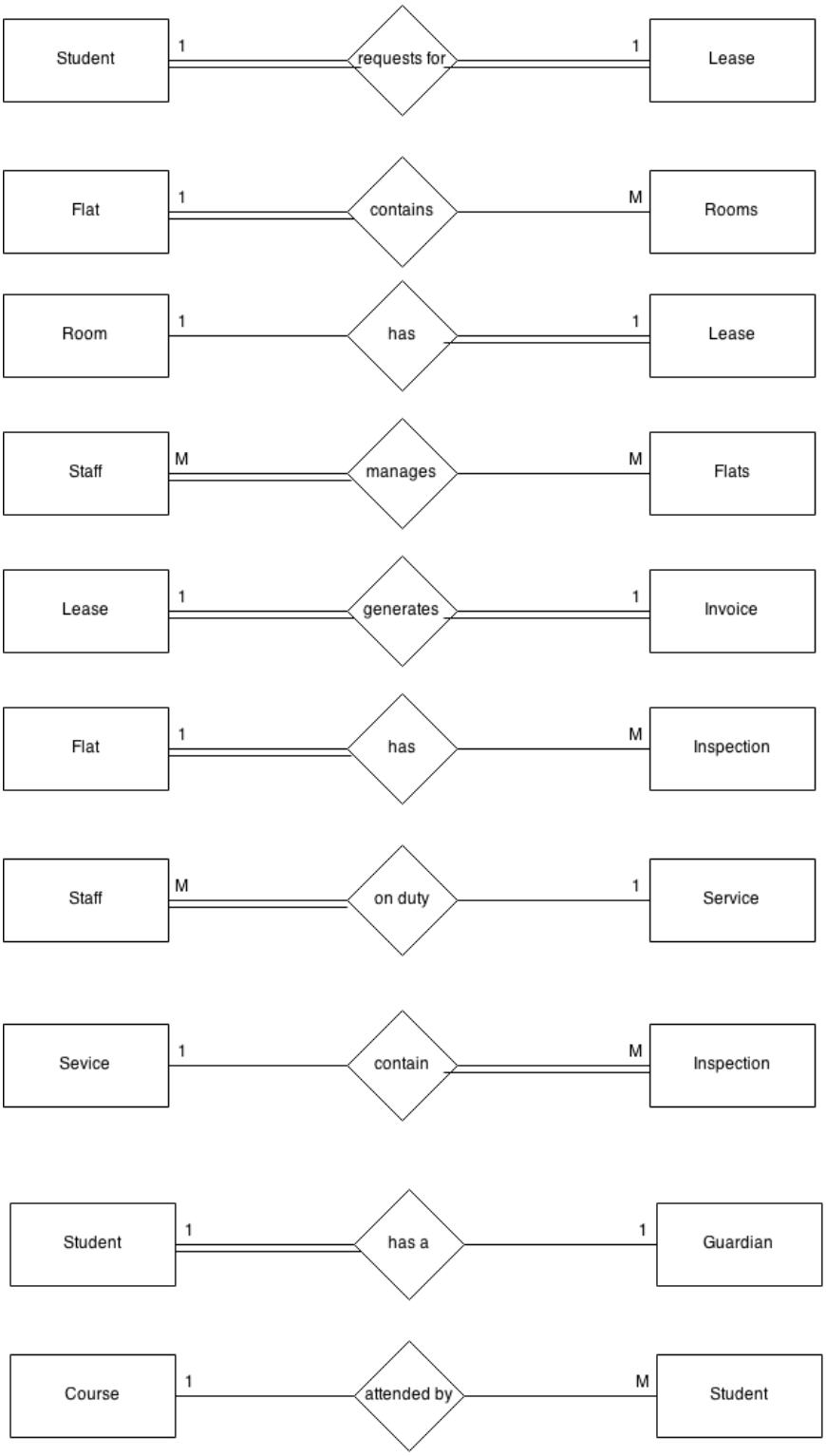
- Every room has a lease agreement requested by a student
  Room- Lease (1:1) has a;
- Staff manages flats in the accommodation
  Staff- Flats (m:m) manages;
- Lease results in generation of invoice in the name of student
  Lease- Invoice (1:m) generates;
- Each flat has a periodic inspection
  Flat- Inspection (1:m) has;
- Staff on duty in a service
  Staff- Service (m:1) on duty;
- Service contains Inspection
  Service- Inspection (1:m) contains;
- Student records have a guardian in case of emergency
  Student- Guardian (1:1) has a;
- A student can attend course at the accommodation
  Course- Student (1:m) attended by;

3.3 OPTIONALITY-

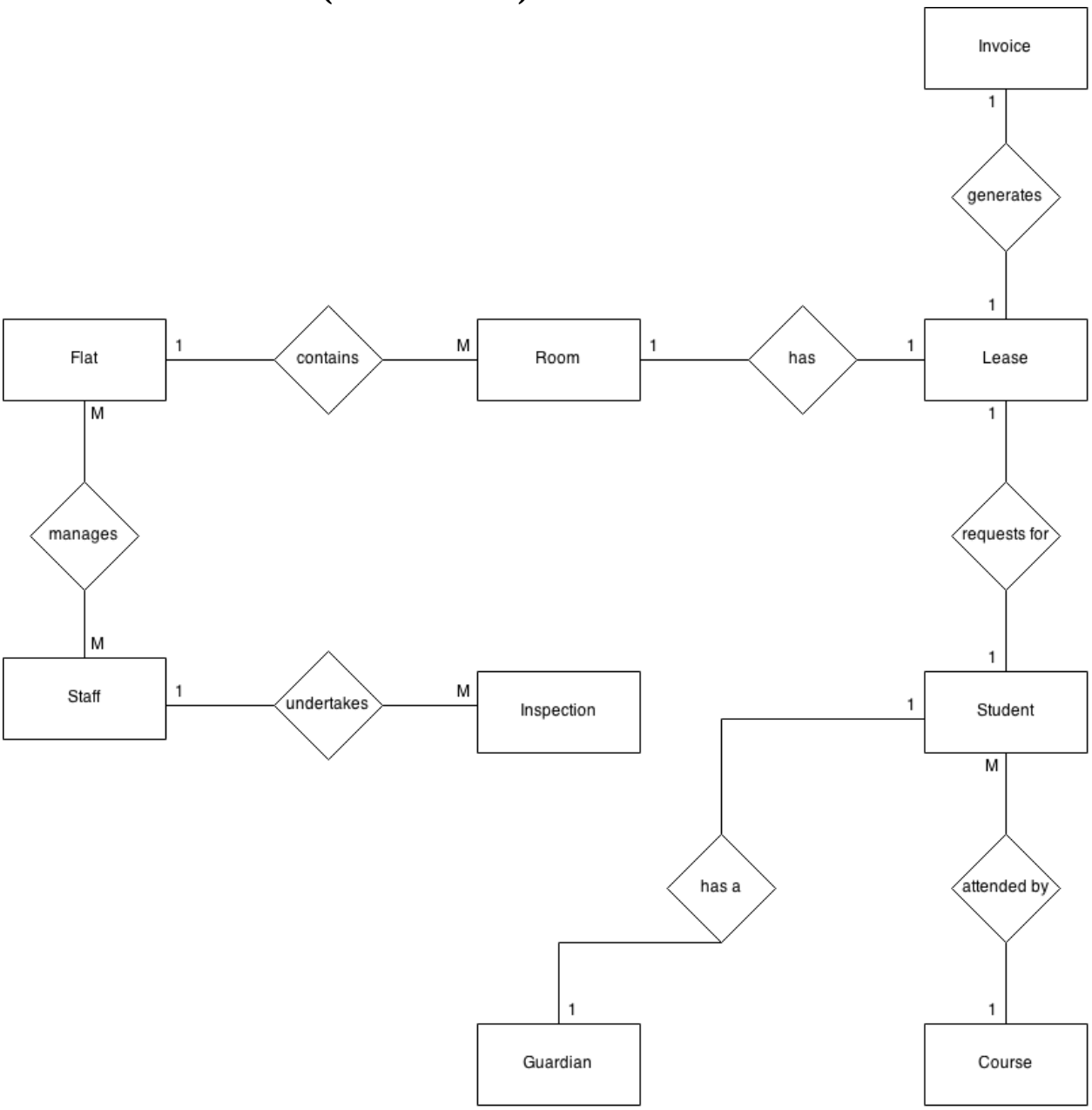
- A student is mandatory to a lease, so is the lease. Both, student and lease are mandatory;
- A flat must contain number of rooms, so flat is mandatory;
- A lease should be associated to a room held by a student, so lease is mandatory;
- A member of staff manages flats, so staff is mandatory;
- A lease results in generation of an invoice, and invoice is mandatory. So, both are mandatory;
- Inspection cannot be done without flat, so Flat is mandatory;
- Staff on duty is involved in a service, so Staff is mandatory.
- Each service contains Inspection which is a vital entity. Therefore, Inspection is mandatory.
- On the emergency records, a Student has a Guardian. So, Student is mandatory.
- A student can attend a course, both are optional.

4 ER DIAGRAM

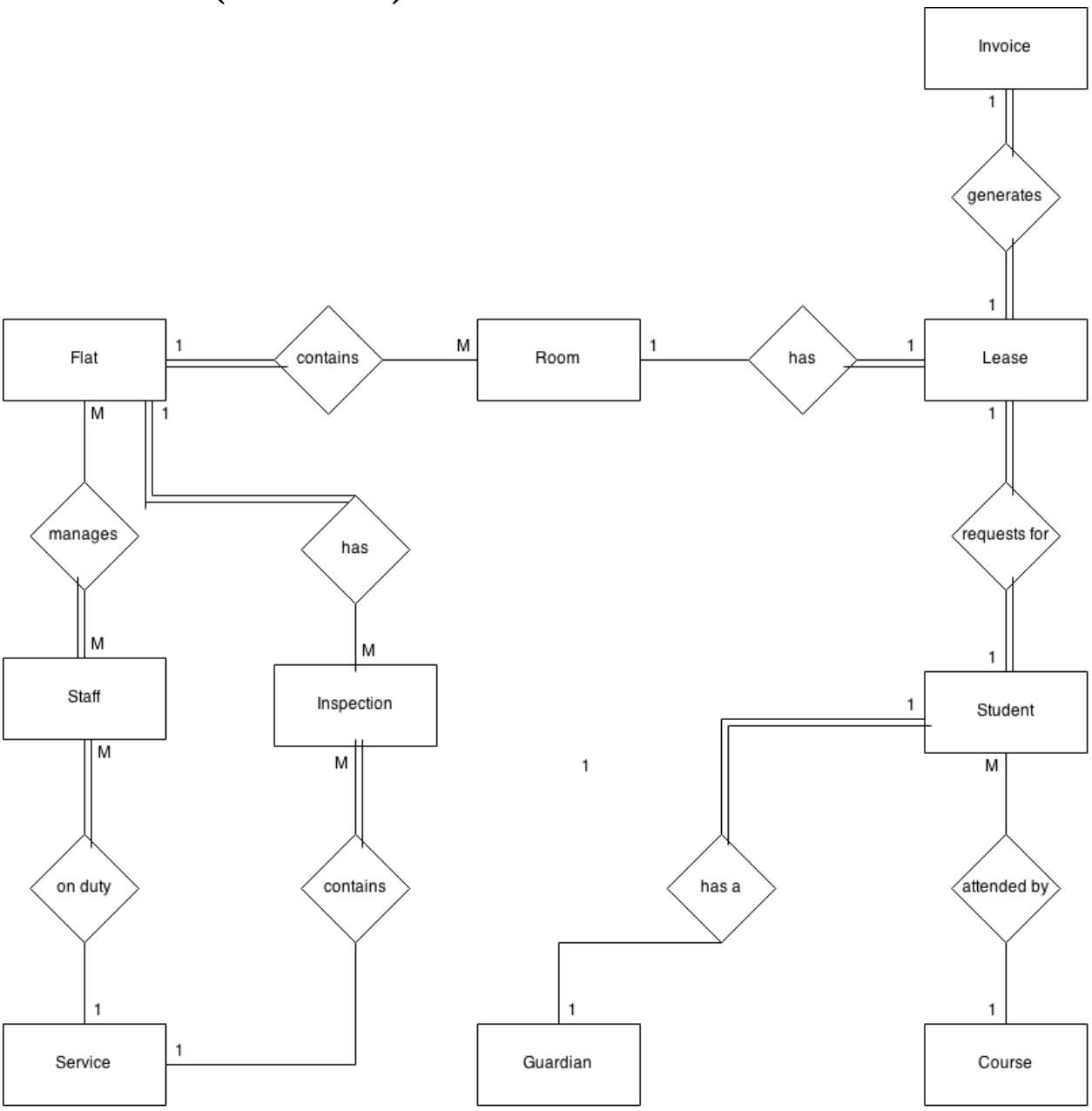
4.1 RELATIONSHIPS



4.2 INITIAL ER MODEL (BEFORE SPLIT)



4.3 ER MODEL (AFTER SPLIT)



5 DATABASE SETUP

This section details the creation of tables with suitable constraints and adding values into them.

