**School:** Computer Science

**Institution:** University of Windsor

**Term:** Fall 2019

**Course:** Comp-3150 (03-60-315-1) : Database Management Systems

**Instructor:** Dr. C. I. Ezeife

**Assignment #**3 : Total: 50 marks

**Handed Out:Thurs. Oct. 24, 2019; Due Thurs Nov. 14, 2019**

**Objective of Assignment**: To test on knowledge and use of relational database query languages SQL and relational algebra for implementing relational databases.

**Scope**: Assignment covers materials from Chapters 6, 7 and 8 of book discussed in class.

**Electronic Assignment Submission:** Done through <http://blackboard.uwindsor.ca>

**Marking Sheme** : The mark for each of the questions is indicated beside each question.

**Academic Integrity Statement**: Remember to submit only work that is yours and include the following confidentiality agreement and statement at the beginning of your assignment.

**CONFIDENTIALITY AGREEMENT & STATEMENT OF HONESTY**

**I confirm that I will keep the content of this assignment/examination confidential.**

**I confirm that I have not received any unauthorized assistance in preparing for or doing this assignment/examination. I confirm knowing that a mark of 0 may be assigned for copied work.**

\_\_\_\_\_\_\_\_Andrea Bonato\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_Andrea Bonato\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Signature Student Name (please print)

\_\_104760390\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_2019-11-16\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student I.D. Number Date

**Marking Scheme : The mark for each question and sub question is shown with the question below. Place your solutions in tables where possible.**

**For office Use only**

|  |  |
| --- | --- |
| **Question** | **Mark** |
| **1** | **/15** |
| **2** | **/10** |
| **3** | **/5** |
| **4** | **/5** |
| **5** | **/5** |
| **6** | **/10** |
| **Total** | **/50** |

**CHAPTER 6: Basic SQL**

1. Given a database state of a version of the AIRLINE RESERVATION database shown in Figure 1.1, with schema shown in Figure 1.2,   
    (Total for que 1 is 15 marks)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Fig 1.1: An Example Database State of Parts of the Airline Database of Fig 5.8 of book  *AIRPORT*   |  |  |  |  | | --- | --- | --- | --- | | *Airport\_Code* | *Name* | *City* | *State* | | *YYZ* | *Pearson Airport* | *Toronto* | *ON* | | *YTZ* | *Island Airport* | *Toronto* | *ON* | | *YQT* | *Thunder Bay Airport* | *Thunder Bay* | *ON* | | *YQG* | *Windsor Airport* | *Windsor* | *ON* | | *YYC* | *Calgary Airport* | *Calgary* | *AB* |   *FLIGHT*   |  |  |  | | --- | --- | --- | | *Flight\_number* | *Airline* | *Weekdays* | | *WJ250* | *WestJet* | *3* | | *WJ261* | *WestJet* | *4* | | *AC275* | *Air Canada* | *2* | | *AC300* | *Air Canada* | *3* | | *AC320* | *Air Canada* | *2* | | *PA233* | *Porter Airline* | *5* | | *PA280* | *Porter Airline* | *4* |   *FLIGHT\_LEG*   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | *Flight\_number* | *Leg\_number* | *Departure\_airport\_code* | *Scheduled\_departure\_time* | *Arrival\_airport\_code* | *Scheduled\_arrival\_time* | | *WJ250* | *1* | *YQG* | *08.00* | *YYZ* | *09.00* | | *WJ250* | *2* | *YYZ* | *12.00* | *YYC* | *17.00* | | *AC275* | *1* | *YQT* | *10.00* | *YYZ* | *11.00* | | *AC275* | *2* | *YYZ* | *13.00* | *YQG* | *14.00* | | *PA233* | *1* | *YQG* | *16.00* | *YYZ* | *17.00* | | *PA280* | *1* | *YYZ* | *11.00* | *YQG* | *12.00* |   *LEG\_INSTANCE*   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | *Flight\_number* | *Leg\_number* | *FDate* | *Num\_available\_seats* | *Airplane\_id* | *Departure\_airport\_code* | *Departure\_time* | *Arrival\_airport\_code* | *Arrival\_time* | | *WJ250* | *1* | *15-JAN-17* | *200* | *Boeing\_777* | *YQG* | *8.00* | *YYZ* | *8.50* | | *WJ250* | *2* | *15-JAN-17* | *200* | *Boeing\_777* | *YYZ* | *12.30* | *YYC* | *16.50* | | *AC275* | *1* | *10-JAN-17* | *60* | *Airbus\_200* | *YQT* | *10.00* | *YYZ* | *11.00* | | *AC275* | *2* | *10-JAN-17* | *60* | *Airbus\_200* | *YYZ* | *13.00* | *YQG* | *14.00* | | *PA233* | *1* | *30-JAN-17* | *120* | *Boeing\_737* | *YQG* | *16.00* | *YYZ* | *17.00* | | *PA280* | *1* | *20-JAN-17* | *120* | *Boeing\_737* | *YYZ* | *11.00* | *YQG* | *12.00* |   *FARES*   |  |  |  |  | | --- | --- | --- | --- | | *Flight\_number* | *Fare\_code* | *Amount* | *Restrictions* | | *WJ250* | *F1* | *$250* | *None* | | *WJ250* | *F2* | *$350* | *None* | | *AC275* | *F1* | *$275* | *Non-refundable* | | *AC275* | *F2* | *$300* | *Non-refundable* | | *PA233* | *F3* | *$150* | *None* | | *PA280* | *F3* | *$150* | *None* |   *SEAT\_RESERVATION*   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | *Flight\_number* | *Leg\_number* | *FDate* | *Seat\_number* | *Customer\_name* | *Customer\_phone* | | *WJ250* | *1* | *15-JAN-17* | *20A* | *Mariane Mooer* | *519-253-3000* | | *WJ250* | *2* | *15-JAN-17* | *13D* | *Mariane Mooer* | *519-253-3000* | | *WJ250* | *1* | *15-JAN-17* | *6A* | *Tony Appa* | *226-253-4000* | | *WJ250* | *2* | *15-JAN-17* | *7D* | *Tony Appa* | *226-253-4000* | | *WJ250* | *1* | *15-JAN-17* | *6B* | *Karen Appa* | *226-253-4000* | | *WJ250* | *2* | *15-JAN-17* | *7C* | *Karen Appa* | *226-253-4000* | | *AC275* | *1* | *10-JAN-17* | *5C* | *Mark Black* | *226-212-5341* | | *AC275* | *2* | *10-JAN-17* | *14B* | *Mark Black* | *226-212-5341* | | *PA233* | *1* | *30-JAN-17* | *5A* | *Peter Opo* | *519-444-1234* | | *PA280* | *1* | *20-JAN-17* | *16D* | *Chris Natta* | *226-123-5555* | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Fig 1.2: Schema of the Airline Reservation database of Figure 1.1  AIRPORT   |  |  |  |  | | --- | --- | --- | --- | | Airport\_Code | Name | City | State |   FLIGHT   |  |  |  | | --- | --- | --- | | Flight\_number | Airline | Weekdays |   FLIGHT\_LEG   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Flight\_number | Leg\_number | Departure\_airport\_code | Scheduled\_departure\_time | Arrival\_airport\_code | Scheduled\_arrival\_time |   LEG\_INSTANCE   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Flight\_number | Leg\_number | FDate | Num\_available\_seats | Airplane\_id | Departure\_airport\_code | Departure\_time | Arrival\_airport\_code | Arrival\_time |   FARES   |  |  |  |  | | --- | --- | --- | --- | | Flight\_number | Fare\_code | Amount | Restrictions |   SEAT\_RESERVATION   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Flight\_number | Leg\_number | FDate | Seat\_number | Customer\_name | Customer\_phone | |

