**Prac #9**

**SQL Practice III:**

**Writing/Executing SQL queries – Further Exercises**

Note:

This prac (Prac #9) is the last prac of the subject and is double weight thus 4 marks in full.

Do not forget to save each query as soon as after writing/running it. For example, save the query as ‘Q1.sql’ for the first question and keep it for your own records. For the submission of this prac, you are required to submit one WORD or Text file containing all query texts you completed as Q1~Q24 to be marked off for this prac activity.

This prac consists of two section of tasks.

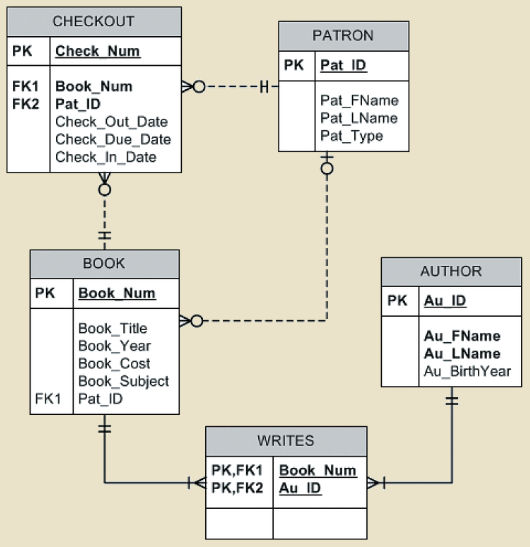
For the first task section, you are going to practice some more advanced SQL query techniques to extract useful information using MySQL Workbench. In particular, you will practice to use subqueries for various purposes. To be continuous from the previous prac, you will use the same library database you created and used in previous pracs, and write a number of SQL queries to extract information from the database.

For the second task section, you are going to practice your overall SQL query techniques using another database provided.

* **Learning outcomes and objectives**

Student will be able to

* write subqueries when it is necessary to process data based on other processed data
* apply various SQL query techniques for a specific database model provided.
* **Task 1 Exercises**

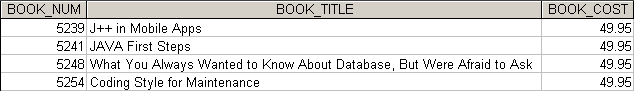
Open the library database you created in the previous lab on MySQL Workbench. The conceptual model of the database is presented in the ERD as shown here.

Use this ERD as a quick reference when you compose SQL queries for Task 1 exercises.

Solutions are provided for you for some exercises. For each of these exercises, you will need to write/save an SQL query (though some exercises already show solutions, you are always recommended to write and run the code yourself).

For each of the following exercises (Q1~Q5), there may be some other various solutions but you are in particular required to compose SQL query by including a sub-query.

1. Write a query to display the book number, title, and cost of books that have the lowest cost of any books in the system. Sort the results by book number. (See the figure below for the output)



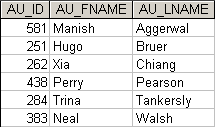
Answer provided:

SELECT BOOK\_NUM, BOOK\_TITLE, BOOK\_COST

FROM BOOK

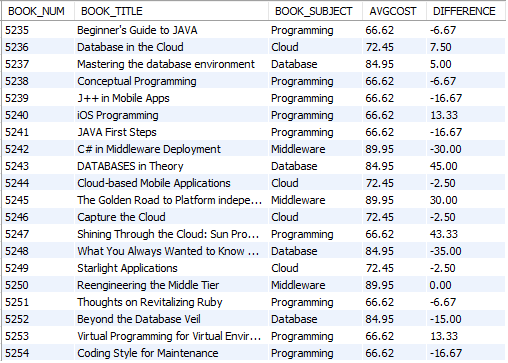
WHERE BOOK\_COST = (SELECT Min(BOOK\_COST) FROM BOOK)

ORDER BY BOOK\_NUM;