5.1 INITIAL SETUP

```
-- Created a new workspace called 'Project'

-- Login into data base using credentials for new workspace

CONNECT;

-- entering login credentials here

-- to get output on screen

SET SERVEROUTPUT ON;
```

5.2 CREATING TABLES

5.2.1 Student Table

```
CREATE TABLE STUDENT (

StudentNo INT NOT NULL,

fName VARCHAR(10) NOT NULL,

-- first name

lName VARCHAR(10) NOT NULL,

-- last name

address VARCHAR(20) NOT NULL,

date_of_birth DATE NOT NULL,

sex VARCHAR(6) NOT NULL,

degree VARCHAR(5) NOT NULL,

nationality VARCHAR(10) NOT NULL,

spl_req VARCHAR(20),

-- Altered table, special needs aren't necessary

PRIMARY KEY (StudentNo)

-- defining a primary key

);
```

```
C:\Windows\system32\cmd.exe - sqlplus

Enter user-name: project
Enter password:

Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production

SQL> show user;
USER is "PROJECT"
SQL> CREATE TABLE STUDENT (
  2  StudentNo INT NOT NULL,
  3  fName VARCHAR(10) NOT NULL,
  4  lName VARCHAR(10) NOT NULL,
  5  address VARCHAR(20) NOT NULL,
  6  date_of_birth DATE NOT NULL,
  7  sex VARCHAR(6) NOT NULL,
  8  degree VARCHAR(5) NOT NULL,
  9  nationality VARCHAR(10) NOT NULL,
 10  spl_req VARCHAR(20) NOT NULL,
 11  PRIMARY KEY (StudentNo)
 12  );

Table created.

SQL>
```

5.2.2 Flat Table

```
CREATE TABLE Flat(

flatNo int NOT NULL,

faddr VARCHAR(20) NOT NULL,

no_of_rooms INT NOT NULL,

PRIMARY KEY (flatNo)

);
```

5.2.3 Room Table

```
CREATE TABLE Room(

roomNo int NOT NULL,

rent INT NOT NULL,

-- rent of each room

flatNo int NOT NULL,

-- flat in which room is located

PRIMARY KEY (roomNo),

FOREIGN KEY (flatNo) REFERENCES Flat(flatNo)

-- flatNo is the foreign key for table Room

-- which is primary key for table Flat

);
```

```
C:\Windows\system32\cmd.exe - sqlplus

SQL> drop table flat;

Table dropped.

SQL> CREATE TABLE Flat(
  2  flatNo int NOT NULL,
  3  faddr VARCHAR(20) NOT NULL,
  4  no_of_rooms INT NOT NULL,
  5  PRIMARY KEY (flatNo)
  6  );

Table created.

SQL> CREATE TABLE Room(
  2  roomNo int NOT NULL,
  3  rent INT NOT NULL,
  4  flatNo int NOT NULL,
  5  PRIMARY KEY (roomNo),
  6  FOREIGN KEY (flatNo) REFERENCES Flat(flatNo)
  7  );

Table created.

SQL> _
```

5.2.4 Course Table

```
CREATE TABLE Course(

courseNo int NOT NULL,

cTitle VARCHAR(10) NOT NULL,

-- title of courseNo

coach VARCHAR(10) NOT NULL,

-- as per the desc., coach is not member of staff

-- so no need to store other details

PRIMARY KEY (courseNo)

);
```

5.2.5 Staff Table

```
CREATE TABLE Staff(

staffid int NOT NULL,

fName VARCHAR(20) NOT NULL,

lName VARCHAR(20),

addr VARCHAR(20) NOT NULL,

position VARCHAR(10) NOT NULL,

PRIMARY KEY (staffid)

);
```

```
C:\Windows\system32\cmd.exe - sqlplus
SQL>
SQL>
SQL>
SQL>
SQL> CREATE TABLE Course(
  2  courseNo int NOT NULL,
  3  cTitle VARCHAR(10) NOT NULL,
  4  coach VARCHAR(10) NOT NULL,
  5  PRIMARY KEY (courseNo)
  6  );

Table created.

SQL> CREATE TABLE Staff(
  2  staffid int NOT NULL,
  3  fName VARCHAR(20) NOT NULL,
  4  lName VARCHAR(20),
  5  addr VARCHAR(20) NOT NULL,
  6  position VARCHAR(10) NOT NULL,
  7  PRIMARY KEY (staffid)
  8  );

Table created.

SQL>
```

5.2.6 Lease Table

```
CREATE TABLE Lease(

leaseNo int NOT NULL,

length int NOT NULL,

--length is in no of days

StudentNo INT NOT NULL,

roomNo int NOT NULL,

checkin DATE NOT NULL,

-- date when student checkin room

checkout DATE NOT NULL,

-- date when student checkout room

PRIMARY KEY (leaseNo),

FOREIGN KEY (StudentNo) REFERENCES STUDENT(StudentNo),

-- lease is dependent on studentNo

FOREIGN KEY (roomNo) REFERENCES Room(roomNo)

-- dependency on roomNo

);
```

```
C:\Windows\system32\cmd.exe - sqlplus
  3  fName VARCHAR(20) NOT NULL,
  4  lName VARCHAR(20),
  5  addr VARCHAR(20) NOT NULL,
  6  position VARCHAR(10) NOT NULL,
  7  PRIMARY KEY (staffid)
  8  );

Table created.

SQL> CREATE TABLE Lease(
  2  leaseNo int NOT NULL,
  3  length int NOT NULL,
  4  --length is in no of days
  5  StudentNo INT NOT NULL,
  6  roomNo int NOT NULL,
  7  checkin DATE NOT NULL,
  8  checkout DATE NOT NULL,
  9  PRIMARY KEY (leaseNo),
 10  FOREIGN KEY (StudentNo) REFERENCES STUDENT(StudentNo),
 11  FOREIGN KEY (roomNo) REFERENCES Room(roomNo)
 12  );

Table created.

SQL>
```

5.2.7 Invoice Table

```
CREATE TABLE Invoice(

invNo int NOT NULL,

payment int NOT NULL,

payment_date DATE NOT NULL,

payment_type VARCHAR(10) NOT NULL,

-- type includes card/ cash/ cheque

leaseNo int NOT NULL,

PRIMARY KEY (invNo),

FOREIGN KEY (leaseNo) REFERENCES Lease(leaseNo)

);
```

5.2.8 Inspection Table

```
CREATE TABLE Inspection(

date_of_insp DATE NOT NULL,

remarks VARCHAR(20) NOT NULL,

-- remarks like conditions are given by staff

flatNo int NOT NULL,

staffid int NOT NULL,

FOREIGN KEY (flatNo) REFERENCES Flat(flatNo),

FOREIGN KEY (staffid) REFERENCES Staff(staffid)

);
```

```
C:\Windows\system32\cmd.exe - sqlplus

SQL> CREATE TABLE Invoice(
  2  invNo int NOT NULL,
  3  payment int NOT NULL,
  4  payment_date DATE NOT NULL,
  5  payment_type VARCHAR(10) NOT NULL,
  6  leaseNo int NOT NULL,
  7  PRIMARY KEY (invNo),
  8  FOREIGN KEY (leaseNo) REFERENCES Lease(leaseNo)
  9  );

Table created.

SQL> CREATE TABLE Inspection(
  2  date_of_insp DATE NOT NULL,
  3  remarks VARCHAR(20) NOT NULL,
  4  flatNo int NOT NULL,
  5  staffid int NOT NULL,
  6  FOREIGN KEY (flatNo) REFERENCES Flat(flatNo),
  7  FOREIGN KEY (staffid) REFERENCES Staff(staffid)
  8  );

Table created.

SQL> _
```

5.2.9 Service Table

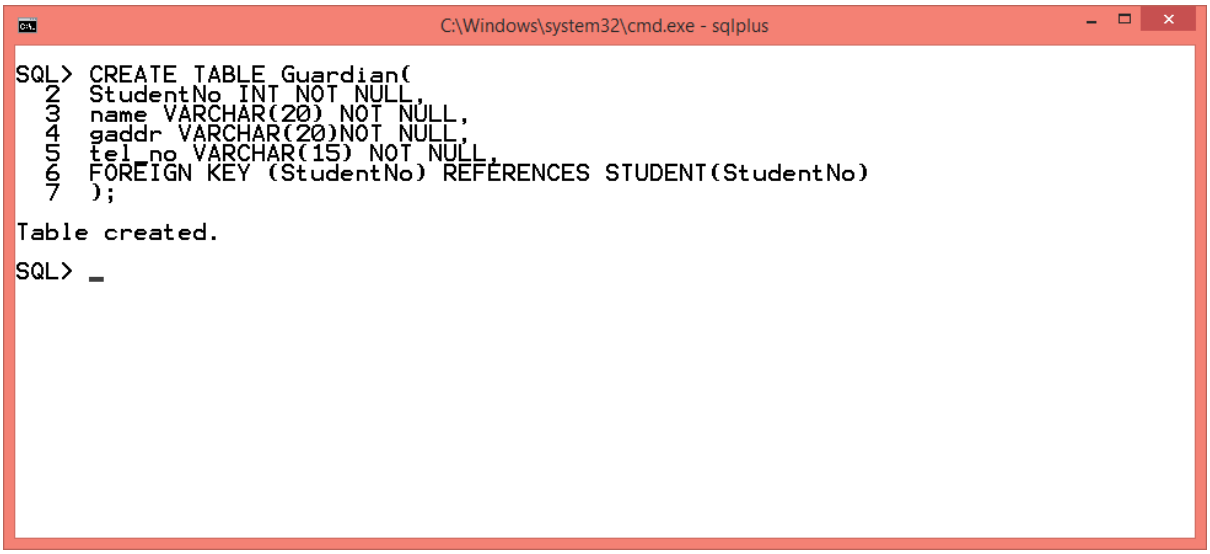
```
CREATE TABLE Service(

serviceNo int NOT NULL,

staffid int NOT NULL,

PRIMARY KEY (serviceNo),

FOREIGN KEY (staffid) REFERENCES Staff(staffid)

);
```

```
SQL> CREATE TABLE Service(
  2  serviceNo int NOT NULL,
  3  staffid int NOT NULL,
  4  PRIMARY KEY (serviceNo),
  5  FOREIGN KEY (staffid) REFERENCES Staff(staffid)
  6  );

Table created.

SQL> _
```

5.2.10 Guardian Table

```
CREATE TABLE Guardian(

StudentNo INT NOT NULL,

name VARCHAR(20) NOT NULL,

gaddr VARCHAR(20)NOT NULL,

-- address of guardian

tel_no VARCHAR(15) NOT NULL,

-- emergency tel_no of guardian

FOREIGN KEY (StudentNo) REFERENCES STUDENT(StudentNo)
```

```
);
```