1. List all the referential integrity constraints that should hold on the database schema?

(2.5 marks)

1. Write appropriate SQL DDL statements to define the database with the integrity constraints and store in a text file called userid\_airlineschema.sql. Attach this file or also show it in your script file of (v) using more file.sql command before or after running sqlplus. Do the same for the files in (iii) and (iv). (2.5 marks)
2. To insert the data in the database tables, also write appropriate SQL DML instructions in a text file called userid\_airlinedata.sql. (2.5 marks)
3. To remove any inserted data and destroy all created tables in the airline reservation database, write appropriate SQL DML and DDL statements in a text file called userid\_airlinedroptable.sql to first delete all data in the tables and then drop the tables.

(2.5 marks)

1. Using Oracle Sqlplus, implement this database design by creating all the tables with the integrity constraints using the SQL DDL you defined in (ii) above. You can create all these 6 tables by running your .sql file at the SQL prompt with the command:

@userid\_airlineschema.sql. After creating your tables successfully, you load your data with the .sql file you created in (iii) above by running @userid\_airlinedata.sql. If there are errors and you need to correct them, you might want to delete the tuples and drop the tables first using the .sql file you created in (iv) above as with @userid\_airlinedroptable.sql before re-creating the schema and re-loading the data.Then, using a script file, show the contents of all 6 tables in the database by selecting \* from each of the tables. For example, select \* from airport; and saving on Unix script file called username\_assn3que1.txt. You can do this using the following sequence of Unix/Linux commands after you have created the database and inserted data. (5 marks)

**(Note: remember to create the entity tables with primary keys before the relationship tables that reference them through foreign key attributes. When inserting data, do the same. If you need to delete the data and tables at any time, go in the reverse order (that is, delete the tuples that reference a primary key attribute tuple in another table, before deleting the parent primary keyed tuple))**

>script username\_assn3que1.txt  
>sqlplus <username>  
>password   
sqlplus> select \* from AIRPORT; //repeat this instruction for each table  
sqlplus> exit //to exit sqlplus

exit // to exit and create script file

\*\*Now attach the saved log of your session that is in username\_assn3que1.txt with an inclusion in this script file of all the 3 .sql files in questions (ii), (iii) an (iv) or the attachment of those files as your solution.