1. Write a query to display the author ID, first and last name for all authors that have never written a book with the subject Programming. Sort the results by author last name. (See the figure below for the output)

SELECT AU\_ID, AU\_FNAME, AU\_LNAME   
FROM AUTHOR  
WHERE AU\_ID   
NOT IN (SELECT AU\_ID FROM WRITES   
 WHERE BOOK\_NUM=(SELECT BOOK\_NUM FROM BOOK WHERE BOOK\_SUBJECT='Programming')  
 )  
ORDER BY AU\_LNAME;

1. Write a query to display the book number, title, subject, average cost of books within that subject, and the difference between each book’s cost and the average cost of books in that subject. Sort the results by book number. (See the figure below for the output)



Answer provided:

SELECT BOOK\_NUM, BOOK\_TITLE, BOOK.BOOK\_SUBJECT, Round(AVGCOST, 2) AS AVGCOST, Round(BOOK\_COST - AVGCOST, 2) AS DIFFERENCE

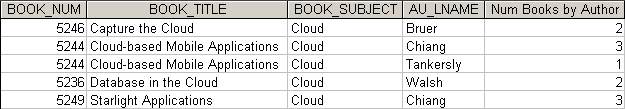
FROM BOOK JOIN (SELECT BOOK\_SUBJECT, Avg(BOOK\_COST) AS AVGCOST

FROM BOOK BOOK2

GROUP BY BOOK\_SUBJECT) AS SUBAVGS ON BOOK.BOOK\_SUBJECT = SUBAVGS.BOOK\_SUBJECT

ORDER BY BOOK\_NUM;

1. Write a query to display the book number, title, subject, author last name, and the number of books written by that author. Limit the results to books in the Cloud subject. Sort the results by book title and then author last name. (See the figure below for the output)



SELECT T1.BOOK\_NUM, T1.BOOK\_TITLE, T1.BOOK\_SUBJECT, T1.AU\_LNAME,

T2.NUM\_BOOKS\_BY\_AUTHOR

FROM

(

SELECT BOOK.BOOK\_NUM, BOOK.BOOK\_TITLE, BOOK.BOOK\_SUBJECT,

AUTHOR.AU\_LNAME, AUTHOR.AU\_ID

FROM BOOK JOIN WRITES ON BOOK.BOOK\_NUM=WRITES.BOOK\_NUM

JOIN AUTHOR ON AUTHOR.AU\_ID=WRITES.AU\_ID

) T1

JOIN

(

SELECT AU\_ID, COUNT(BOOK\_NUM) AS NUM\_BOOKS\_BY\_AUTHOR

WRITES

GROUP BY AU\_ID

) T2 ON T1.AU\_ID=T2.AU\_ID;

1. Write a query to display the lowest average cost of books within a subject and the highest average cost of books within a subject. (See the figure below for the output)

FigP8-67-Lowest-and-Highest-Average-Subject-Cost

SELECT T1.LOWEST\_AVG\_COST, T2.HIGHEST\_AVG\_COST

FROM

(

SELECT BOOK\_SUBJECT, AVG(BOOK\_COST) AS LOWEST\_AVG\_COST

FROM BOOK

GROUP BY BOOK\_SUBJECT

ORDER BY AVG(BOOK\_COST)

LIMIT 1

) T1,

(

SELECT BOOK\_SUBJECT, AVG(BOOK\_COST) AS HIGHEST\_AVG\_COST

FROM BOOK

GROUP BY BOOK\_SUBJECT

ORDER BY AVG(BOOK\_COST) DESC

LIMIT 1

) T2;

* **Task 2 Exercises**

For this task exercises, you will use a database named LargeCo database. Using the text file provided: LargeCo\_MySQL.txt, you are required to build this database on MySQL Workbench before proceeding the following SQL exercises.