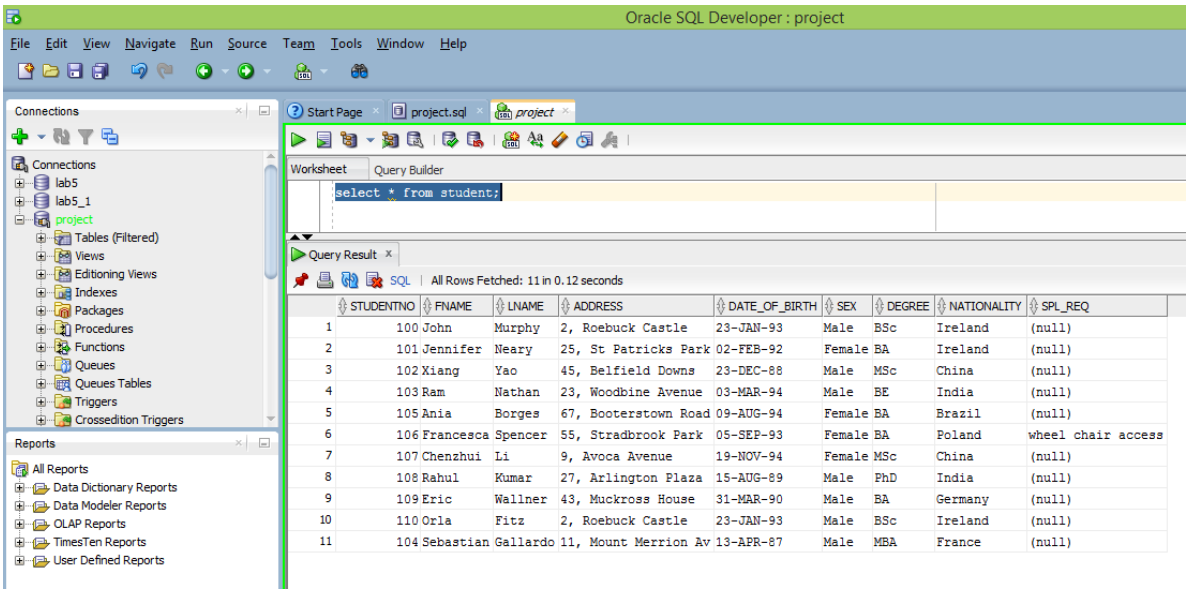
5.3 ADDING VALUES TO TABLES

5.3.1 Student Table

```
INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (100, 'John', 'Murphy', '2, Roebuck
Castle', '23-Jan-93', 'Male', 'BSc', 'Ireland');

INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (101, 'Jennifer', 'Neary', '25, St
Patricks Park', '2-Feb-92', 'Female', 'BA', 'Ireland');

INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (102, 'Xiang', 'Yao', '45, Belfield
Downs', '23-Dec-88', 'Male', 'MSc', 'China');

INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (103, 'Ram', 'Nathan', '23, Woodbine
Avenue', '3-Mar-94', 'Male', 'BE', 'India');

INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (104, 'Sebastian', 'Gallardo', '11,
Mount Merrion Av', '13-Apr-87', 'Male', 'MBA', 'France');

INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (105, 'Ania', 'Borges', '67,
Booterstown Road', '9-Aug-94', 'Female', 'BA', 'Brazil');

INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality, spl_req) VALUES (106, 'Francesca',
'Spencer', '55, Stradbrook Park', '5-Sep-93', 'Female', 'BA',
'Poland','wheel chair access');
```

```
INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (107, 'Chenzhui', 'Li', '9, Avoca
Avenue', '19-Nov-94', 'Female', 'MSc', 'China');

INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (108, 'Rahul', 'Kumar', '27,
Arlington Plaza', '15-Aug-89', 'Male', 'PhD', 'India');

INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (109, 'Eric', 'Wallner', '43,
Muckross House', '31-Mar-90', 'Male', 'BA', 'Germany');

INSERT INTO STUDENT (StudentNo, fName, lName, address, date_of_birth,
sex, degree, nationality) VALUES (110, 'Orla', 'Fitz', '2, Roebuck
Castle', '23-Jan-93', 'Male', 'BSc', 'Ireland');

--I've used SQL Developer to take the table as the output in command
prompt was too messy
```

```
C:\Windows\system32\cmd.exe - sqlplus
 STUDENTNO FNAME      LNAME      ADDRESS              DATE_OF_B SEX    DEGRE
---------- ---------- ---------- -------------------- --------- ------ -----
NATIONALIT SPL_REQ
---------- --------------------
       107 Chenzhui   Li         9, Avoca Avenue      19-NOV-94 Female MSc
China
       108 Rahul      Kumar      27, Arlington Plaza  15-AUG-89 Male   PhD
India
       109 Eric       Wallner    43, Muckross House   31-MAR-90 Male   BA
Germany

 STUDENTNO FNAME      LNAME      ADDRESS              DATE_OF_B SEX    DEGRE
---------- ---------- ---------- -------------------- --------- ------ -----
NATIONALIT SPL_REQ
---------- --------------------
       110 Orla       Fitz       2, Roebuck Castle    23-JAN-93 Male   BSc
Ireland

10 rows selected.
SQL> _
```



5.3.2 Flat Table

```
INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (1, '2,Mount
Merrion',4);

INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (2, '3,Mount
Merrion',5);

INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (3, '3,Mount
Merrion',5);

INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (4, '3,Mount
Merrion',5);

INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (5, '2,Mount
Merrion',4);

INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (6, '2,Mount
Merrion',4);

INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (7, '2,Mount
Merrion',4);

INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (8, '5,Mount
Merrion',6);

INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (9, '5,Mount
Merrion',6);

INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (10, '5,Mount
Merrion',6);
```

```
C:\Windows\system32\cmd.exe - sqlplus

SQL> SELECT * FROM Flat;

    FLATNO FADDR                 NO_OF_ROOMS
---------- -------------------- -----------
         1 2,Mount Merrion                4
         2 3,Mount Merrion                5
         3 3,Mount Merrion                5
         4 3,Mount Merrion                5
         5 2,Mount Merrion                4
         6 2,Mount Merrion                4
         7 2,Mount Merrion                4
         8 5,Mount Merrion                6
         9 5,Mount Merrion                6
        10 5,Mount Merrion                6

10 rows selected.

SQL> _
```

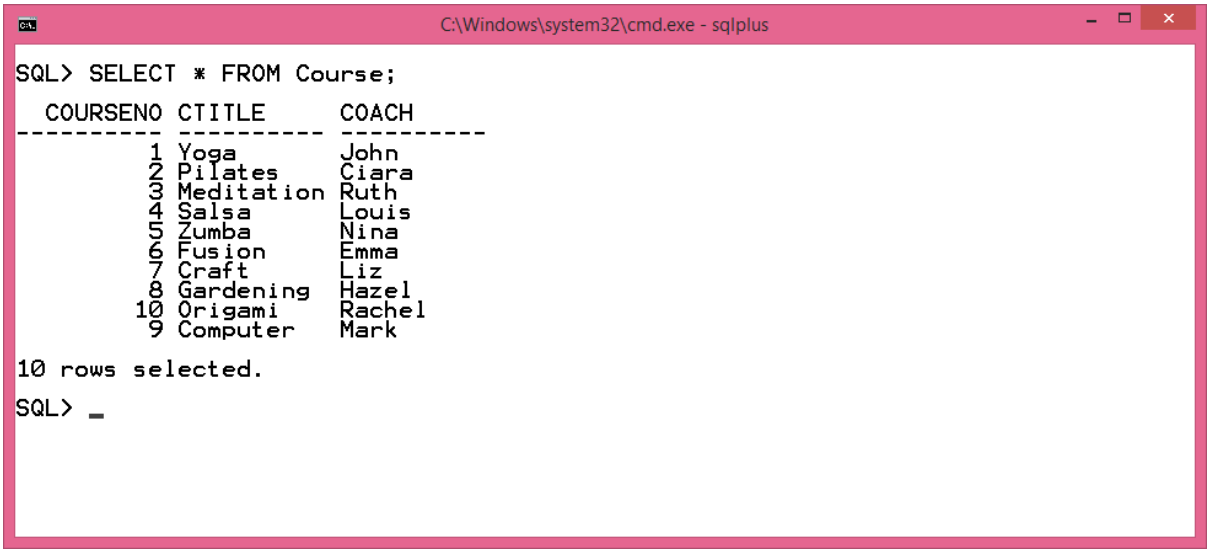
5.3.3 Table Room

```
INSERT INTO Room (roomNo, rent, flatNo) VALUES (21, 500, 2);

INSERT INTO Room (roomNo, rent, flatNo) VALUES (22, 500, 2);

INSERT INTO Room (roomNo, rent, flatNo) VALUES (11, 600, 1);

INSERT INTO Room (roomNo, rent, flatNo) VALUES (12, 600, 5);

INSERT INTO Room (roomNo, rent, flatNo) VALUES (13, 600, 1);

INSERT INTO Room (roomNo, rent, flatNo) VALUES (23, 500, 4);

INSERT INTO Room (roomNo, rent, flatNo) VALUES (31, 450, 9);

INSERT INTO Room (roomNo, rent, flatNo) VALUES (32, 450, 10);

INSERT INTO Room (roomNo, rent, flatNo) VALUES (33, 450, 8);

INSERT INTO Room (roomNo, rent, flatNo) VALUES (24, 500, 4);
```

```
C:\Windows\system32\cmd.exe - sqlplus

SQL> SELECT * FROM Room;

    ROOMNO       RENT     FLATNO
---------- ---------- ----------
        21        500          2
        22        500          2
        11        600          1
        12        600          5
        13        600          1
        23        500          4
        31        450          9
        32        450         10
        33        450          8
        24        500          4

10 rows selected.

SQL>
```

5.3.4 Table Course

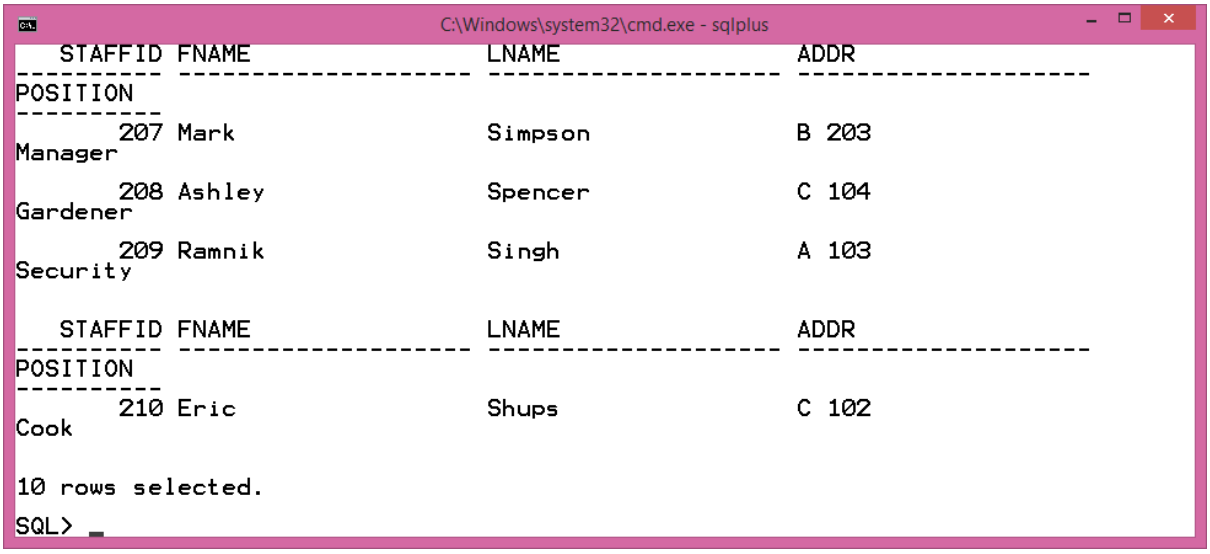
```
INSERT INTO Course (courseNo, cTitle, coach) VALUES (1, 'Yoga',
'John');

INSERT INTO Course (courseNo, cTitle, coach) VALUES (2,'Pilates',
'Ciara');

INSERT INTO Course (courseNo, cTitle, coach) VALUES
(3,'Meditation','Ruth');

INSERT INTO Course (courseNo, cTitle, coach) VALUES
(4,'Salsa','Louis');

INSERT INTO Course (courseNo, cTitle, coach) VALUES (5, 'Zumba',
'Nina');

INSERT INTO Course (courseNo, cTitle, coach) VALUES
(6,'Fusion','Emma');

INSERT INTO Course (courseNo, cTitle, coach) VALUES (7,'Craft','Liz');

INSERT INTO Course (courseNo, cTitle, coach) VALUES
(8,'Gardening','Hazel');

INSERT INTO Course (courseNo, cTitle, coach) VALUES
(9,'Computer','Mark');

INSERT INTO Course (courseNo, cTitle, coach) VALUES
(10,'Origami','Rachel');
```