**Solution 1 (i) (mark: 2.5)**

|  |
| --- |
| We will write a referential integrity constraint as R.A --> S (or R.(X) --> T) whenever attribute A (or the set of attributes X) of relation R form a foreign key that references the primary key of relation S (or T).  Some of the referential integrity constraints are as followed:   * In order for a LEG\_INSTANCE to exist, the corresponding SEAT\_RESERVATION must exist.   + SEAT\_RESERVATION 🡪 LEG\_INSTANCE   + SEAT\_RESERVATION(Flight\_number) 🡪 LEG\_INSTANCE(Flight\_number)   + SEAT\_RESERVATION(Leg\_number) 🡪 LEG\_INSTANCE(Leg\_number)   + SEAT\_RESERVATION(FDate) 🡪 LEG\_INSTANCE(FDate)   + FARES(Flight\_number) 🡪 FLIGHT(Flight\_number)   + FLIGHT\_LEG(Flight\_number) 🡪 FLIGHT(Flight\_number) * The next section explains how the airport makes reference to the FLIGHT\_LEG and LEG\_INSTANCE which means that FLIGHT\_LEG and LEG\_INSTANCE can exist without AIRPORT   + LEG\_INSTANCE(Departure\_airport\_code) 🡪 AIRPORT(Airport\_Code)   + LEG\_INSTANCE(Arrival\_airport\_code) 🡪 AIRPORT(Airport\_Code)   + FLIGHT\_LEG(Departure\_Airport\_code) 🡪 AIRPORT(Airport\_Code)   + FLIGHT\_LEG(Arrival\_airport\_code) 🡪 AIRPORT(Airport\_Code) |

**Solution 1 (ii): (mark: 2.5)**

|  |
| --- |
| One possible set of CREATE TABLE statements to define the database is given below given in the file userid\_airlineschema.sql as:  CREATE TABLE AIRPORT(  Airport\_Code varchar2(3),  Name varchar2(30),  City varchar2(20),  State varchar(3),  primary key(Airport\_Code)  );  CREATE table FLIGHT(  Flight\_number varchar2(5),  Airline varchar2(20),  Weekdays varchar2(20),  primary key(Flight\_number)  );  CREATE table FLIGHT\_LEG(  Flight\_number varchar2(5),  Leg\_number varchar2(2),  Departure\_airport\_code varchar2(3),  Scheduled\_departure\_time varchar2(20),  Arrival\_airport\_code varchar2(5),  Scheduled\_arrival\_time varchar2(20),  foreign key (Departure\_airport\_code) references AIRPORT(Airport\_Code),  foreign key (Arrival\_airport\_code) references AIRPORT(Airport\_Code),  foreign key (Flight\_number) references FLIGHT(Flight\_number),  primary key(Flight\_number, Leg\_number)  );  CREATE table LEG\_INSTANCE(  Flight\_number varchar2(5),  Leg\_number varchar2(2),  FDate varchar2(20),  Num\_available\_seats number(3),  Airplane\_id varchar2(15),  Departure\_airport\_code varchar2(3),  Departure\_time varchar2(5),  Arrival\_airport\_code varchar2(5),  Arrival\_time varchar2(5),  foreign key (Departure\_airport\_code) references AIRPORT(Airport\_Code),  foreign key (Arrival\_airport\_code) references AIRPORT(Airport\_Code),  primary key (Flight\_number, Leg\_number, FDate)  );  CREATE table FARES(  Flight\_number varchar2(5),  Fare\_code varchar2(2),  Amount varchar2(5),  Restrictions varchar2(15),  foreign key (Flight\_number) references FLIGHT(Flight\_number),  primary key (Flight\_number, Fare\_code)  );  CREATE table SEAT\_RESERVATION(  Flight\_number varchar2(5),  Leg\_number varchar2(2),  FDate varchar2(20),  Seat\_number varchar2(5),  Customer\_name varchar2(20),  Customer\_phone varchar(12),  foreign key (Flight\_number, Leg\_number, FDate) references LEG\_INSTANCE(Flight\_number, Leg\_number, FDate)  ); |