The LargeCo database stores data for a company that sells paint products. The company tracks the sale of products to customers. The database keeps data on customers (LGCUSTOMER), sales (LGINVOICE), products (LGPRODUCT), which products are on which invoices (LGLINE), employees (LGEMPLOYEE), the salary history of each employee (LGSALARY\_HISTORY), department (LGDEPARTMENT), product brands (LGBRAND), vendors (LGVENDOR), and which vendors supply each product (LGSUPPLIES).

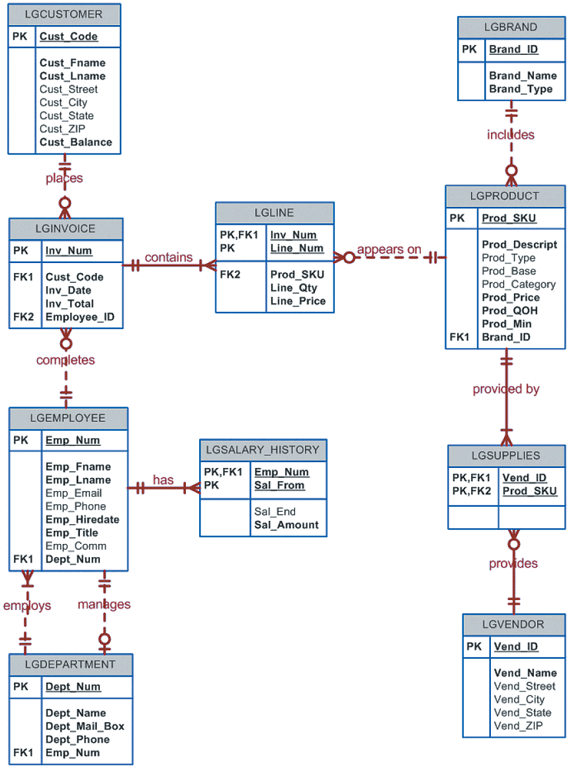
Use the following ERD as a quick reference when you compose SQL queries for this task exercises.

Note:

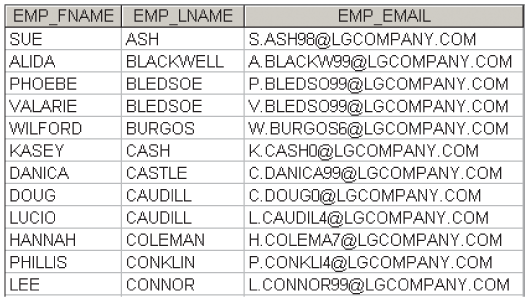
Some of the tables contain only a few rows of data, while other tables are quite larger; for example, there are only eight departments, but more than 3,300 invoices containing over 11,000 invoice lines. For the following SQL problems, a figure of the correct output for each question is provided. If the output of the query is very large, only the first several rows of the output are shown.

Reference:

Coronel/Morris textbook 13th edition Ch7 Problems Q27~Q55.



1. Write a query to display the first name, last name, and email address of employees hired from January 1, 2005, to December 31, 2014. Sort the output by last name and then by first name. (See the figure below for first 12 rows of the output. Actual output has 104 rows)

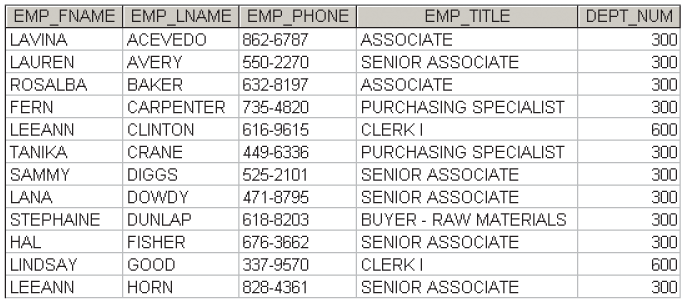


SELECT EMP\_FNAME,EMP\_LNAME,EMP\_EMAIL FROM LGEMPLOYEE

WHERE EMP\_HIREDATE>='2005-01-01' AND EMP\_HIREDATE<='2014-12-31'