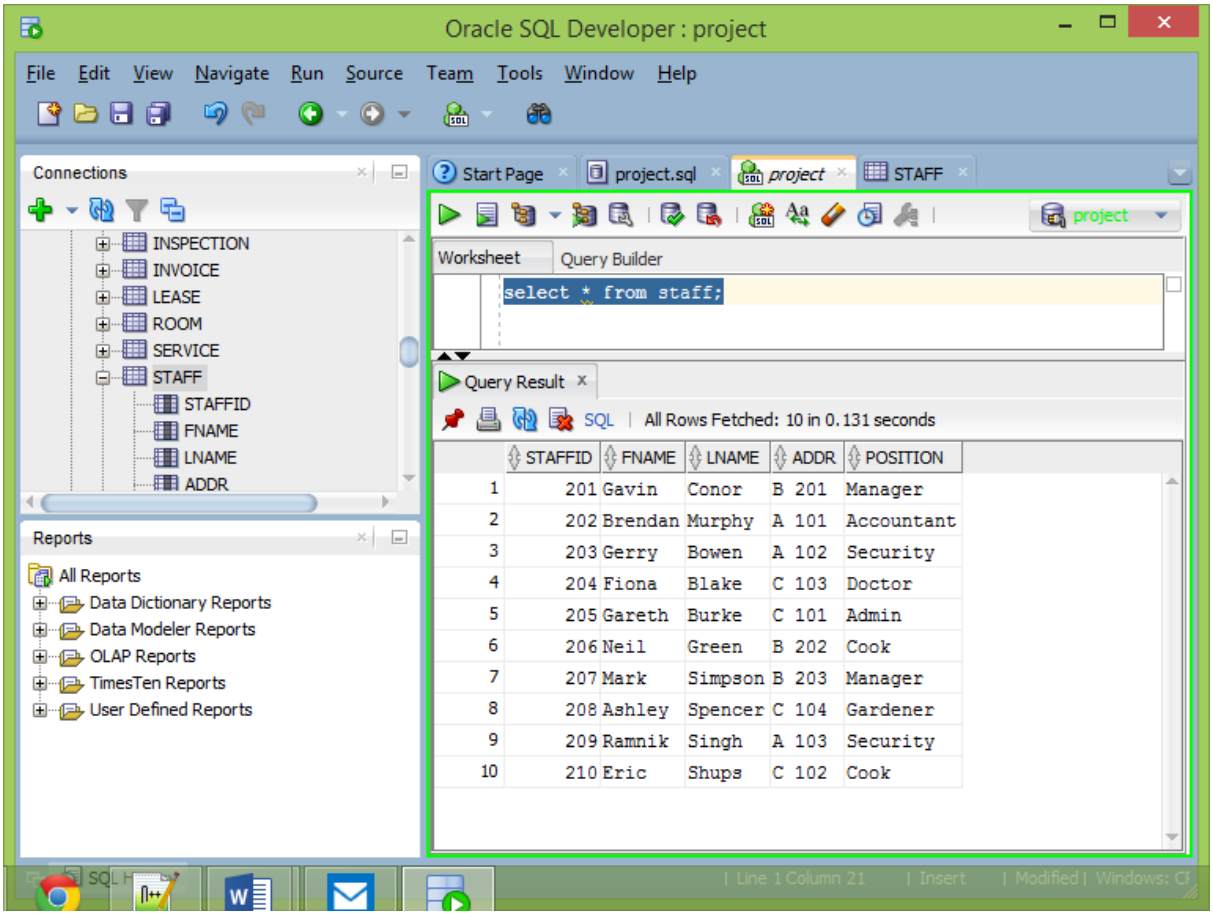


5.3.5 Table Staff

```
INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (201,
'Gavin', 'Conor', 'B 201', 'Manager');

INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (202,
'Brendan', 'Murphy', 'A 101', 'Accountant');
```

```
INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (203,
'Gerry', 'Bowen', 'A 102','Security');

INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (204,
'Fiona','Blake','C 103','Doctor');

INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (205,
'Gareth','Burke','C 101','Admin');

INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (206,
'Neil', 'Green','B 202','Cook');

INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (207,
'Mark', 'Simpson', 'B 203', 'Manager');

INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (208,
'Ashley', 'Spencer', 'C 104','Gardener');

INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (209,
'Ramnik', 'Singh','A 103','Security');

INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (210,
'Eric', 'Shups', 'C 102','Cook');

--I've used SQL Developer to take the table as the output in command
prompt was too messy
```

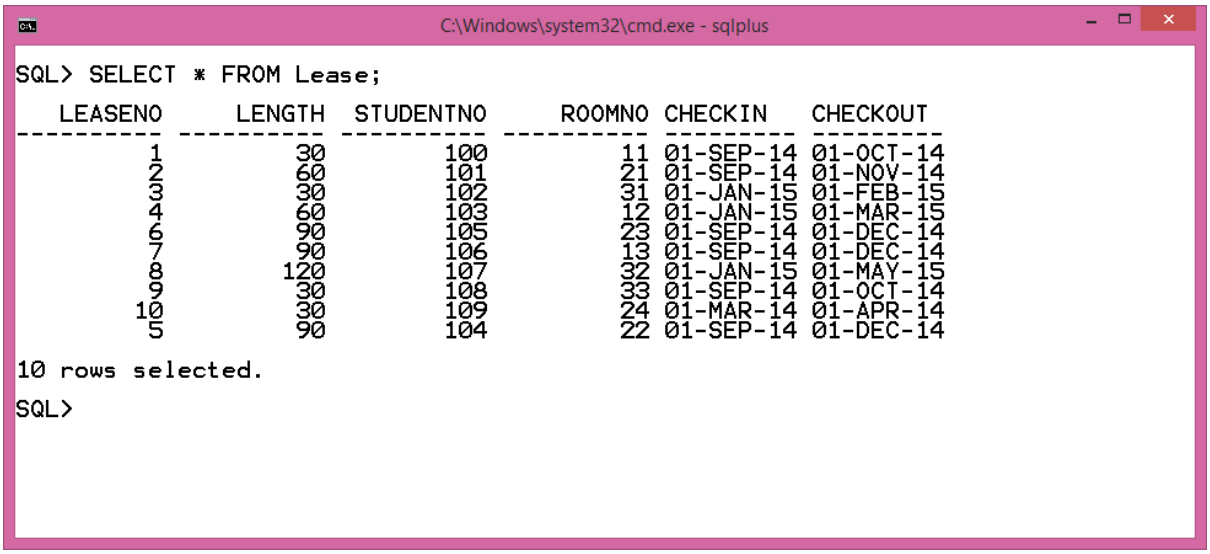




5.3.6 Table Lease

```
INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (1,30,100,11,'01-Sep-14','01-Oct-14');

INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (2,60,101,21,'01-Sep-14','01-Nov-14');

INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (3,30,102,31,'01-Jan-15','01-Feb-15');

INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (4,60,103,12,'01-Jan-15','01-Mar-15');

INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (5,90,104,22,'01-Sep-14','01-Dec-14');

INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (6,90,105,23,'01-Sep-14','01-Dec-14');

INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (7,90,106,13,'01-Sep-14','01-Dec-14');

INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (8,120,107,32,'01-Jan-15','01-May-15');
```

```
INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (9,30,108,33,'01-Sep-14','01-Oct-14');

INSERT INTO Lease (leaseNo, length, studentNo, roomNo, checkin,
checkout) VALUES (10,30,109,24,'01-Mar-14','01-Apr-14');
```



```
C:\Windows\system32\cmd.exe - sqlplus

SQL> SELECT * FROM Lease;

   LEASENO     LENGTH  STUDENTNO     ROOMNO CHECKIN   CHECKOUT
---------- ---------- ---------- ---------- --------- ---------
         1         30        100         11 01-SEP-14 01-OCT-14
         2         60        101         21 01-SEP-14 01-NOV-14
         3         30        102         31 01-JAN-15 01-FEB-15
         4         60        103         12 01-JAN-15 01-MAR-15
         6         90        105         23 01-SEP-14 01-DEC-14
         7         90        106         13 01-SEP-14 01-DEC-14
         8        120        107         32 01-JAN-15 01-MAY-15
         9         30        108         33 01-SEP-14 01-OCT-14
        10         30        109         24 01-MAR-14 01-APR-14
         5         90        104         22 01-SEP-14 01-DEC-14

10 rows selected.

SQL>
```

5.3.7 Table Invoice

```
INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (141,600,'01-Oct-14','Cash',1);

INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (142,1000,'01-Nov-14','Cheque',2);

INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (143,1250,'01-Feb-15','Cash',3);

INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (144,1200,'01-Mar-15','Card',4);

INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (145,1500,'01-Dec-14','Cash',5);

INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (146,1500,'01-Dec-14','Card',6);

INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (147,1800,'01-Dec-14','Card',7);

INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (148,1800,'01-May-15','Cheque',8);

INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (149,450,'01-Oct-14','Cheque',9);

INSERT INTO Invoice (invNo, payment,payment_date,payment_type,leaseNo)
VALUES (150,500,'01-Apr-14','Cash',10);
```

```
C:\Windows\system32\cmd.exe - sqlplus

SQL> SELECT * FROM Invoice;

     INVNO    PAYMENT PAYMENT_D PAYMENT_TY    LEASENO
---------- --------- --------- ---------- ----------
       141       600 01-OCT-14 Cash                1
       142      1000 01-NOV-14 Cheque              2
       143      1250 01-FEB-15 Cash                3
       144      1200 01-MAR-15 Card                4
       145      1500 01-DEC-14 Cash                5
       146      1500 01-DEC-14 Card                6
       147      1800 01-DEC-14 Card                7
       148      1800 01-MAY-15 Cheque              8
       149       450 01-OCT-14 Cheque              9
       150       500 01-APR-14 Cash               10

10 rows selected.

SQL>
```