**Solution 1 (iii): (mark: 2.5)**

|  |
| --- |
| One possible set of INSERT INTO TABLE statements to load the database data is given below given in the file userid\_airlinedata.sql as:  INSERT into AIRPORT values('YYZ','Pearson Airport','Toronto','ON');  INSERT into AIRPORT values('YTZ','Island Airport','Toronto','ON');  INSERT into AIRPORT values('YQT','Thunder Bay Airport','Thunder Bay','ON');  INSERT into AIRPORT values('YQG','Windsor Airport','Windsor','ON');  INSERT into AIRPORT values('YYC','Calgary Airport','Calgary','AB');  COMMIT;  INSERT into FLIGHT values('WJ250','WestJet','3');  INSERT into FLIGHT values('WJ261','WestJet','4');  INSERT into FLIGHT values('AC275','Air Canada', '2');  INSERT into FLIGHT values('AC300','Air Canada','3');  INSERT into FLIGHT values('AC320','Air Canada','2');  INSERT into FLIGHT values('PA233','Porter Airline','5');  INSERT into FLIGHT values('PA280','Porter Airline','4');  COMMIT;  INSERT into FLIGHT\_LEG values('WJ250','1','YQG','08:00','YYZ','09:00');  INSERT into FLIGHT\_LEG values('WJ250','2','YYZ','12:00','YYC','17:00');  INSERT into FLIGHT\_LEG values('AC275','1','YQT','10:00','YYZ','11:00');  INSERT into FLIGHT\_LEG values('AC275','2','YYZ','13:00','YQG','14:00');  INSERT into FLIGHT\_LEG values('PA233','1','YQG','16:00','YYZ','17:00');  INSERT into FLIGHT\_LEG values('PA280','1','YYZ','11:00','YQG','12:00');  COMMIT;  INSERT into LEG\_INSTANCE values('WJ250','1','15-JAN-17',200,'Boeing\_777','YQG','8:00','YYZ','8:50');  INSERT into LEG\_INSTANCE values('WJ250','2','15-JAN-17',200,'Boeing\_777','YYZ','12:30','YYC','16:50');  INSERT into LEG\_INSTANCE values('AC275','1','10-JAN-17',60,'Airbus\_200','YQT','10:00','YYZ','11:00');  INSERT into LEG\_INSTANCE values('AC275','2','10-JAN-17',60,'Airbus\_200','YYZ','13:00','YQG','14:00');  INSERT into LEG\_INSTANCE values('PA233','1','30-JAN-17',120,'Boeing\_737','YQG','16:00','YYZ','17:00');  INSERT into LEG\_INSTANCE values('PA280','1','20-JAN-17',120,'Boeing\_737','YYZ','11:00','YQG','12:00');  COMMIT;  INSERT into FARES values('WJ250','F1','$250','None');  INSERT into FARES values('WJ250','F2','$350','None');  INSERT into FARES values('AC275','F1','$275','Non-refundable');  INSERT into FARES values('AC275','F2','$300','Non-refundable');  INSERT into FARES values('PA233','F3','$150','None');  INSERT into FARES values('PA280','F3','$150','None');  COMMIT;  INSERT into SEAT\_RESERVATION values('WJ250','1','15-JAN-17','20A','Mariane Mooer','519-253-3000');  INSERT into SEAT\_RESERVATION values('WJ250','2','15-JAN-17','13D','Mariane Mooer','519-253-3000');  INSERT into SEAT\_RESERVATION values('WJ250', '1','15-JAN-17','6A','Tony Appa','226-253-4000');  INSERT into SEAT\_RESERVATION values('WJ250','2','15-JAN-17','7D','Tony Appa','226-253-4000');  INSERT into SEAT\_RESERVATION values('WJ250','1','15-JAN-17','6B','Karen Appa','226-253-4000');  INSERT into SEAT\_RESERVATION values('WJ250','2','15-JAN-17','7C','Karen Appa','226-253-4000');  INSERT into SEAT\_RESERVATION values('AC275','1','10-JAN-17','5C','Mark Black','226-212-5341');  INSERT into SEAT\_RESERVATION values('AC275','2','10-JAN-17','14B','Mark Black','226-212-5341');  INSERT into SEAT\_RESERVATION values('PA233','1','30-JAN-17','5A','Peter Opo','519-444-1234');  INSERT into SEAT\_RESERVATION values('PA280','1','20-JAN-17','16D','Chris Natta','226-123-5555');  COMMIT; |

**Solution 1 (iv): (mark: 2.5)**

|  |
| --- |
| One possible set of DELETE FROM Tablename and DROP TABLE statements to first delete data from the tables and then, destroy the database tables is given below in the file userid\_airlinedroptable.sql as:    delete from seat\_reservation;  delete from fares;  delete from leg\_instance;  delete from flight\_leg;  delete from flight;  delete from airport;  commit;  drop table seat\_reservation;  drop table fares;  drop table leg\_instance;  drop table flight\_leg;  drop table flight;  drop table airport;  commit; |

1 (v). (5 marks) for the Unix script file showing correct interaction with Oracle Sqlplus creating and loading data in these 6 tables. Note that you need to first use a text editor like pico or Gedit to type your SQL instructions into an SQL script file with extension .sql so you can run it in Sqlplus or you need to type each of those instructions each time in Sqlplus. To execute the SQL instructions in your .sql file, you need to load and run that file in Sqlplus with the command: @file.sql.

**2.** Specify the following 5 queries in SQL on the airline database schema of Figure 1.1.

(Total for que 2 is 10 marks)

i. List all your 5 queries in the table below first in SQL. (5 marks)

ii. Implement the answering of your 5 queries in 2(i) using Sqlplus and the same database you created in question 1, providing your execution and answers to this questions in a script file called username\_assn3que2.txt. (5 marks)

(a) Retrieve the names and flight numbers of all customers who have taken a flight departing from Windsor.

(b) Retrieve the flight number and flight legs of all flights that have more than one leg.

(c) For each flight, retrieve the flight number, airline, flight date, and number of customers in the flight.

(d) Retrieve the name, seat number, and flight number of each customer in all Air Canada flights. (e) Retrieve the names and flight numbers, departure and arrival airports of all customers who did not fly in a flight that is more than the first leg.