ORDER BY EMP\_LNAME, EMP\_FNAME;

1. Write a query to display the first name, last name, phone number, title, and department number of employees who work in department 300 or have the title “CLERK I”. Sort the results by last name and then by first name. (See the figure below for first 12 rows of the output. Actual output has 31 rows)

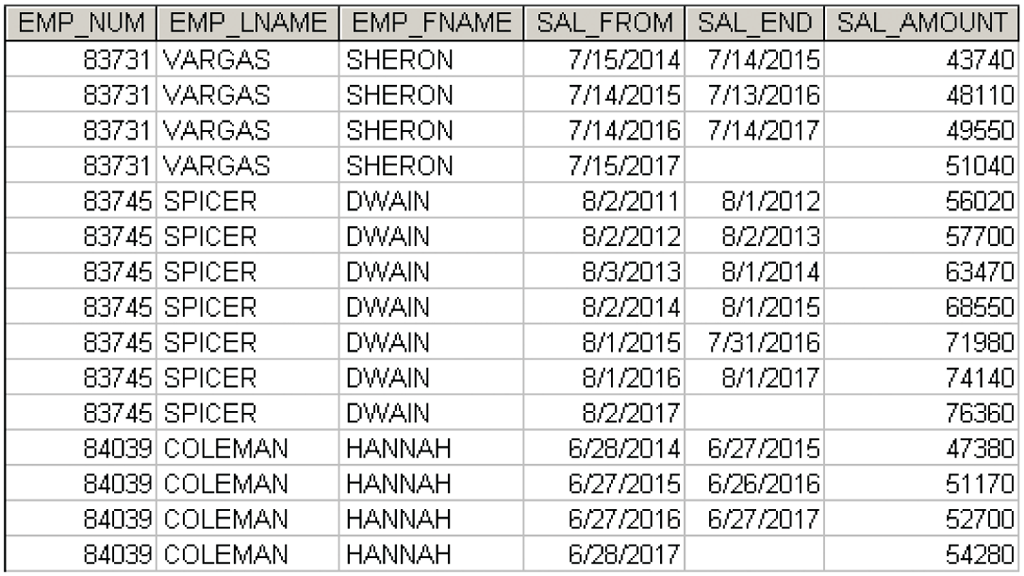


SELECT EMP\_FNAME,EMP\_LNAME,EMP\_PHONE, EMP\_TITLE, DEPT\_NUM FROM LGEMPLOYEE

WHERE EMP\_TITLE LIKE '%CLERK I' OR DEPT\_NUM=300

ORDER BY EMP\_LNAME, EMP\_FNAME;

1. Write a query to display the employee number, last name, first name, salary “from” date, salary “end” date, and salary amount for employees 83731, 83745, and 84039. Sort the results by employee number and then salary “from” date (See the figure below for the output)



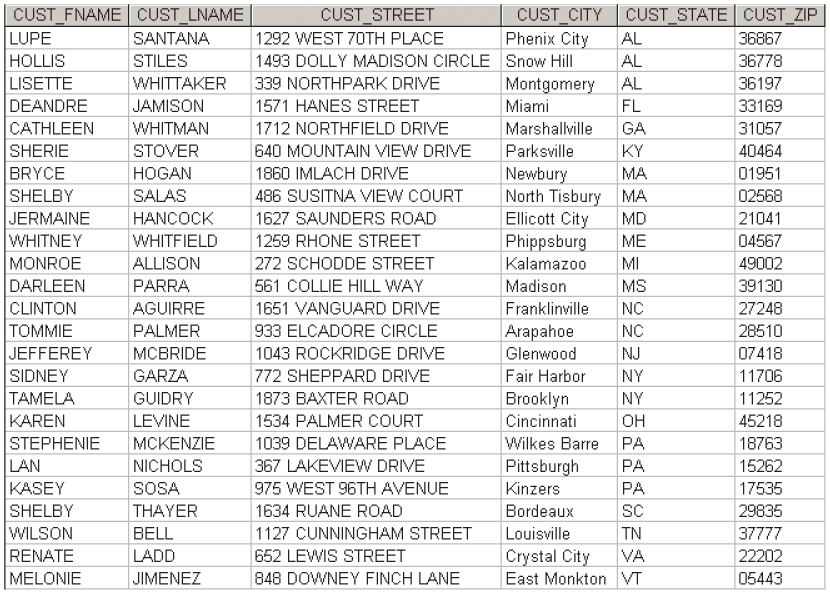
SELECT LGEMPLOYEE.EMP\_NUM, EMP\_FNAME,EMP\_LNAME,SAL\_FROM, SAL\_END, SAL\_AMOUNT

