**COEN 178 Intro to Database Systems Winter2018**

**Lab 2 (100 pts)**

**Objectives:**

Learn writing SQL queries with

* Orderby,
* aggregate functions
* Joining tables,

**Note**: In this lab, you will create a few new tables in addition to the Employee data file you created and loaded last time.

**Capturing the session and output:**

You should capture the session (queries and output) into a file called, **lab2\_session.lst**.

**Capturing and Recording your Session**

One way to capture your SQL session is to use the spool command that SQLPLUS provides to save query results to a file. At the SQL> prompt, type spool <filename>. For example,

SQL> **spool foo;**

This will create a foo.lst file in the current directory and will capture all input and output during your session at SQLPLUS (until you exit SQL).

You can terminate the capture with the command, **spool off;**

You can submit the foo.lst file for grading.

Oracle SQL Datatypes, Reference: http://www.ss64.com/orasyntax/datatypes.html

**PART 1 (20 pts)**

In this part, you will create a few tables and load them with given values and values of your choice.

**Creating Tables**

Assume that a customer can schedule a variety of home delivery services (grocery, movies, books etc).

Create the following tables based on the description given below **(primary keys are underlined**):

a) **Customer**: **custid** (a string of at most 5 characters), **firstname** (a string of at most 10 characters) **lastname** (a string of at most 15 characters), **city** (a string of at most, 10 characters). Primary key: custid.

b) **DeliveryService**: **serviceid** (a string of at most, 10 characters), **item** (a string of 15 characters), **location** (a string of at most 15 characters), **servicefee** (a number with two points of precision after decimal point). Primary key: serviced.

c) **Schedule**: **service id, custid, day** (a string of up to 2 char and cannot be null). We will define a rule/constraint for the day attribute, that the day has to be Mon (‘m’) through Fri (‘f’). The rule is defined using a **CHECK** clause and **IN** to indicate the set of values that are allowed. Use **‘m’,’t’,’w’,’r’,’f’** to represent the day values.

See the example table below and use the same approach to define the rule for day in Schedule table.

**Example table** (this is an example to show the usage of CHECK. Do **NOT** create this table):

Create table Trip (id varchar(5) PRIMARY KEY,

source varchar(15), dest varchar(15)

CHECK (dest in ('NY','Paris','London')));

Foreign key: **custid** referencing the table Customer.

Foreign key: **serviceid** referencing the table DeliveryService.

**Inserting Tuples**

Add the following tuples into the tables (use a script file – i.e., part1.sql – to add the data).

**Customer:**

'c1', 'John', 'Smith','SJ'

'c2', 'Mary', 'Jones','SFO'

'a1', 'Vincent ','Chen','SJ';

'a12', 'Greg ', 'King','SJ';

'c7', 'James ', 'Bond','LA';

'x10', 'Susan ','Blogg','SFO';

***Add a few more tuples of your choice.***

**NOTE:** be aware of the single quote (‘) character. It may look right, but there are variations that will not be accepted by SQLPLUS. You may need to retype them in your file editor.

**DeliveryService:**

'dsg1','grocery','SJ',25.0

'dsb1','books','SJ',10.0

'dsm2','movies','LA',10.0

'dby3','babygoods','SFO',15.0

'dsg2','grocery','SFO',20.0

'dg5','greengoods','SFO',30.0

***Add a few more tuples of your choice.***

**Schedule**

'dsg1','c1','m'

'dsg1','a12','w'

'dby3','x10','f'

'dg5','c1','r'

'dg5','c1','t'

'dg5','c32','t'

**Q: Did you successfully insert all the above tuples? If not, explain the reason for error. (5 pts)**

**A: I was able to add all but the last tuple due to the fact that c32 is not in the customer list**

Now, try to insert the following tuple into Schedule table.

'dsg2','c1','s'

**Q: Did you successfully insert all the above tuples? If not, explain the reason for error. (5 pts)**

**A: I did not add the above tuple because dsg2 is not a serviceid and s is not a valid day**

***Add a few more tuples of your choice****.*

**Part 2 (50 pts)**

In this part, you will execute SQL commands from a script file.