**Solution 2 (i):** Queries(5 marks) and 2(ii) Results (5 marks)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| (a) Retrieve the names and flight numbers of all customers who have taken a flight departing from Windsor.  SQL Query:  SELECT SEAT\_RESERVATION.Customer\_name, SEAT\_RESERVATION.Flight\_number, LEG\_INSTANCE.Departure\_airport\_code  FROM SEAT\_RESERVATION, LEG\_INSTANCE  WHERE LEG\_INSTANCE.Departure\_airport\_code = 'YQG' AND LEG\_INSTANCE.Flight\_number = SEAT\_RESERVATION.Flight\_number AND LEG\_INSTANCE.Leg\_number = SEAT\_RESERVATION.Leg\_number;  Result of SQL Query (a):  -Retrieved result follows:   |  |  |  | | --- | --- | --- | | CUSTOMER\_NAME | FLIGHT | DEPARTURE\_AIRPORT\_CODE | | Mariane Mooer | WJ250 | YQG | | Tony Appa | WJ250 | YQG | | Karen Appa | WJ250 | YQG | | Peter Opo | PA233 | YQG |   (b) Retrieve the flight number and flight legs of all flights that have more than one leg.  SQL Query b:  SELECT FLIGHT\_LEG.Flight\_number, FLIGHT\_LEG.Leg\_number  FROM FLIGHT\_LEG  WHERE FLIGHT\_LEG.Leg\_number > 1;  Result of query b given below:   |  |  | | --- | --- | | FLIGHT | LEG\_NUMBER | | AC275 | 2 | | WJ250 | 2 |   (c ) For each flight, retrieve the flight number, airline, flight date, and number of customers in the flight.  SQL Query c:  SELECT SEAT\_RESERVATION.Flight\_number, FLIGHT.Airline, SEAT\_RESERVATION.FDATE,  (SELECT COUNT(Customer\_name)  FROM SEAT\_RESERVATION  WHERE FLIGHT.Flight\_number = SEAT\_RESERVATION.Flight\_number) AS CustomerNum  FROM SEAT\_RESERVATION, FLIGHT  WHERE SEAT\_RESERVATION.Flight\_number = FLIGHT.Flight\_number;  -- Result of query c is given below:  FLIGH AIRLINE FDATE CUSTOMERNUM  ----- -------------------- -------------------- -----------  AC275 Air Canada 10-JAN-17 2  AC275 Air Canada 10-JAN-17 2  PA233 Porter Airline 30-JAN-17 1  PA280 Porter Airline 20-JAN-17 1  WJ250 WestJet 15-JAN-17 4  WJ250 WestJet 15-JAN-17 4  WJ250 WestJet 15-JAN-17 4  WJ250 WestJet 15-JAN-17 4  (d) Retrieve the name, seat number, and flight number of each customer in all Air Canada flights.  SQL for Query d:  SELECT SEAT\_RESERVATION.Flight\_number, SEAT\_RESERVATION.Seat\_number, SEAT\_RESERVATION.Customer\_name  FROM SEAT\_RESERVATION, FLIGHT  WHERE FLIGHT.Flight\_number = SEAT\_RESERVATION.Flight\_number AND FLIGHT.Airline = 'Air Canada';  Result for query d is:   |  |  |  | | --- | --- | --- | | FLIGHT | SEAT\_NUMBER | CUSTOMER\_NAME | | AC275 | 5C | Mark Black | | AC2275 | 14B | Mark Black |   **(e)** Retrieve the names and flight numbers, departure and arrival airports of all customers who did not fly in a flight that is more than the first leg.  (Note that only attributes in the outer query retrievable as results of the query and thus, a join of a copy of Leg\_instance table is used in the outer query to get the departure and arrival airport codes and the NOT exists is used in the nested query to avoid tuples that flew in a flight with more than one leg).  **SQL for query e:**  SELECT SEAT\_RESERVATION.Customer\_name, SEAT\_RESERVATION.Flight\_number, LEG\_INSTANCE.Departure\_airport\_code, LEG\_INSTANCE.Arrival\_airport\_code  FROM SEAT\_RESERVATION, LEG\_INSTANCE  WHERE SEAT\_RESERVATION.Flight\_number = LEG\_INSTANCE.Flight\_number AND LEG\_INSTANCE.Leg\_number = SEAT\_RESERVATION.Leg\_number AND LEG\_INSTANCE.Leg\_number > (SELECT MIN(Leg\_number) FROM LEG\_INSTANCE);  Query result e is  CUSTOMER\_NAME FLIGH DEP ARRIV  -------------------- ----- --- -----  Mariane Mooer WJ250 YYZ YYC  Tony Appa WJ250 YYZ YYC  Karen Appa WJ250 YYZ YYC  Mark Black AC275 YYZ YQG |

2 (ii). (5 marks) distributed as: 2.5 marks for the script file showing correct interaction with Oracle Sqlplus posing these queries; and 2.5 marks for correctly posing the queries and retrieving correct results.

**3.** Write four SQL update statements to do the following updates on the database schema shown in Figure 1.2. Show the affected tables after update through script file in sqlplus and in a script file created as before and named username\_assn3que3.txt. (5 marks)

(Total for que 3 is 5 marks)

(a) Insert a new airport <'YTT', ‘Tintin Airport’, ‘Tintin’, 'ON'> in the database.

(b) Change the Customer\_phone of customer ‘Tony Appa’ to 519-253-4001.

(c) Insert a new flight <'AC331,'Air Canada', 2>.

(d) Delete all reservation records for the customer whose name is 'Mariane Mooer '.

