Project 1: Writing SQL Queries Using Oracle's SQL*Plus

(Due: April 15, 2021)

The following four tables will be used for this project. These tables are either directly from or modified from the relations obtained in Homework 3.

- Employees(eid, name, telephone#, email)
- Customers(cid, name, telephone#, visits_made, last_visit_date)
- Products(pid, name, qoh, qoh_threshold, regular_price, discnt_rate)
- Purchases(pur#, eid, pid, cid, pur_date, qty, unit_price, total, saving)

The meaning of each attribute is the same as that in the Requirements Document for RBMS.

The SQL statements for creating and populating these tables are provided in the file RBMSTablesScript.txt, which is uploaded to MyCourses. MyCourses also has instructions to use this file. This project is based on the tables created by the SQL statements in RBMSTablesScript.txt. **No changes are allowed to these tables for this project.**

There are 20 statements in this project. If not otherwise specified in a question, you are asked to write a single SQL query for each statement. Some questions may ask you to write more than one SQL query in different style. You are not allowed to create views or other (temporary) tables for this project. Inline views (i.e., select in the "from clause") are allowed. Your query should take into consideration that the tuples currently in the tables may change. In other words, your solution to each question must be correct no matter what valid tuples are in the tables.

To make the project more interesting, it is required that your queries do not return un-necessary duplicate results while keeping necessary duplicate results. For this project, identical results corresponding to different entities are considered to be necessary duplicates while identical results corresponding to the same entity are considered un-necessary duplicates. The following example illustrates necessary and un-necessary duplicate results. Consider query "select name from customers;". If two customers have the same name "Mike", then displaying "Mike" twice is considered necessary but displaying "Mike" more than twice is considered un-necessary.

The query (or queries) for each statement is worth 5 points. Keeping un-necessary duplicates or not keeping necessary duplicates will result in one point deduction.

It is very important that you understand each query statement correctly. If you have any doubt about the correct interpretation of a statement, please ask the instructor for clarification through the MyCourses discussion forum for this project.

I suggest that you first test each query individually and save each query in a different file (with extension .sql) in your harveyv account. After all queries have been tested to your satisfaction, you can run all the queries in a sequence and save the entire session in a spool file. Suppose you have saved your queries in files query1.sql, ..., query20.sql. Following the steps below to generate the spool file after you have logged into your Oracle.

```
SQL> set echo on
SQL> spool project1.txt
SQL> start query1
.....
SQL> start query20
SQL> spool off
```

For each of the 20 statements, your output needs to show both the query and the result of the query. This can be achieved by "set echo on" as shown above. Before you submit the file project1.txt, you need to add your name, section number, and the following statement "I have done this assignment completely on my own. I have not copied it, nor have I given my solution to anyone else. I understand that if I am involved in plagiarism or cheating I will have to sign an official form that I have cheated and that this form will be stored in my official university record. I also understand that I will receive a grade of 0 for the involved assignment and my grade will be reduced by one level (e.g., from A to A- or from B+ to B) for my first offense, and that I will receive a grade of "F" for the course for any additional offense of any kind." to the top of the file. Remember to (digitally) sign the statement before you submit the file.

The following are the 20 statements for this project:

1. Find the pid, name and quantity on hand of each product that has a discount rate between 10% and 20% (i.e., including 10% and 20%) and needs restocking if 5 such products are sold (i.e., (qoh – 5) < qoh_threshold).
2. **Find the name of each customer who visited the retail business either in August 2020 or January 2021. (Use to_char(pur_date, 'Month') and to_char(pur_date, 'YYYY') to extract month (e.g., August) and year (e.g., 2020) from pur_date, respectively. Alternatively, substr(pur_date, 4, 3) can be used to extract the 3 letters representing the month.)**
3. Find the name and telephone# of each customer who has purchased facemasks from the business.
4. Find the eids and names of those employees who have not sold any products, i.e., not involved in purchases. Write two queries for this statement with one having an uncorrelated subquery and another one having a correlated subquery.
5. Find all attributes of those purchases where a customer whose name starts with "K" purchased a product whose regular price is lower than 20 from an employee whose telephone's area code is 888.
6. Find the pur#, product name (with header "product name"), pur_date, total and saving of each purchase. Product name needs to be found from the Products table. pur_date must be displayed in a format as illustrated by the following example: March 30, 2021 Tuesday.

7. Find the eid of each employee whose telephone number has the same area code as that of at least one customer (the customer is not necessarily the employee's customer, i.e., the customer may not necessarily have made a purchase from the employee). Order the eids in ascending order.