FROM LGEMPLOYEE JOIN LGSALARY\_HISTORY ON LGEMPLOYEE.EMP\_NUM=LGSALARY\_HISTORY.EMP\_NUM

WHERE LGEMPLOYEE.EMP\_NUM IN ('83731', '83745','84039')

ORDER BY LGEMPLOYEE.EMP\_NUM,SAL\_FROM;

1. Write a query to display the first name, last name, street, city, state, and zip code of any customer who purchased a Foresters Best brand top coat between July 15, 2017, and July 31, 2017. If a customer purchased more than one such product, display the customer’s information only once in the output. Sort the result by state, last name, and then first name. (See the figure below for the output)



SELECT DISTINCT CUST\_FNAME, CUST\_LNAME, CUST\_STREET, CUST\_CITY, CUST\_STATE, CUST\_ZIP

FROM LGCUSTOMER

JOIN LGINVOICE ON LGCUSTOMER.CUST\_CODE= LGINVOICE.CUST\_CODE

JOIN LGLINE ON LGLINE.INV\_NUM=LGINVOICE.INV\_NUM

JOIN LGPRODUCT ON LGPRODUCT.PROD\_SKU=LGLINE.PROD\_SKU

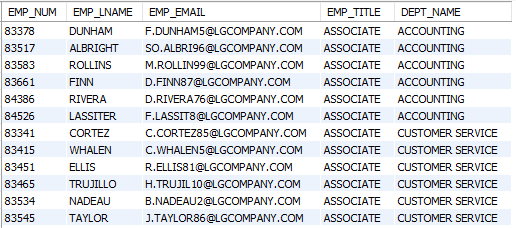
JOIN LGBRAND ON LGBRAND.BRAND\_ID=LGPRODUCT.BRAND\_ID

WHERE LGINVOICE.INV\_DATE BETWEEN '2017-07-15' AND '2017-07-31'

AND LGBRAND.BRAND\_NAME='FORESTERS BEST' AND LGPRODUCT.PROD\_CATEGORY='Top Coat'

ORDER BY CUST\_STATE, CUST\_LNAME, CUST\_FNAME;

1. Write a query to display the employee number, last name, email address, title, and department name of each employee whose job title ends in the word “ASSOCIATE”. Sort the result by department name, employee title, and then employee number. (See the figure below for the first 12 row output. Actual output has 168 rows.)



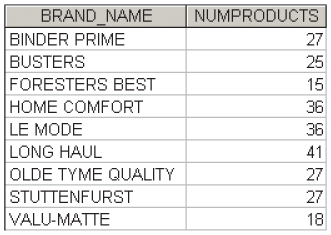
SELECT LGEMPLOYEE.EMP\_NUM, EMP\_LNAME,EMP\_EMAIL,EMP\_TITLE, DEPT\_NAME

FROM LGEMPLOYEE JOIN LGDEPARTMENT ON LGEMPLOYEE.DEPT\_NUM=LGDEPARTMENT.DEPT\_NUM

WHERE EMP\_TITLE LIKE '%ASSOCIATE'

ORDER BY DEPT\_NAME, EMP\_TITLE, LGEMPLOYEE.EMP\_NUM;