**Solution 3 (i):** (5 marks)

|  |
| --- |
| (a) Insert a new airport <'YTT', ‘Tintin Airport’, ‘Tintin’, 'ON'> in the database.  INSERT into AIRPORT values(‘YTT’,’Tintin Airport’, ‘Tintin’, ‘ON’);  (b) Change the Customer\_phone of customer ‘Tony Appa’ to 519-253-4001  UPDATE SEAT\_RESERVATION  SET Customer\_phone = ‘519-253-4001’  WHERE Customer\_name = ‘Tony Appa’;  (c) Insert a new flight <'AC331,'Air Canada', 2>.  INSERT into FLIGHT values(‘AC331’, ‘Air Canada’, ‘2’);  (d) Delete all reservation records for the customer whose name is 'Mariane Mooer '.  DELETE from SEAT\_RESERVATION  WHERE Customer\_name = ‘Mariane Mooer’; |

**CHAPTER 7: More SQL: Complex Queries, Triggers, Views, and Schema Modification**

**4. (i)** Write the following 2 queries in SQL on the database schema of Figure 1.2 using EXISTS or NOT EXISTS as appropriate.   
 (2.5 marks)   
(ii) Implement the answering of your 2 queries in 4(i) using Sqlplus and the same database you created in question 1, providing your execution and answers to this question in a script file called username\_assn3que4. (2.5 marks)   
 (Total for que 4 is 5 marks)

(a) Retrieve the customer names and flight number of all 2-leg flyers (customers who have flights with 2 legs in all their flights).  
(b) Retrieve the customer names and flight number of all customers who do not have any 2-leg flight in any of their flights.

**Solution 4 (i):** (2.5 marks)

|  |
| --- |
| (a) Retrieve the customer names and flight number of all 2-leg flyers (customers who have flights with 2 legs in all their flights).  SQL Query:  SELECT Customer\_name, Flight\_number  FROM SEAT\_RESERVATION  WHERE EXISTS (SELECT Leg\_number FROM LEG\_INSTANCE WHERE Leg\_number = 2 AND Leg\_number = SEAT\_RESERVATION.Leg\_number);  Result of Query a.  CUSTOMER\_NAME FLIGHT  -------------------- -----  Mark Black AC275  Karen Appa WJ250  Tony Appa WJ250  Mariane Mooer WJ250  (b) Retrieve the customer names and flight number of all customers who do not have any 2-leg flight in any of their flights.  SQL Query b:  SELECT Customer\_name, Flight\_number  FROM SEAT\_RESERVATION  WHERE NOT EXISTS (SELECT Leg\_number FROM LEG\_INSTANCE WHERE Leg\_number = 2 AND Leg\_number = SEAT\_RESERVATION.Leg\_number);  Result of Query b:  CUSTOMER\_NAME FLIGH  -------------------- -----  Chris Natta PA280  Peter Opo PA233  Mark Black AC275  Karen Appa WJ250  Tony Appa WJ250  Mariane Mooer WJ250 |

4 (ii). (2.5 marks) distributed as: 0.5 marks for the script file showing correct interaction with Oracle Sqlplus posing these queries; and 2 marks for correctly posing the queries and retrieving correct results.

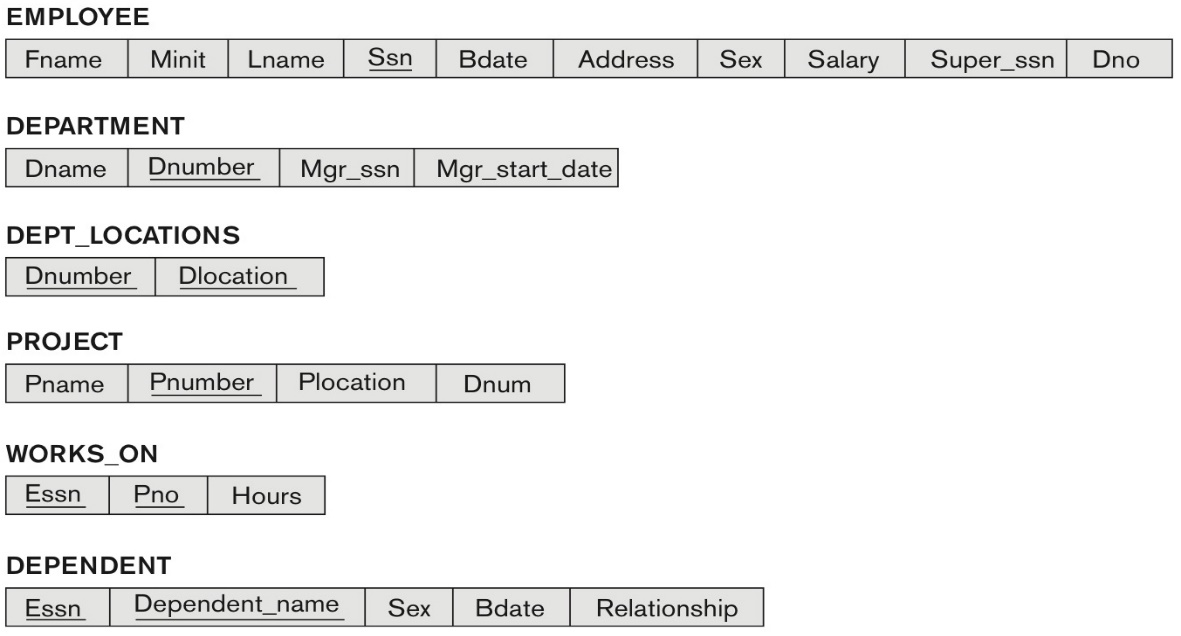
**5**. In SQL, specify the following 3 queries on the COMPANY database of Figure 5.5 using the concept of nested queries and the concepts described in chapter 7. (Total for que 5 is 5 marks)

a. Retrieve the names of all employees who work in the department that has the employee with the highest salary among all employees.

b. Retrieve the names of all employees whose supervisor’s supervisor has '333445555' for Ssn.

c. Retrieve the names of employees who make at least $15,000 more than the employee who is paid the least in the company.