8. Find the name of each customer who has visited the retail business at least twice but did not purchase any item whose unit price (i.e., the discount price) is higher than 100 in his/her most recent visit to the retail business.

9. Find the name of each employee who has not sold any product whose regular price is $150 or higher. Write two SQL queries for this statement, one uses “not in” and the other one uses “not exists”.

10. Find the cid and name of each customer who has purchased all of the products that are also purchased by customer c006. Customer c006 herself should be included in the result.

11. Find the pid and name of each product that has a regular price over $150 and has been sold by all employees whose telephone # has an area code 666.

12. Find the eid and name of each employee who has made sale to all the customers who have purchased either camera or chair.

13. Find the name of each customer who has made at least one purchase that has the highest total cost among all individual purchases. Note that it is possible for multiple purchases to have the same highest total cost.

14. Find the eid of the employee that has the highest total sale in terms of absolute dollar amount. Also show this dollar amount with header “total sale amount”. It is possible that multiple employees have the same highest total sale amount.

15. For each month in chronological order (header: “Month”), output the total sale (header: “Total Sale”) of the month (which is the sum of all total for the month). Under “Month”, show both month and year in the format as illustrated by “2020/08” for August 2020.

16. Find the pur# and saving of the purchases that have the three largest savings (in absolute values) among all individual purchases.

17. Same as Question 16 but the following differences: (1) The saving attribute in Purchases cannot be used. This means that you need to compute the saving using information from the Products table and the Purchases table. (2) The headers of the two output columns are pur# and savings, respectively.

18. Find the pur# of each such purchase whose total cost is greater than or equal to each of the total costs of those purchases placed by customer c006. Write two SQL queries for this question, one uses an aggregate function and the other one does not use any aggregate function.

19. Find the cids and names of the top-3 customers who have spent the most amounts of money at the retail business. Also show the total amount of money spent (header: “total amount spent”) by each of these customers. If more than one customer is tied at the third place, showing anyone (just one) of them is acceptable.

20. For each employee, find their eid and name as well as the number of different customers (header: “number of customers served”) the employee has served (i.e., made a sale to). If an employee has not made any sale, number 0 should be shown for the employee in the result.

Q1) select name,pid,qoh,qoh_threshold from products where discnt_rate >=.1 and discnt_rate <=.2 and (qoh-5)<QOH_THRESHOLD

Do I need to return the products which have qoh-5<threhhold

Q2) select distinct name, to_char(pur_date,'YYYY') as myyear from customers join purchases on purchases.CID=customers.cid where (to_char(purchases.pur_date,'Month')='August ' and to_char(purchases.pur_date,'YYYY')='2020') or (to_char(purchases.pur_date,'Month')='January ' and to_char(purchases.pur_date,'YYYY')='2021');

select name from customers where cid in(select distinct customers.cid from customers join purchases on purchases.CID=customers.cid where (to_char (purchases.pur_date, 'Month') ='August ' and to_char(purchases.pur_date,'YYYY')='2020') or (to_char(purchases.pur_date, 'Month') ='January ' and to_char(purchases.pur_date,'YYYY')='2021'));

select distinct name, to_char(LAST_VISIT_DATE,'YYYY') as myyear from customers where (to_char(LAST_VISIT_DATE,'Month')='August ' and to_char(LAST_VISIT_DATE,'YYYY')='2020') or

(to_char(LAST_VISIT_DATE,'Month')='January ' and to_char(LAST_VISIT_DATE,'YYYY')='2021');

Q3) select name,telephone# from customers where cid in(select cid from purchases join products on products.pid=purchases.pid where products.name='facemask');

Q4)uncorrelated: select eid, name from employees where eid not in(select eid from purchases);

Correlated: select eid,name from employees emp where not exists(select eid from purchases where emp.eid=eid);

Q5) select *from purchases where cid in(select cid from customers where cid in(select cid from products join purchases on products.pid =purchases.pid where eid in(select eid from employees where substr(TELEPHONE#,1,3)='888') and products.REGULAR_PRICE<20) and name like 'K%') ;

select * from purchases where pur# in(select pur# from purchases join customers on customers.cid=purchases.cid join employees on employees.eid=purchases.eid join products on products.pid=purchases.pid where purchases.eid in(select eid from employees where

substr(TELEPHONE#,1,3)='888') and purchases.cid in(select cid from customers where name like 'K%') and purchases.pid in(select pid from products where REGULAR_PRICE<20));