**Step 1**: Create a folder structure called **COEN178/labs/lab2**.

**Step 2**: Create a text file called lab2\_**queries1.sql.** This file will contain the SQL queries that you want to execute.

**Write the SQL for the following queries (5 pts each) *(You are free to refer to the class notes and examples posted).***

Show all the data in the tables, **Customer,** **DeliveryService** and **Schedule.**

1. Show the data in the Customer table as follows:
   * 1. custid, fullname, city, where fullname is the result of concatenating First name and last (‘**||**’ is the concatenation operator). Rename the column name as the fullname.
2. Show data in the Customer, sorted by Customer last name (use ***order by***).
3. Show data in the Schedule table, sorted by service id and then by customer id in descending order (use ***order by***).
4. Show service ids of delivery services that are not in the schedule table (think **set difference**).
5. The following query is given to show the names of customers who ordered a delivery service on Monday (‘M’). Will it work? If not, fix it and show the query and results.

**Select name from customer, schedule where day = 'M';**

1. Show the last names of the customers that are scheduled delivery services. (What tables is the data coming from?)
2. Show the highest servicefee in Delivery Services (think of the aggregate function, max) renaming the result as highest\_Servicefee.
3. Show the number of delivery services scheduled by day (think of aggregate function, **count()** and **group by (day)).**
4. The incomplete query below is given to show pairs of customer ids from the same city. Complete it.

Select A.custid,B.custid,A.city

from Customer A, Customer B

where A.city = B.city;

**Q: Will it work? If not fix it and run it again. Show the result.**

**A: It shows duplicates**

1. Write a query to show the customers (who scheduled delivery services) where the customer city and location of the delivery services are in the same city.

**Do the following two queries against staff\_2010 table that you have created and loaded in lab1.**

1. Write a query to show the minimum salary and maximum salary of staff members in **staff\_2010** table. Display appropriate headings.

**Part 3 (25 pts)**

**Create two tables and load them with values given.**

**Creating Tables**

Create the tables, L\_EMP and L\_DEPT using the DDL statements below:

Create table L\_EMP (empNo Integer Primary Key, empname CHAR(10),deptId CHAR(5));

Create table L\_DEPT (deptId CHAR(5) Primary Key, deptname CHAR(10));

**Note**: We have not defined any foreign key constraint in these tables.

**Inserting Tuples**

Add the following tuples into the tables (use a script file to add the data).

insert into L\_EMP values(1,'smith','d1');

insert into L\_EMP values(2,'jones','d2');

insert into L\_EMP values(3,'wayne','d1');

insert into L\_EMP values(4,'moor','d3');

insert into L\_EMP values(5,'king','d1');

insert into L\_EMP values(6,'chen','d1');

insert into L\_EMP values(7,'winger','d3');

insert into L\_DEPT values('d1','Research');

insert into L\_DEPT values('d2','Devt');

insert into L\_DEPT values('d3','Testing');

insert into L\_DEPT values('d4','Advert');

Create a text file called lab2\_**queries2.sql and include the queries for the exercises below.**

Using a simple Select query, show all the rows in the tables, L\_EMP and L\_DEPT.

**Exercise 1 (15 pts)**

**Show the names of employees (from L\_EMP) who work in the dept with name, ‘testing’.**

**Approach 1:** You can use L\_EMP and L\_Dept in the “from” clause, use a bridge (L\_EMP.deptid = L\_Dept.deptid) and retrieve the dept.name.

**Approach 2**: In this approach, you will use **in** to connect a subquery to the outer query. The subquery should return the dept.id of depts. with *deptname = ‘testing’*, from table, L\_Dept. In the outer query, select those employee names with deptids in the set of deptids returned from the subquery. Run the query and check if you see the correct result.

1. If you did not see any results, it may be because the string comparison for ‘testing’ returned false. How do you ensure that you are comparing the strings correctly, whether the *deptname* in the table is upper or lower case? Make the correction to your query and run it again. Do you see the correct results now? (3 pts)

**Note:** We will use approach 3 in the next lab.

**Exercise 2 (10 pts)**

Write a query (using two approaches) to **show the dept names with no employees**.

Write the query using **iWrite a query to show the minimum salary and maximum salary of staff members in staff\_2010 table. Display appropriate headingsn** (or **not in**)