**Figure 5.5** Schema diagram for the COMPANY relational database schema.

****

**Solution 5: (5 marks)**

|  |
| --- |
| 1. SQL query   SELECT Fname, Lname  FROM EMPLOYEE  WHERE EMPLOYEE.Dno = (SELECT Dno FROM EMPLOYEE WHERE EMPLOYEE.Salary = Max(EMPLOYEE.Salary))   1. SQL query   SELECT Fname, Lname  FROM EMPLOYEE  WHERE Super\_ssn = ‘3334455555’;   1. SQL query   SELECT Fname, Lname  FROM EMPLOYEE  WHERE EMPLOYEE.Salary > SUM(MIN(EMPLOYEE.Salary) + 15000); |

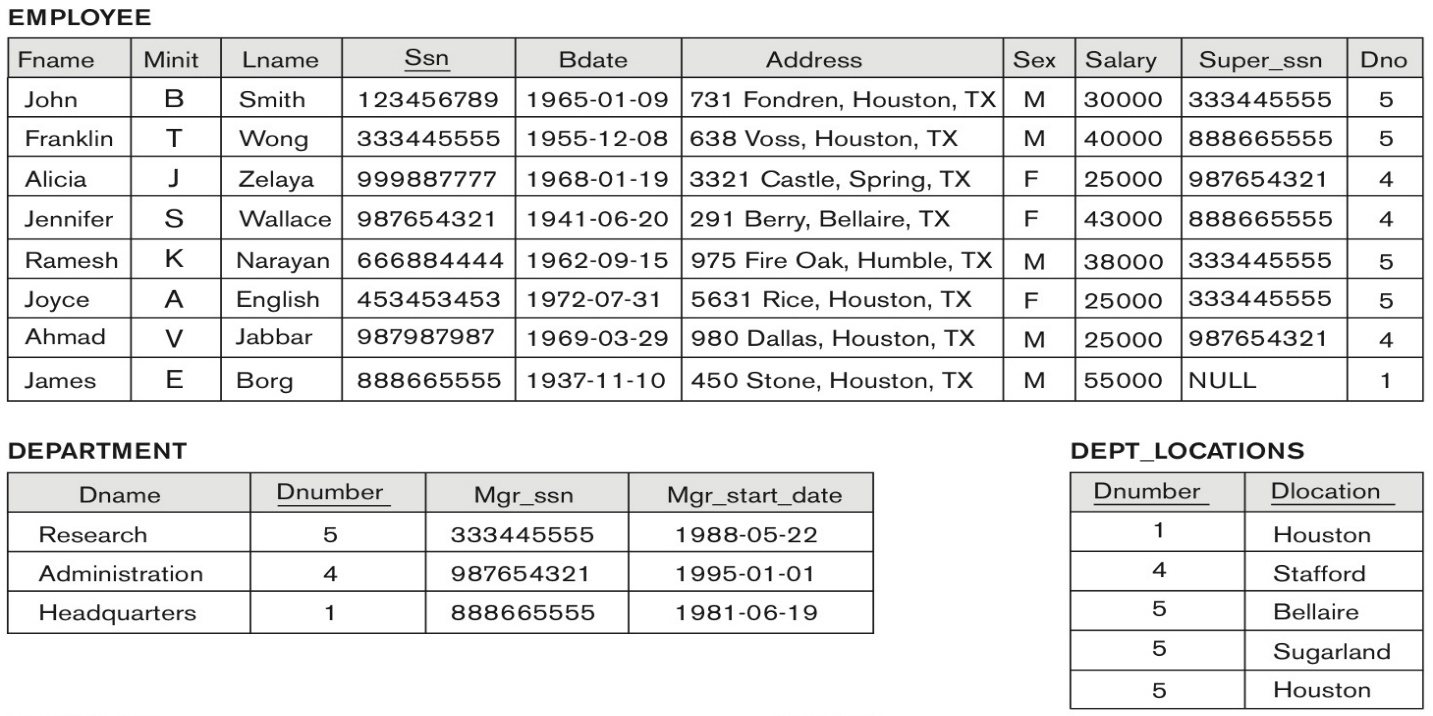
**CHAPTER 8: THE RELATIONAL ALGEBRA AND RELATIONAL CALCULUS**

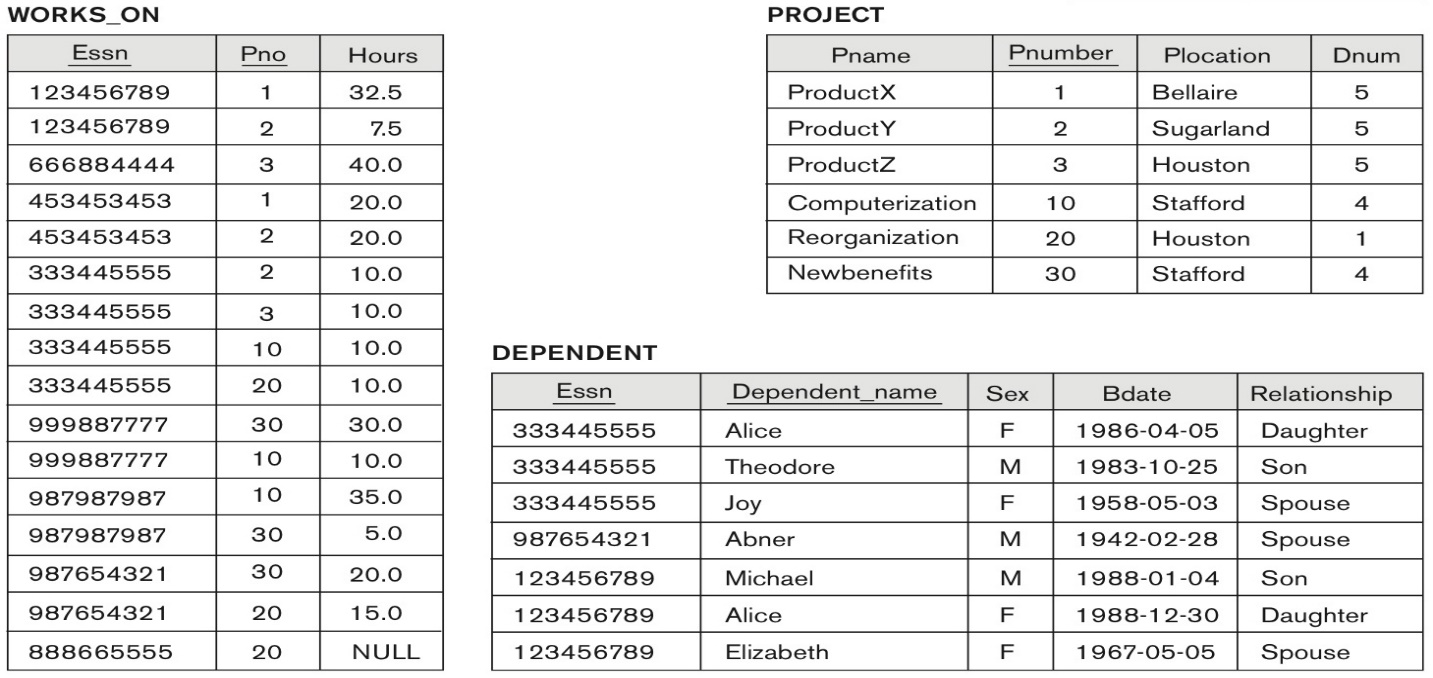
**6.** Specify the following 5 queries on the COMPANY relational database schema shown in Figure 5.5, using the relational operators discussed in chapter 8. Also show the result of each query as it would apply to the database state of Figure 5.6.

Some symbols for solving queries you may copy and reuse are: π, σ , ρ

(Total for que 6 is 10 marks)

**Figure 5.6** One possible database state for the COMPANY relational database schema.





(i) List the names of employees who have a dependent with the same first name as themselves.

(ii) Find the names of employees that are directly supervised by 'James Borg'.

(iii) For each project, list the project name and the total hours per week (by all employees) spent on that project.

(iv) Retrieve the names of employees who work on every project.

(v) Retrieve the maximum salary of all female employees.