Q6) select PUR#,products.NAME "product name",TO_CHAR(PUR_DATE, 'fmMonth')||' '|| TO_CHAR (PUR_DATE, 'fmDDTH')||', '||TO_CHAR(PUR_DATE, 'YYYY')|| ' ' || TO_CHAR(PUR_DATE,'fmDay') "PUR_DATE",TOTAL,SAVING from purchases join products on products.pid=purchases.pid;

Q7) select eid from employees where substr(employees.TELEPHONE#,1,3) in(select substr(employees.TELEPHONE#,1,3) from employees intersect select substr(customers.TELEPHONE#,1,3) from customers) order by eid asc;

select eid from employees where substr(employees.TELEPHONE#,1,3) in(select substr(customers.TELEPHONE#,1,3) from customers) order by eid asc;

Q8) select name from customers cu where cu.visits_made >1 and cu.cid not in (select purchases.cid from purchases where purchases.unit_price > 100 and

purchases.pur_date = cu.last_visit_date);

Also use grp by purchase.visit_made.

Q9) select eid,name from employees where eid not in(select distinct eid from purchases join products on purchases.pid= products.pid where products.REGULAR_PRICE>150);

select eid,name from employees e where not exists(select distinct eid from purchases join products on purchases.pid= products.pid where products.REGULAR_PRICE>150 and e.eid=purchases.eid);

Q10) select cid,name from customers where cid in(select cid from purchases where cid='c006' or pid in (select pid from purchases where cid='c006')); not right

select cid from purchases where pid in (select pid from purchases where cid='c006') group by(cid) having count(cid)=(select count(pid) from purchases group by(cid) having cid='c006');

Q11) select pid,name from products where pid in(select pid from purchases join employees on purchases.eid=employees.eid where substr(employees.TELEPHONE#,1,3)='666') and REGULAR_PRICE>=150;

select pid,name from products p where not exists(select eid from employees e where substr(e.TELEPHONE#,1,3)='666' and not exists(select eid from purchases p1 where e.eid=p1.eid and p1.pid=p.pid)) and REGULAR_PRICE>150;

Q12) select eid,name from employees where eid in(select eid from purchases join products on products.pid=purchases.pid where products.name='camera' or products.name='chair');

select eid,name from employees where eid in(select eid from purchases join products on products.pid=purchases.pid where products.name='camera'

or products.name='chair' group by(eid) having count(eid)=

(select count(distinct purchases.pid) "total no of customers who purchases" from purchases join products

on products.pid=purchases.pid where products.name='camera' or products.name='chair'));

select eid, name from employees where eid in(select eid from(select count(distinct cid) employeessold,eid from purchases where cid in (select distinct purchases.cid from purchases join products on products.pid=purchases.pid where products.name='camera' or products.name='chair') group by(eid)) where employeessold=(select count(distinct purchases.cid)"total no of customers who purchases" from purchases join products on products.pid=purchases.pid where products.name='camera' or products.name='chair'));

Q13) **select cid,name from customers where cid in(select cid from purchases where total= (select max(total) as total1 from purchases));**

Q14) select Total_sale "total sale amount",eid from(select eid,sum(total) as Total_sale from purchases group by (eid)) where Total_sale=(select max(Total_sale) from(select eid,sum(total) as Total_sale from purchases group by (eid)));

Q15) select to_char(PUR_DATE, 'YYYY/MM'),sum(total) from purchases group by(to_char(PUR_DATE, 'YYYY/MM')) order by (to_char(PUR_DATE, 'YYYY/MM'));

Q16) SELECT pur#, saving FROM (SELECT *FROM purchases ORDER BY saving desc)WHERE rownum <= 3 ORDER BY saving;

Q17) select pur#,savings from(select purchases.QTY*(products.REGULAR_PRICE-purchases.UNIT_PRICE) as savings,pur# from purchases join products on products.pid=purchases.pid order by purchases.QTY*(products.REGULAR_PRICE-purchases.UNIT_PRICE) desc) where rownum<=3 order by savings;

Q18) select * from purchases where total>= all(select total from purchases where cid='c006');

select * from purchases where total>=(select max(total) from purchases where cid='c006');

Q19) select customers.cid,customers.name,x."Total amount spent" from customers join(select cid,"Total amount spent" from(select cid,sum(total)as "Total amount spent" from purchases group by(cid) order by(sum(total)) desc)x where rownum<=3 order by"Total amount spent")x on x.cid=customers.cid;

Q20) select employees.eid,NVL("number of customers served",0) as "number of customers served" from employees left join(select eid,count(distinct cid) as "number of customers served" from purchases group by(eid))x on x.eid=employees.eid ;