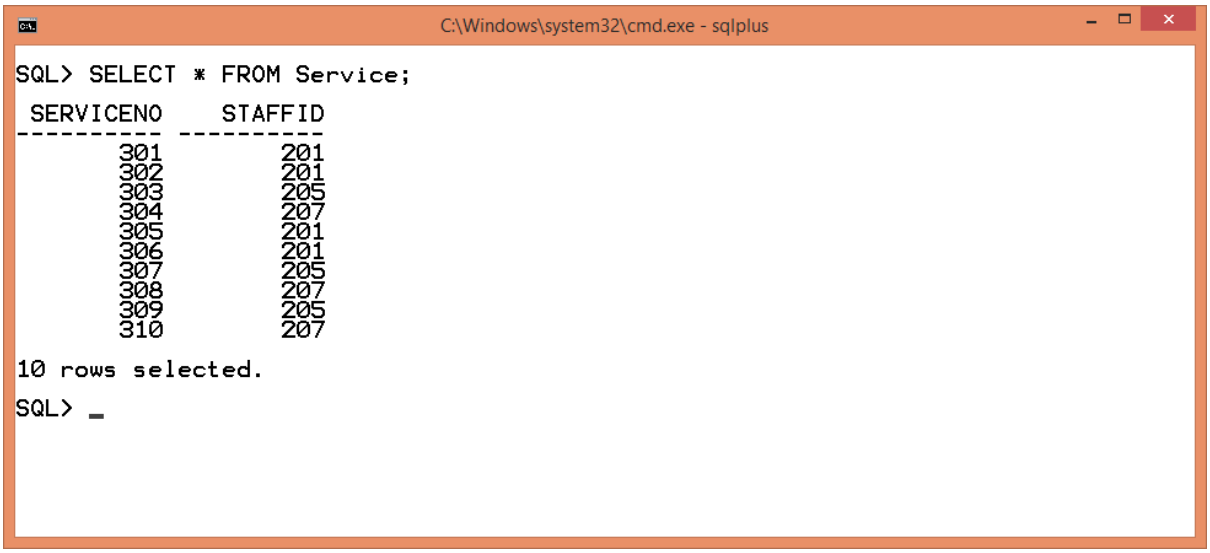
5.3.8 Table Inspection

```
INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(1,'15-Oct-14','Clean','201');

INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(2,'15-Oct-14','Dirty','201');

INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(3,'15-Apr-15','Smelly','205');

INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(4,'15-Oct-14','Wash','205');

INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(5,'15-Oct-14','Dirty','201');

INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(6,'01-Oct-14','Clean','207');

INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(7,'25-Oct-14','Clean','201');

INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(8,'15-Apr-15','Smelly','205');

INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(9,'05-Mar-15','Clean','207');

INSERT INTO Inspection (flatNo, date_of_insp, remarks, staffid) VALUES
(10,'15-Oct-14','Wash','207');
```

```
C:\Windows\system32\cmd.exe - sqlplus

SQL> SEWLECT * FROM Inspection;
SP2-0734: unknown command beginning "SEWLECT * ..." - rest of line ignored.
SQL> SELECT * FROM Inspection;

DATE_OF_I REMARKS                  FLATNO    STAFFID
--------- -------------------- ---------- ----------
15-OCT-14 Clean                         1        201
15-OCT-14 Dirty                         2        201
15-APR-15 Smelly                        3        205
15-OCT-14 Wash                          4        205
15-OCT-14 Dirty                         5        201
01-OCT-14 Clean                         6        207
25-OCT-14 Clean                         7        201
15-APR-15 Smelly                        8        205
05-MAR-15 Clean                         9        207
15-OCT-14 Wash                         10        207

10 rows selected.

SQL>
```

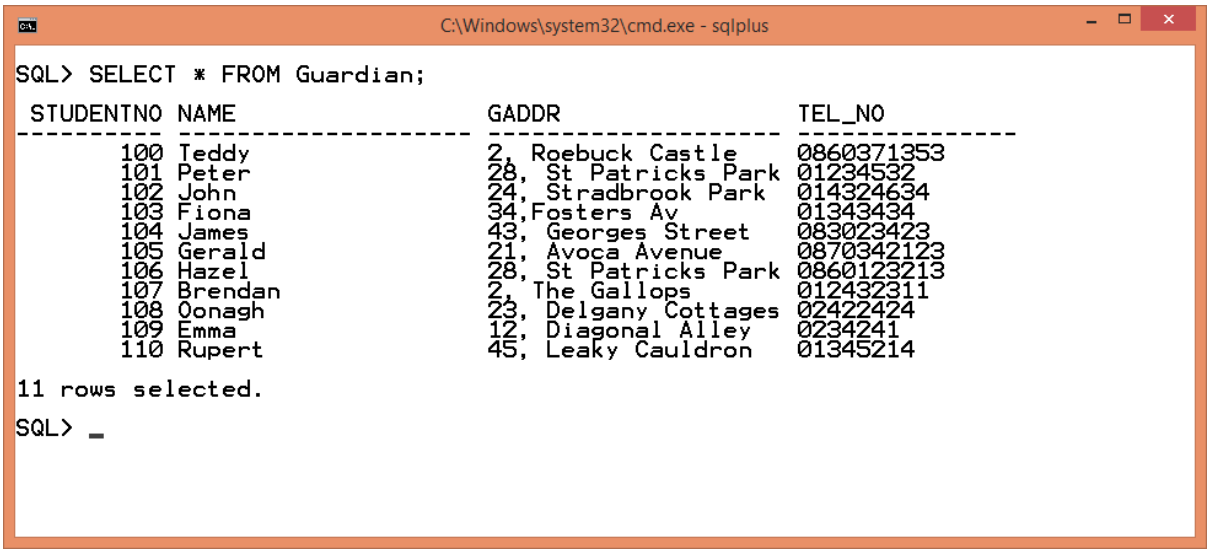
5.3.9 Table Service

```
INSERT INTO Service (serviceNo, staffid) VALUES (301,201);

INSERT INTO Service (serviceNo, staffid) VALUES (302,201);

INSERT INTO Service (serviceNo, staffid) VALUES (303,205);

INSERT INTO Service (serviceNo, staffid) VALUES (304,207);

INSERT INTO Service (serviceNo, staffid) VALUES (305,201);

INSERT INTO Service (serviceNo, staffid) VALUES (306,201);

INSERT INTO Service (serviceNo, staffid) VALUES (307,205);

INSERT INTO Service (serviceNo, staffid) VALUES (308,207);

INSERT INTO Service (serviceNo, staffid) VALUES (309,205);

INSERT INTO Service (serviceNo, staffid) VALUES (310,207);
```



5.3.10 Table Guardian

```
INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES (100,
'Teddy','2, Roebuck Castle', '0860371353');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES
(101,'Peter','28, St Patricks Park','01234532');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES
(102,'John','24, Stradbrook Park','014324634');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES
(103,'Fiona','34,Fosters Av','01343434');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES (104,
'James', '43, Georges Street','083023423');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES
(105,'Gerald','21, Avoca Avenue','0870342123');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES
(106,'Hazel','28, St Patricks Park','0860123213');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES
(107,'Brendan','2, The Gallops', '012432311');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES
(108,'Oonagh','23, Delgany Cottages','02422424');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES
(109,'Emma','12, Diagonal Alley','0234241');

INSERT INTO Guardian (studentNo, name,gaddr,tel_no) VALUES
(110,'Rupert','45, Leaky Cauldron','01345214');
```



```
C:\Windows\system32\cmd.exe - sqlplus

SQL> SELECT * FROM Guardian;

 STUDENTNO NAME                 GADDR                TEL_NO
---------- -------------------- -------------------- ---------------
       100 Teddy                2, Roebuck Castle    0860371353
       101 Peter                28, St Patricks Park 01234532
       102 John                 24, Stradbrook Park  014324634
       103 Fiona                34,Fosters Av        01343434
       104 James                43, Georges Street   083023423
       105 Gerald               21, Avoca Avenue     0870342123
       106 Hazel                28, St Patricks Park 0860123213
       107 Brendan              2, The Gallops       012432311
       108 Oonagh               23, Delgany Cottages 02422424
       109 Emma                 12, Diagonal Alley   0234241
       110 Rupert               45, Leaky Cauldron   01345214

11 rows selected.

SQL> _
```

6 JOINS

6.1 INNER JOINS

6.1.1 Show the list of student details along with the name of guardian and emergency contact number.

```
/* Query 1

Displaying student details along with respective guardian

*/

SELECT s.fname, s.lname, g.name,g.tel_no FROM STUDENT s JOIN Guardian
g

ON s.studentNo= g.studentNo;

-- since the column names were similar

-- aliases have been used
```

```
C:\Windows\system32\cmd.exe - sqlplus

11 rows selected.

SQL>
SQL> SELECT s.fname, s.lname, g.name,g.tel_no FROM STUDENT s JOIN Guardian g
  2  ON s.studentNo= g.studentNo;

FNAME      LNAME      NAME                 TEL_NO
---------- ---------- -------------------- ---------------
John       Murphy     Teddy                0860371353
Jennifer   Neary      Peter                01234532
Xiang      Yao        John                 014324634
Ram        Nathan     Fiona                01343434
Sebastian  Gallardo   James                083023423
Ania       Borges     Gerald               0870342123
Francesca  Spencer    Hazel                0860123213
Chenzhui   Li         Brendan              012432311
Rahul      Kumar      Oonagh               02422424
Eric       Wallner    Emma                 0234241
Orla       Fitz       Rupert               01345214

11 rows selected.

SQL>
```

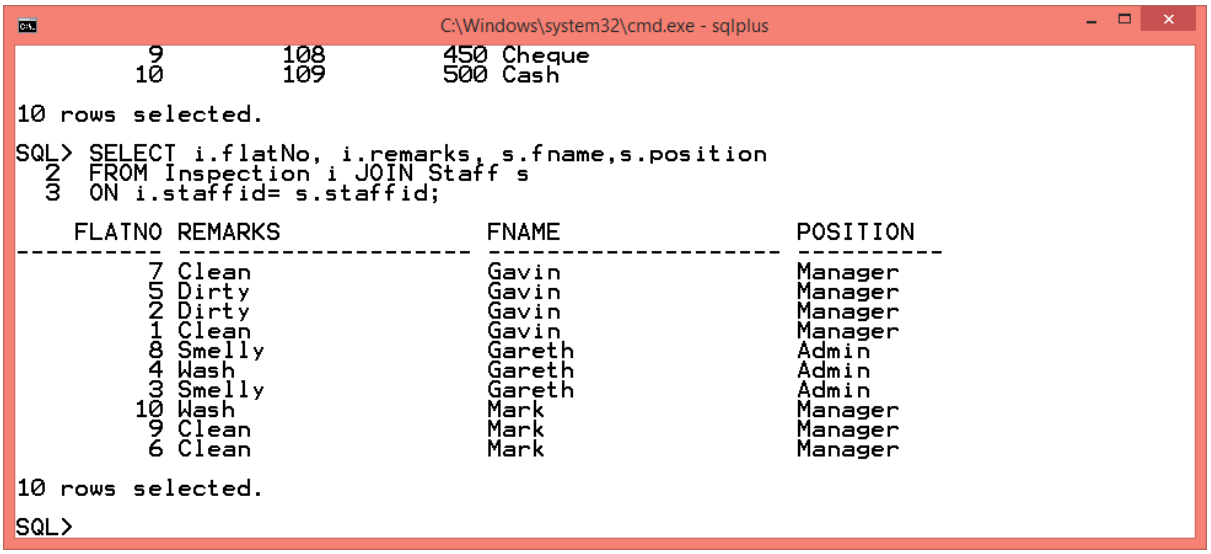
6.1.2 Show the details of each lease along with invoice associated with it, showing the amount paid and its payment type

```
/*Query 2

Display list of lease details with invoice associated with it, payment
amount and payment type

*/

SELECT l.leaseNo, l.studentNo, i.payment, i.payment_type

FROM Lease l JOIN Invoice i

-- since the column names were similar

-- aliases have been used

ON l.leaseNo= i.leaseNo;
```

```
C:\Windows\system32\cmd.exe - sqlplus
Eric       Wallner    Emma                 0234241
Orla       Fitz       Rupert               01345214

11 rows selected.

SQL> SELECT l.leaseNo, l.studentNo, i.payment, i.payment_type
  2  FROM Lease l JOIN Invoice i
  3  ON l.leaseNo= i.leaseNo;

   LEASENO  STUDENTNO    PAYMENT PAYMENT_TY
---------- ---------- ---------- ----------
         1        100        600 Cash
         2        101       1000 Cheque
         3        102       1250 Cash
         4        103       1200 Card
         5        104       1500 Cash
         6        105       1500 Card
         7        106       1800 Card
         8        107       1800 Cheque
         9        108        450 Cheque
        10        109        500 Cash

10 rows selected.

SQL> _
```