1. Write a query to display a brand name and the number of products of that brand that are in the database. Sort the result by brand name. (See the figure below for the output)



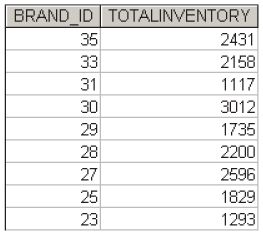
SELECT BRAND\_NAME, COUNT(LGPRODUCT.PROD\_SKU) AS NUMPRODUCTS

FROM LGBRAND JOIN LGPRODUCT ON LGBRAND.BRAND\_ID=LGPRODUCT.BRAND\_ID

GROUP BY BRAND\_NAME

ORDER BY BRAND\_NAME;

1. Write a query to display the total inventory – that is, the sum of all products on hand for each brand ID. Sort the result by brand ID in descending order. (See the figure below for the output)



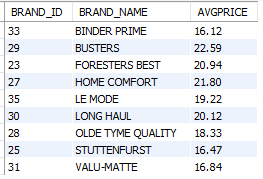
SELECT BRAND\_ID, SUM(PROD\_QOH) AS TOTALINVENTORY

FROM LGPRODUCT

GROUP BY BRAND\_ID

ORDER BY BRAND\_ID DESC;

1. Write a query to display the brand ID, brand name, and average price of products of each brand. Sort the result by brand name. Results are shown with the average price rounded to two decimal places. (See the figure below for the output) 🡪 Hint: use round() function to set the decimal places of the average price: round(avg(prod\_price),2)



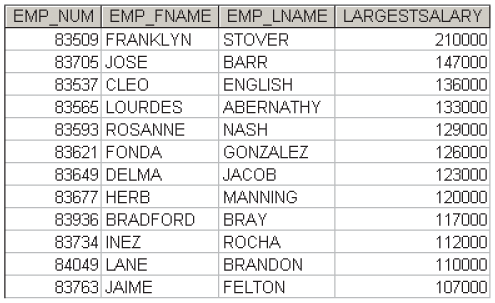
SELECT LGBRAND.BRAND\_ID, BRAND\_NAME, ROUND(AVG(PROD\_PRICE),2) AS AVGPRICE

FROM LGBRAND JOIN LGPRODUCT ON LGBRAND.BRAND\_ID=LGPRODUCT.BRAND\_ID

GROUP BY LGBRAND.BRAND\_ID,BRAND\_NAME

ORDER BY BRAND\_NAME;

1. Write a query to display the employee number, first name, last name, and largest salary amount for each employee in department 200. Sort the result by largest salary in descending order. (See the figure below for the first 12 rows of the output. Actual output has 33 rows)



SELECT LGEMPLOYEE.EMP\_NUM, EMP\_FNAME,EMP\_LNAME, MAX(SAL\_AMOUNT)

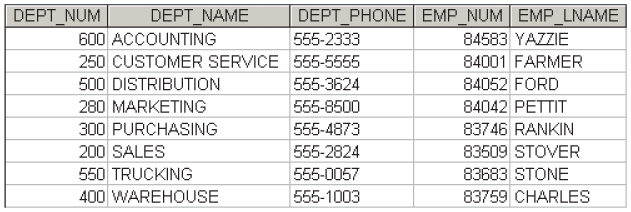
FROM LGEMPLOYEE JOIN LGSALARY\_HISTORY ON LGEMPLOYEE.EMP\_NUM=LGSALARY\_HISTORY.EMP\_NUM

WHERE DEPT\_NUM=200

GROUP BY LGEMPLOYEE.EMP\_NUM, EMP\_FNAME,EMP\_LNAME

ORDER BY MAX(SAL\_AMOUNT) DESC;

1. Write a query to display the department number, department name, department phone number, employee number, and last name of each department manager. Sort the result by department name. (See the figure below for the output)