Some symbols for solving queries you may copy and reuse are: π, σ , ρ

Solution 6: (10 marks)

|  |
| --- |
| In the relational algebra, as in other languages, it is possible to specify the same query in multiple ways. We give one possible solution for each query.   1. Query   πFname, Dependent\_name σEMPLOYEE.Fname =DEPENDENT.Dependent\_name(EMPLOYEE X DEPENDENT)  Result of query i)  Fname Dependent\_name  --------- ------------------------  Empty  (ii))Query:  πFname σ EMPLOYEE.Super\_ssn = EMPLOYEE.Ssn AND EMPLOYEE.Fname = ‘James’ AND EMPLOYEE.Lname = ‘Borg’  (EMPLOYEE)  Result of query ii:  Fname  ----------  Jennifer  iii) Query  PROJHOURS <- (π Pno, Hours (WORKS\_ON))  AVG <- (Pno  SUM Hours(PROJHOURS))  PROJ <- (π Pname, Pnumber (PROJECT))  ALLPROJ <- ( AVG |X| Pnumber = Pno PROJ)  RESULT <- (π Pname, Hours (ALLPROJ))  Result of query iii:  Pname Hours  ---------- ---------  ProductX 52.5  ProductY 17.5  ProductZ 40  Computerization 55  Reorganization 25  Newbenefits 55    (iv) Query  WORKS <- (π ESSN, Pno (WORKS\_ON))  PROJ <- (π Pnumber(PROJECT))  ALLPROJ <- (WORKS ÷ PROJ)  EMPS <- (π Fname, Minit, Lname (EMPLOYEE))  EMPSALL <- (EMPS |X| Ssn = Essn ALLPROJ)  RESULT <- (π Fname, Minit, Lname (EMPSALL))  Result of query iv:  Fname  ----------  Empty  No person works on all projects  (v) Query  FEMALES <- (σ Sex = ‘F’ (EMPLOYEE))  RESULT <- (MAXIMUM Salary (FEMALES))  Result of query v:  Salary  --------  43000 |