6.1.3 List all the details of the inspection conducted on a flat and the staff who did the inspection on a particular flat. Also give the position of the staff.

```
/*Query 3

Display inspection details, like flat and remarks along with staff
associated with it

*/

SELECT i.flatNo, i.remarks, s.fname,s.position

FROM Inspection i JOIN Staff s

-- since the column names were similar

-- aliases have been used

ON i.staffid= s.staffid;
```



```
C:\Windows\system32\cmd.exe - sqlplus

         9        108        450 Cheque
        10        109        500 Cash

10 rows selected.

SQL> SELECT i.flatNo, i.remarks, s.fname,s.position
  2  FROM Inspection i JOIN Staff s
  3  ON i.staffid= s.staffid;

    FLATNO REMARKS              FNAME                POSITION
---------- -------------------- -------------------- ----------
         7 Clean                Gavin                Manager
         5 Dirty                Gavin                Manager
         2 Dirty                Gavin                Manager
         1 Clean                Gavin                Manager
         8 Smelly               Gareth               Admin
         4 Wash                 Gareth               Admin
         3 Smelly               Gareth               Admin
        10 Wash                 Mark                 Manager
         9 Clean                Mark                 Manager
         6 Clean                Mark                 Manager

10 rows selected.

SQL>
```

6.1.4 Give details of the rooms, the rent on a room with the number of flat it is situated in and its address.

```
/*Query 4

List the rooms and its rent along with flat associated with them and
its address

*/

SELECT r.roomNo,r.rent, f.flatNo,f.faddr

FROM Room r JOIN Flat f

-- since the column names were similar

-- aliases have been used

ON r.flatNo= f.flatNo;
```

```
C:\Windows\system32\cmd.exe - sqlplus
         9 Clean                    Mark                    Manager
         6 Clean                    Mark                    Manager

10 rows selected.

SQL> SELECT r.roomNo,r.rent, f.flatNo,f.faddr
  2  FROM Room r JOIN Flat f
  3  ON r.flatNo= f.flatNo;

    ROOMNO       RENT     FLATNO FADDR
---------- ---------- ---------- --------------------
        13        600          1 2,Mount Merrion
        11        600          1 2,Mount Merrion
        22        500          2 3,Mount Merrion
        21        500          2 3,Mount Merrion
        24        500          4 3,Mount Merrion
        23        500          4 3,Mount Merrion
        12        600          5 2,Mount Merrion
        33        450          8 5,Mount Merrion
        31        450          9 5,Mount Merrion
        32        450         10 5,Mount Merrion

10 rows selected.

SQL> _
```

6.2 OUTER JOINS

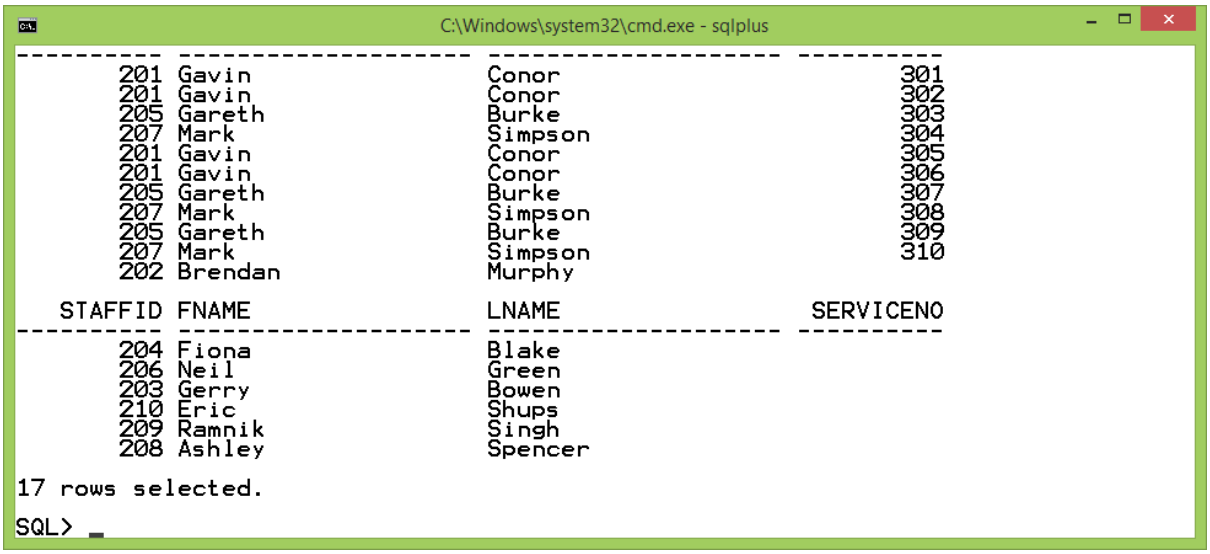
6.2.1 Create a left outer join to list all the flats and remarks given by staff members during periodic inspection.

```
/* Query 1

List the flat number and remarks given by respective staff members

who inspected the flat.

*/

SELECT s.fname,s.position,i.flatNo, i.remarks

FROM Inspection i LEFT OUTER JOIN Staff s

ON i.staffid= s.staffid;
```

```
C:\Windows\system32\cmd.exe - sqlplus
Enter password:

Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production

SQL> SELECT s.fname,s.position,i.flatNo, i.remarks
  2  FROM Inspection i LEFT OUTER JOIN Staff s
  3  ON i.staffid= s.staffid;

FNAME                POSITION       FLATNO REMARKS
-------------------- ---------- ---------- --------------------
Gavin                Manager             1 Clean
Gavin                Manager             2 Dirty
Gareth               Admin               3 Smelly
Gareth               Admin               4 Wash
Gavin                Manager             5 Dirty
Mark                 Manager             6 Clean
Gavin                Manager             7 Clean
Gareth               Admin               8 Smelly
Mark                 Manager             9 Clean
Mark                 Manager            10 Wash

10 rows selected.

SQL>
```

6.2.2 Create a right outer join to give details of the staff service with its service ID, also describe the position of the staff.

```
/* Query 2

List the staff who are assigned a service schedule, also with position
and service id

*/

SELECT s.staffid, s.fName, s.lName, se.serviceNo

FROM Staff s LEFT OUTER JOIN Service se

ON s.staffid= se.staffid;
```



```
C:\Windows\system32\cmd.exe - sqlplus
---------- -------------------- -------------------- ----------
       201 Gavin                Conor                         301
       201 Gavin                Conor                         302
       205 Gareth               Burke                         303
       207 Mark                 Simpson                       304
       201 Gavin                Conor                         305
       201 Gavin                Conor                         306
       205 Gareth               Burke                         307
       207 Mark                 Simpson                       308
       205 Gareth               Burke                         309
       207 Mark                 Simpson                       310
       202 Brendan              Murphy

   STAFFID FNAME                LNAME                SERVICENO
---------- -------------------- -------------------- ----------
       204 Fiona                Blake
       206 Neil                 Green
       203 Gerry                Bowen
       210 Eric                 Shups
       209 Ramnik               Singh
       208 Ashley               Spencer

17 rows selected.

SQL> _
```

6.2.3 Create a right outer join to display all the rooms giving details about their flat and its address.

```
/*Query 1

List the details of the rooms in the respective flat with address

*/

SELECT r.roomNo, r.rent, f.flatNo, f.faddr

FROM Room r RIGHT OUTER JOIN Flat f

ON r.flatNo= f.flatNo;
```

```
C:\Windows\system32\cmd.exe - sqlplus
  2  FROM Room r RIGHT OUTER JOIN Flat f
  3  ON r.flatNo= f.flatNo;

    ROOMNO       RENT     FLATNO FADDR
---------- ---------- ---------- --------------------
        21        500          2 3,Mount Merrion
        22        500          2 3,Mount Merrion
        11        600          1 2,Mount Merrion
        12        600          5 2,Mount Merrion
        13        600          1 2,Mount Merrion
        23        500          4 3,Mount Merrion
        31        450          9 5,Mount Merrion
        32        450         10 5,Mount Merrion
        33        450          8 5,Mount Merrion
        24        500          4 3,Mount Merrion
                               3 3,Mount Merrion

    ROOMNO       RENT     FLATNO FADDR
---------- ---------- ---------- --------------------
                               6 2,Mount Merrion
                               7 2,Mount Merrion

13 rows selected.

SQL> _
```

6.2.4 Create a right outer join to list staff members and the list of services given by them (if any).

```
/*Query 2

List the services performed by the staff members

Include the name and position of the staff

*/

SELECT se.serviceNo, s.fname, s.lname, s.position

FROM Service se RIGHT OUTER JOIN Staff s

-- reports the staffids from both tables

ON se.staffid= s.staffid;
```

```
C:\Windows\system32\cmd.exe - sqlplus
---------- -------------------- -------------------- ----------
       301 Gavin                Conor                Manager
       302 Gavin                Conor                Manager
       303 Gareth               Burke                Admin
       304 Mark                 Simpson              Manager
       305 Gavin                Conor                Manager
       306 Gavin                Conor                Manager
       307 Gareth               Burke                Admin
       308 Mark                 Simpson              Manager
       309 Gareth               Burke                Admin
       310 Mark                 Simpson              Manager
           Brendan              Murphy               Accountant

 SERVICENO FNAME                LNAME                POSITION
---------- -------------------- -------------------- ----------
           Fiona                Blake                Doctor
           Neil                 Green                Cook
           Gerry                Bowen                Security
           Eric                 Shups                Cook
           Ramnik               Singh                Security
           Ashley               Spencer              Gardener

17 rows selected.

SQL>
```

6.2.5 Create a full outer join to list all the rooms along with their corresponding flats. Also, list all the flats whose rooms aren't assigned.

```
/*

Query 1

List the details of the rooms in the respective flat with address

Also, lists the flat whose room details are not taken

*/

SELECT r.roomNo, r.rent, f.flatNo, f.faddr

FROM Room r FULL OUTER JOIN Flat f

ON r.flatNo= f.flatNo;
```

```
C:\Windows\system32\cmd.exe - sqlplus
  2  FROM Room r FULL OUTER JOIN Flat f
  3  ON r.flatNo= f.flatNo;

    ROOMNO       RENT     FLATNO FADDR
---------- ---------- ---------- --------------------
        13        600          1 2,Mount Merrion
        11        600          1 2,Mount Merrion
        22        500          2 3,Mount Merrion
        21        500          2 3,Mount Merrion
                                3 3,Mount Merrion
        24        500          4 3,Mount Merrion
        23        500          4 3,Mount Merrion
        12        600          5 2,Mount Merrion
                                6 2,Mount Merrion
                                7 2,Mount Merrion
        33        450          8 5,Mount Merrion

    ROOMNO       RENT     FLATNO FADDR
---------- ---------- ---------- --------------------
        31        450          9 5,Mount Merrion
        32        450         10 5,Mount Merrion

13 rows selected.

SQL> _
```

6.2.6 List the details of all the lease and also give the room number and rent associated with the particular lease. Mention the student number as well.

```
/*

Query 2

Display the details of the lease (length and student no) along with
the

room associated with that lease

*/

SELECT l.leaseNo,l.length, l.studentNo, r.roomNo, r.rent

FROM Lease l FULL OUTER JOIN Room r

ON l.roomNo= r.roomNo;
```