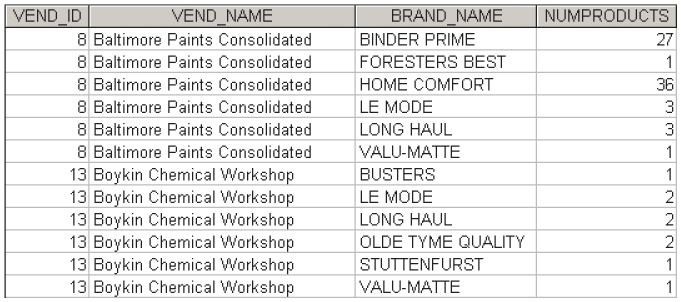


SELECT LGDEPARTMENT.DEPT\_NUM, DEPT\_NAME, DEPT\_PHONE, LGDEPARTMENT.EMP\_NUM, LGEMPLOYEE.EMP\_LNAME

FROM LGDEPARTMENT JOIN LGEMPLOYEE ON LGDEPARTMENT.EMP\_NUM=LGEMPLOYEE.EMP\_NUM

ORDER BY DEPT\_NAME;

1. Write a query to display the vendor ID, vendor name, brand name, and number of products of each brand supplied by each vendor. Sort the result by vendor name, and then by brand name. (See the figure below for the first 12 rows of the output. Actual output has 139 rows)



SELECT LGVENDOR.VEND\_ID, VEND\_NAME, LGBRAND.BRAND\_NAME, COUNT(LGSUPPLIES.PROD\_SKU) AS NUMPRODUCTS

FROM LGVENDOR

JOIN LGSUPPLIES ON LGSUPPLIES.VEND\_ID= LGVENDOR.VEND\_ID

JOIN LGPRODUCT ON LGPRODUCT.PROD\_SKU=LGSUPPLIES.PROD\_SKU

JOIN LGBRAND ON LGBRAND.BRAND\_ID=LGPRODUCT.BRAND\_ID

GROUP BY LGVENDOR.VEND\_ID, VEND\_NAME, LGBRAND.BRAND\_NAME

ORDER BY VEND\_NAME, LGBRAND.BRAND\_NAME;

1. Write a query to display the largest average product price of any brand. (See the figure below for the output)

C:\Users\jc157544\AppData\Local\Microsoft\Windows\INetCache\Content.Word\FigP7-45.gif

SELECT ROUND(MAX(AVG\_PROD\_PRICE),2) AS LARGEST\_AVERAGE

FROM

(

SELECT BRAND\_ID, AVG(PROD\_PRICE) AS AVG\_PROD\_PRICE

FROM LGPRODUCT

GROUP BY BRAND\_ID

) T;

1. Write a query to display the brand ID, brand name, brand type, and average price of products for the brand that has the largest average product price (See the figure below for the output)

C:\Users\jc157544\AppData\Local\Microsoft\Windows\INetCache\Content.Word\FigP7-46.gif

SELECT T1.\*

FROM

(

SELECT LGPRODUCT.BRAND\_ID, BRAND\_NAME, BRAND\_TYPE, ROUND(AVG(PROD\_PRICE),2) AS AVGPRICE

FROM LGPRODUCT JOIN LGBRAND ON LGBRAND.BRAND\_ID=LGPRODUCT.BRAND\_ID

GROUP BY LGPRODUCT.BRAND\_ID, BRAND\_NAME, BRAND\_TYPE

) T1,

(

SELECT ROUND(MAX(AVG\_PROD\_PRICE),2) AS LARGEST\_AVERAGE

FROM

(

SELECT BRAND\_ID, AVG(PROD\_PRICE) AS AVG\_PROD\_PRICE

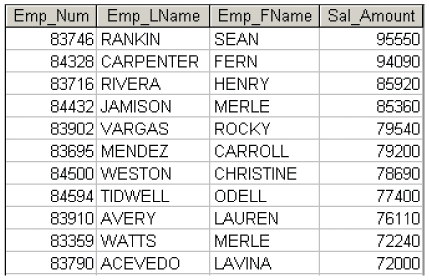
FROM LGPRODUCT

GROUP BY BRAND\_ID

) T

) T2 WHERE T1.AVGPRICE=T2.LARGEST\_AVERAGE

;