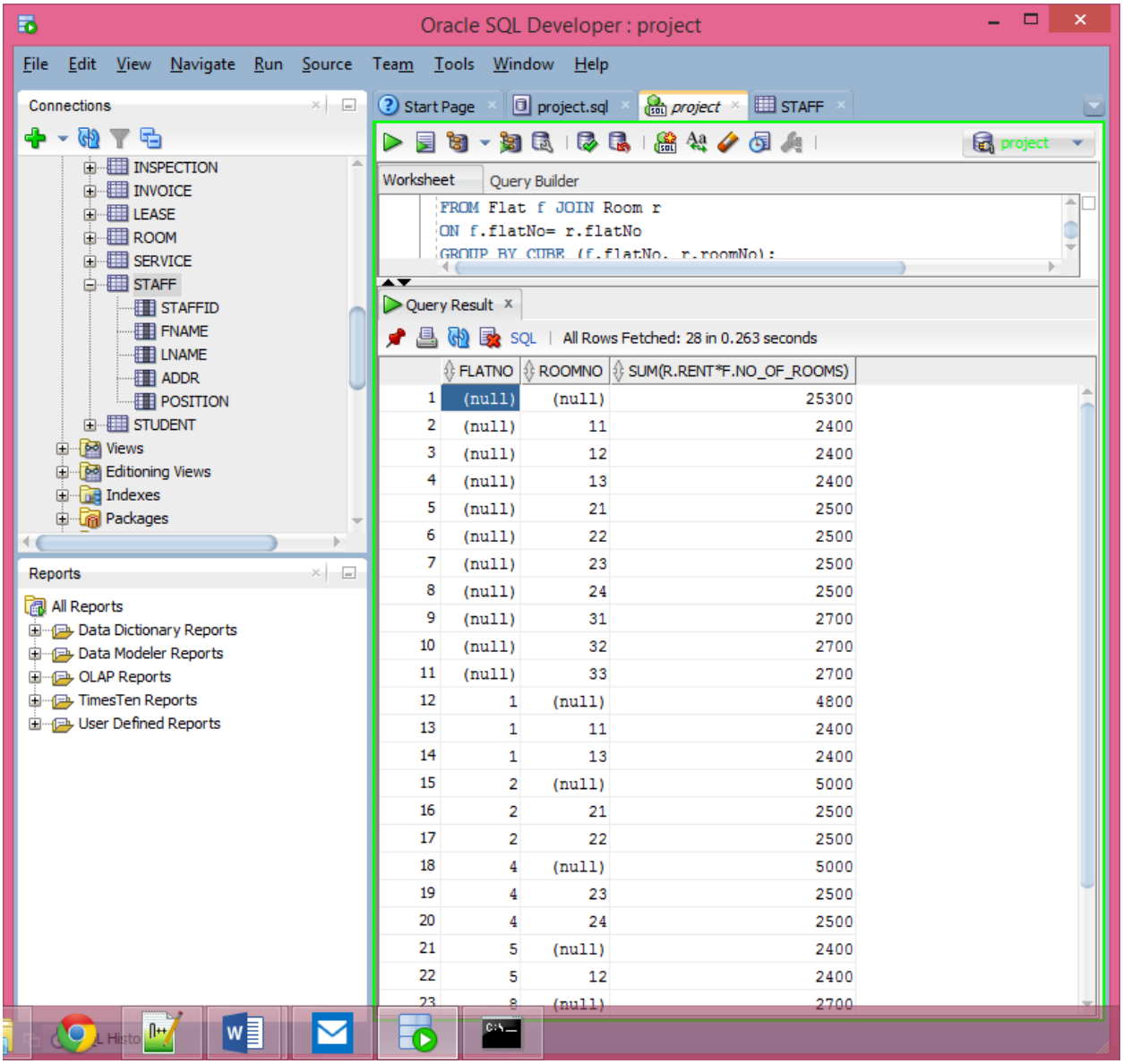
```
C:\Windows\system32\cmd.exe - sqlplus

        31        450          9 5,Mount Merrion
        32        450         10 5,Mount Merrion

13 rows selected.

SQL> SELECT l.leaseNo,l.length, l.studentNo, r.roomNo, r.rent
  2  FROM Lease l FULL OUTER JOIN Room r
  3  ON l.roomNo= r.roomNo;

   LEASENO     LENGTH  STUDENTNO     ROOMNO       RENT
---------- ---------- ---------- ---------- ----------
         2         60        101         21        500
         5         90        104         22        500
         1         30        100         11        600
         4         60        103         12        600
         7         90        106         13        600
         6         90        105         23        500
         3         30        102         31        450
         8        120        107         32        450
         9         30        108         33        450
        10         30        109         24        500

10 rows selected.

SQL> _
```

7 AGGREGATE FUNCTIONS

7.1 CUBE

7.1.1 Calculate the rent collection for a flat, giving room number and the total rent collected.

```
/* CUBE

Display the rent collected from individual flat

Calculate the rent collection and group by flat, then its respective
room, if given.

*/

SELECT f.flatNo, r.roomNo, SUM(r.rent*f.no_of_rooms)

FROM Flat f JOIN Room r

ON f.flatNo= r.flatNo

GROUP BY CUBE (f.flatNo, r.roomNo);

-- lists in order of flat, then rooms in it.

-- using SQL Developer to list all the results (the last 5 results are
not visible in the screenshot)
```



7.2 SUB QUERIES

7.2.1 What payment method was most used by the student during payment of rent due on invoice? Also, give the amount collected from that method.

```
/* Query 1-

Find the payment method that was used for maximum payment by a student

*/

SELECT payment_type, payment FROM Invoice WHERE

payment= (SELECT MAX(payment) FROM Invoice);
```

```
C:\Windows\system32\cmd.exe - sqlplus

SQL> SELECT payment_type, payment FROM Invoice WHERE
  2  payment= (SELECT MAX(payment) FROM Invoice);

PAYMENT_TY    PAYMENT
---------- ----------
Card             1800
Cheque           1800

SQL> _
```

7.2.2 Which student has the room lease for the longest period of time? Give her/his student number.

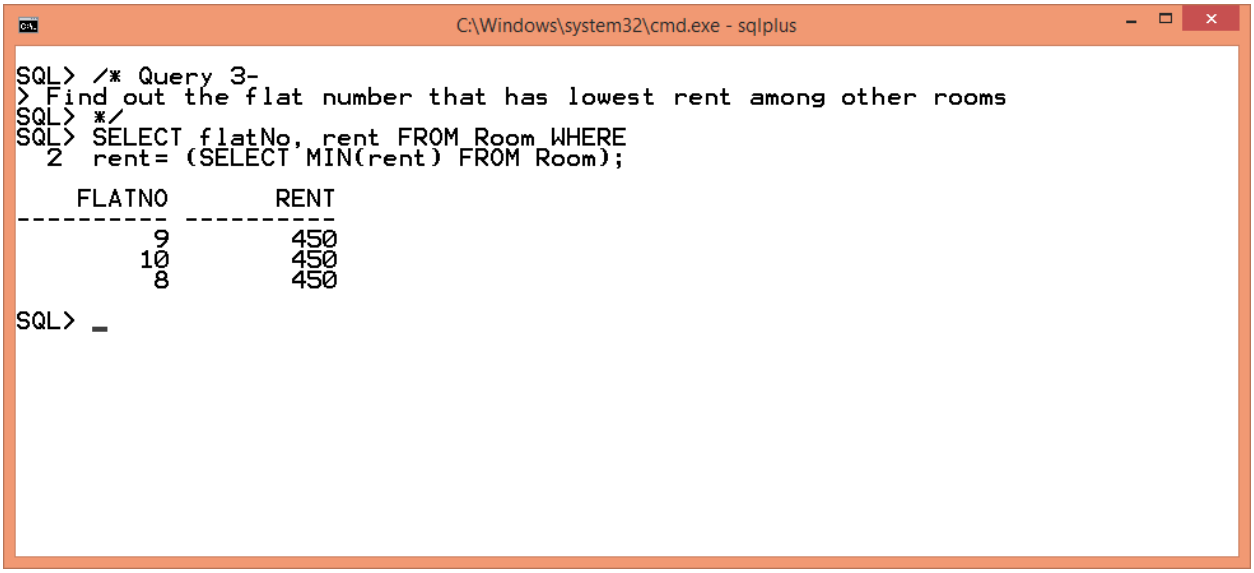
```
/* Query 2-

Find out the student number for whom the lease was let for the longest
period

*/

SELECT studentNo, roomNo FROM Lease WHERE

length= (SELECT MAX(length) FROM Lease);
```

```
C:\Windows\system32\cmd.exe - sqlplus

SQL> SELECT studentNo, roomNo FROM Lease WHERE
  2  length= (SELECT MAX(length) FROM Lease);

 STUDENTNO     ROOMNO
---------- ----------
       107         32

SQL>
```

7.2.3 Which flat has the cheapest rooms among others, give the number of flat with its rent of individual room in it.

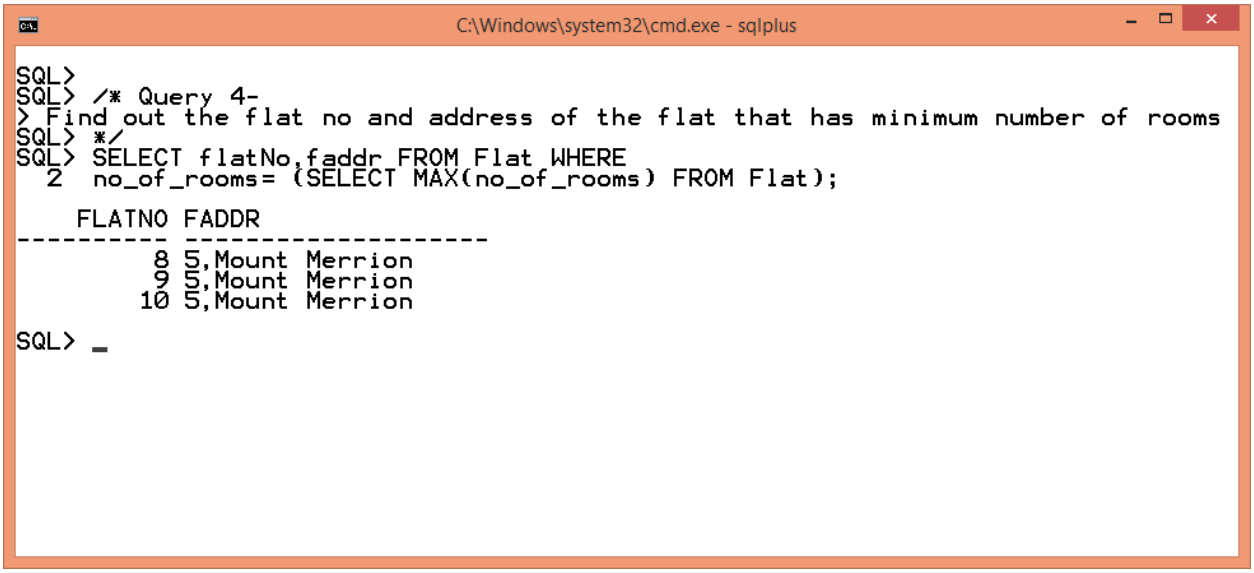
```
/* Query 3-

Find out the flat number that has lowest rent among other
```

```
*/

SELECT flatNo, rent FROM Room WHERE

rent= (SELECT MIN(rent) FROM Room);
```



7.2.4 Give the address of the flat along with its number that has least number of rooms in it.

```
/* Query 4-

Find out the flat no and address of the flat that has minimum number
of rooms

*/

SELECT flatNo,faddr FROM Flat WHERE

no_of_rooms= (SELECT MAX(no_of_rooms) FROM Flat);
```



8 STORED PROGRAMS

8.1 PROCEDURES

There are 5 procedures demonstrating use of CURSOR, SAVEPOINT & ROLLBACK in total, which are described under a single package. The questions for the procedures are given as follows, followed by the SQL statements in the end.

8.1.1 Create a procedure to add new rows to the Room table. Using the SAVEPOINT, demonstrate the functionality of ROLLBACK

8.1.2 Create a CURSOR to check if the rent of a room is greater than 600 and raise an EXCEPTION in that case. Further, check if the rent is greater than 500, raise another exception.

8.1.3 Insert additional rows in table Flat. Demonstrate the use of TRY/ CATCH statements by creating an ambiguous condition (like repeating primary key value). ROLLBACK in case of error.

8.1.4 Similar to the above question, create a procedure to demonstrate an ambiguous condition, without raising an exception. Use the ROLLBACK function if the procedure is unable to add rows into the table.

8.1.5 Insert values in table Staff. Demonstrating the use of SAVEPOINT and ROLLBACK, insert the duplicate value for a primary key constraint and ROLLBACK in that case to the savepoint created before.

```
SET AUTOCOMMIT OFF;

-- To demonstrate the rollback function

-- Creating new rows in Room table and rolling back to savepoint

CREATE OR REPLACE PACKAGE Rosemont1 AS

-- creating a package for 5 procedures

     PROCEDURE proc1;

     PROCEDURE proc2;

     PROCEDURE proc3;

     PROCEDURE proc4;

     PROCEDURE proc5;

END;

/
```

```
CREATE OR REPLACE PACKAGE BODY Rosemont1 AS
PROCEDURE proc1
-- Creating new rows in Room table and rolling back to savepoint
IS
BEGIN
INSERT INTO Room (roomNo, rent, flatNo) VALUES (34, 450, 2);
INSERT INTO Room (roomNo, rent, flatNo) VALUES (14, 600, 5);
SAVEPOINT sp1;
--savepoint created
INSERT INTO Room (roomNo, rent, flatNo) VALUES (15, 600, 8);
ROLLBACK TO sp1;
-- rolling back to savepoint
END proc1;
-- procedure 1 ends

-- Cursor to check whether the room rent is greater than 500,
-- raises an exception in case
PROCEDURE proc2
IS
BEGIN
DECLARE
     CURSOR curs1
     IS SELECT * FROM Room;

     v_room_row curs1%ROWTYPE;
     -- defining type of cursor
     RENT_EXP EXCEPTION;
     -- declaring variable for exception
     BEGIN
```

```
        OPEN curs1;
        -- initialising cursor
        FETCH curs1 INTO v_room_row;
        -- fetching data from the table row wise
        WHILE curs1%FOUND LOOP

            IF v_room_row.rent < 600 THEN
            -- condition 1, if rent is greater than 600
                RAISE_APPLICATION_ERROR(-30453, 'Rent for' ||
v_room_row.roomNo || ' is less than 600');
            -- it raises an exception to the screen
            ELSIF v_room_row.rent > 500 THEN
                -- if the rent exceeds 500
                RAISE RENT_EXP;
                -- another exception is raised
            END IF;

            FETCH curs1 INTO v_room_row;
            -- cursor is bought to the next location
        END LOOP;
        -- end of while loop
    CLOSE curs1;
        -- cursor is exited
    EXCEPTION

        WHEN RENT_EXP THEN
          -- throwing an exception
            DBMS_OUTPUT.PUT_LINE('Rent is greater than 500' ||
v_room_row.roomNo);
            -- complementing with a statement
            RAISE;
```

```
END;
-- end cursor
END proc2;
-- procedure ends



-- Using a try/ catch expression, creating a savepoint
-- Deliberate creating an error scenario
PROCEDURE proc3
IS
BEGIN
BEGIN TRY
     BEGIN TRANSACTION
     SAVEPOINT sp2;
     INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (11, '2,
Mount Merrion',5);
     INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (12, '5,
Mount Merrion',4);
     INSERT INTO Flat (flatNo, faddr, no_of_rooms) VALUES (12, '5,
Mount Merrion',4);
     -- repeating the primary key, to add exception
          COMMIT TRANSACTION;
END TRY;


BEGIN CATCH
     ROLLBACK TO sp2;
     -- rolling back to savepoint
     RAISE_APPLICATION_ERROR(-34545, 'FlatNo already exists');
END CATCH;
END proc3;
```

```
PROCEDURE proc4
IS
BEGIN
INSERT INTO Service (serviceNo, staffid) VALUES(311, 202);
INSERT INTO Service (serviceNo, staffid) VALUES(312, 203);
SAVEPOINT sp1;
--savepoint create
INSERT INTO Service (serviceNo, staffid) VALUES(311, 207);
-- putting ambigous values to the serviceNo
-- which is a primary key
ROLLBACK TO sp1;

END proc4;



-- Creating a
PROCEDURE proc5
IS
BEGIN
SAVEPOINT sp5;
--savepoint created
INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (211,
'Rich', 'Beg', 'C 105','Cook');
INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (213,
'Gerry', 'James', 'C 106','Security');
-- putting ambigous values to the serviceNo
INSERT INTO Staff (staffid, fname, lname, addr, position) VALUES (205,
'Rich', 'Beg', 'C 105','Cook');
-- which is a primary key
ROLLBACK TO sp5;
```

```
END proc5;


END;
--package body ends
/
```

8.2 FUNCTIONS

8.2.1 Create a function that checks for a name that isn't in the table Staff by creating a deliberate fail. Return an error line in that case.

```
CREATE OR REPLACE FUNCTION Func1
RETURN VARCHAR2
IS
DECLARE
     staff1 Staff%ROWTYPE;
     -- initialising variable from table
BEGIN
     SELECT * INTO staff1 FROM Staff WHERE lname='Potter';
     -- deliberate error case by using new lname value
EXCEPTION
     WHEN NO_DATA_FOUND THEN
     -- error when no data is found
          DBMS_OUTPUT.PUT_LINE('No records for this name');
     RETURN NULL;
END Func1;
/
```

8.2.2 Create a function to return the number of students that have any special requirements from the Student table.

```
-- A function that returns the number of students
-- who require special needs
CREATE OR REPLACE FUNCTION Func2(studentNo NUMBER)
RETURN NUMBER
IS
```

```
BEGIN
     RETURN COUNT (*) FROM STUDENT WHERE STUDENT.spl_req IS NOT NULL;
END Func2;
/
```

8.3 TRIGGERS

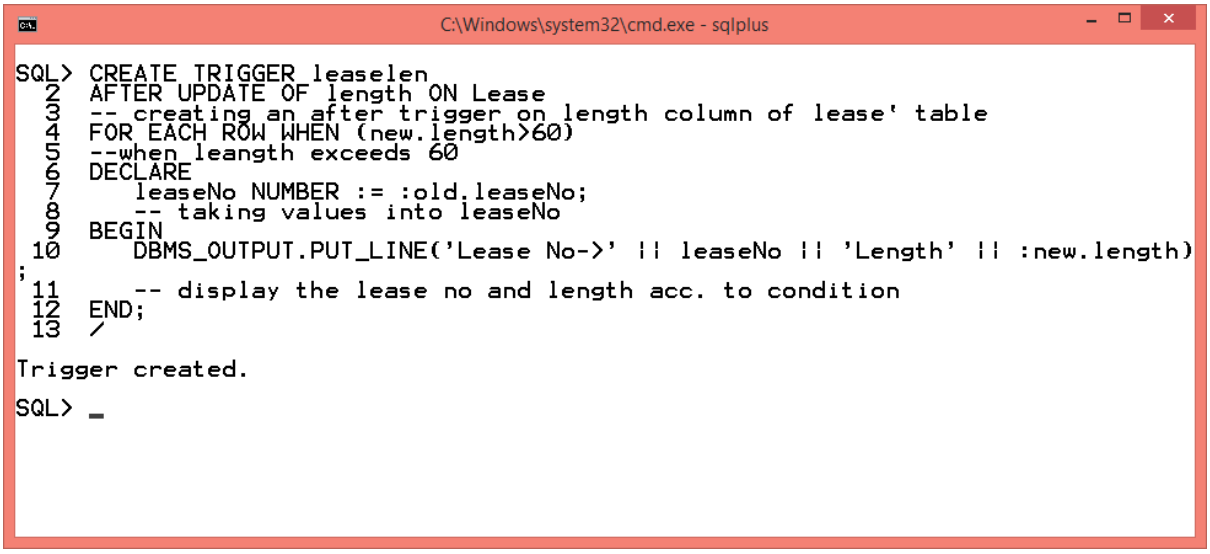
8.3.1 Create a before trigger to put the condition that a member of staff cannot inspect more than 10 flats.

```
CREATE TRIGGER StaffNotHandling
BEFORE INSERT OR UPDATE ON Inspection
FOR EACH ROW
DECLARE
vCount NUMBER;
BEGIN
     SELECT COUNT(*) INTO vCount
     FROM Inspection
     WHERE staffid=: new.staffid;
     IF vCount =10
     -- condition raised on staff
     raise_application_error(-20000,'Staff' || :new.staffid ||'already
inspected 6 flats');
     END IF;
END;
```

8.3.2 Create an after trigger for the lease table and update one of the record in that table. Give the output of the operation by displaying table after trigger is pulled.

```
-- Creating an After TRIGGER for table lease
CREATE TRIGGER leaselen
AFTER UPDATE OF length ON Lease
-- creating an after trigger on length column of lease` table
FOR EACH ROW WHEN (new.length>60)
--when leangth exceeds 60
DECLARE
```

```
      leaseNo NUMBER := :old.leaseNo;

      -- taking values into leaseNo

BEGIN

      DBMS_OUTPUT.PUT_LINE('Lease No->' || leaseNo || 'Length'
|| :new.length);

      -- display the lease no and length acc. to condition

END;

/

-- Checking trigger

UPDATE Lease SET length = 60

WHERE leaseNo = 3;

-- updating records

UPDATE Lease SET length = 90

WHERE leaseNo = 3;

-- updating same records with different length

SELECT * FROM Lease;
```



```
C:\Windows\system32\cmd.exe - sqlplus
SQL> -- updating records
SQL> UPDATE Lease SET length = 90
  2  WHERE leaseNo = 3;

1 row updated.

SQL> -- updating same records with different length
SQL> SELECT * FROM Lease;

   LEASENO     LENGTH  STUDENTNO     ROOMNO CHECKIN   CHECKOUT
---------- ---------- ---------- ---------- --------- ---------
         1         30        100         11 01-SEP-14 01-OCT-14
         2         60        101         21 01-SEP-14 01-NOV-14
         3         90        102         31 01-JAN-15 01-FEB-15
         4         60        103         12 01-JAN-15 01-MAR-15
         6         90        105         23 01-SEP-14 01-DEC-14
         7         90        106         13 01-SEP-14 01-DEC-14
         8        120        107         32 01-JAN-15 01-MAY-15
         9         30        108         33 01-SEP-14 01-OCT-14
        10         30        109         24 01-MAR-14 01-APR-14
         5         90        104         22 01-SEP-14 01-DEC-14

10 rows selected.

SQL>
```

8.3.3 Create a before trigger for the Guardian table to check if each row is related to one student only.

```
-- Creating a BEFORE TRIGGER to check the guardian is related to one
student only

CREATE TRIGGER GuardianChk

BEFORE INSERT OR UPDATE ON Guardian

FOR EACH ROW

DECLARE

vCount NUMBER;

BEGIN

      SELECT COUNT(*) INTO vCount

      FROM Guardian

      WHERE studentNo=: new.studentNo;

      IF vCount =1

      raise_application_error(-20000,'Student' || :new.staffid
||'already has one guardian');

      END IF;

END;

/
```

9 WEAKNESSES/ IMPROVEMENTS

1. The size for the name variables (`fname, lname`) in the Student table was assigned as 10. It created a problem while giving long names. Improvements can be made by altering table.
2. The Course table wasn't linked to any table, which rendered it as useless.
3. As per the design of the database, more robustness could be provided by adding roles to the database. This could be done when implementing the database in a real scenario by adding privileges to users.
4. More numerical data could have been added so as to foster efficient use of queries, like salary in Staff table.
5. Security of the database is not covered in this project.

10 REFERENCES

1. Database Administration, Oracle Database Online Documentation 11g Release 2 (11.2)
2. Oracle Data Types, http://docs.oracle.com/cd/B28359_01/server.111/b28318/datatype.htm#CNCPT413
3. Database Systems, A practical Approach to design, Implementation, and Management, Connolly T., Begg C., pp. 255- 267

-End-