1. Write a query to display the current salary for each employee in department 300. Assume that only current employees are kept in the system, and therefore the most current salary for each employee is the entry in the salary history with a NULL end date. Sort the result in descending order by salary amount. (See the figure below for the first 11 rows of the output. Actual output has 25 rows)

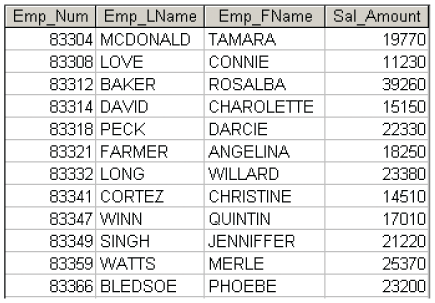
SELECT LGEMPLOYEE.EMP\_NUM, EMP\_LNAME, EMP\_FNAME, SAL\_AMOUNT

FROM LGEMPLOYEE JOIN LGSALARY\_HISTORY ON LGEMPLOYEE.EMP\_NUM=LGSALARY\_HISTORY.EMP\_NUM

WHERE DEPT\_NUM=300 AND LGSALARY\_HISTORY.SAL\_END IS NULL

ORDER BY SAL\_AMOUNT DESC;

1. Write a query to display the starting salary for each employee. The starting salary would be the entry in the salary history with the oldest salary start date for each employee. Sort the output by employee number. (See the figure below for the first 12 rows of the output. Actual output has 363 rows)



SELECT LGEMPLOYEE.EMP\_NUM,EMP\_LNAME, EMP\_FNAME,SAL\_AMOUNT

FROM LGEMPLOYEE JOIN LGSALARY\_HISTORY ON LGEMPLOYEE.EMP\_NUM=LGSALARY\_HISTORY.EMP\_NUM

JOIN

(

SELECT EMP\_NUM, MIN(SAL\_FROM) AS OLDEST\_SAL

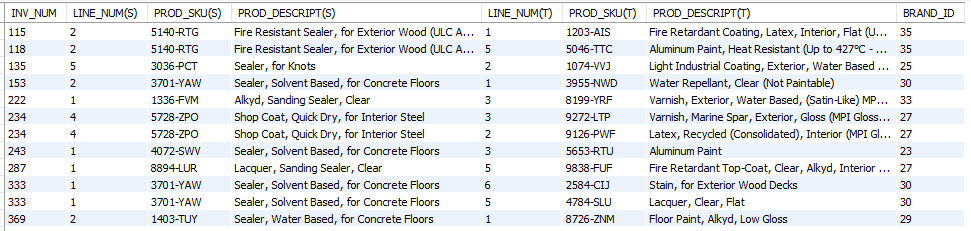
FROM LGSALARY\_HISTORY

GROUP BY EMP\_NUM

) T ON T.EMP\_NUM=LGEMPLOYEE.EMP\_NUM AND T.OLDEST\_SAL=LGSALARY\_HISTORY.SAL\_FROM

ORDER BY LGEMPLOYEE.EMP\_NUM;

1. Write a query to display the invoice number, line numbers, product SKUs, product descriptions, and brand ID for sales of sealer and top coat products of the same brand on the same invoice. Sort the results by invoice number in ascending order, first line number in ascending order, and then by second line number in descending order. (See the figure below for the first 12 rows of the output. Actual output has 130 rows)



SELECT T1.INV\_NUM, T1.LINE\_NUM AS "LINE\_NUM(S)", T1.PROD\_SKU AS "PROD\_SKU(S)",

T1.PROD\_DESCRIPT AS "PROD\_DESCRIPT(S)", T2.LINE\_NUM AS "LINE\_NUM(T)", T2.PROD\_SKU AS "PROD\_SKU(T)",

T2.PROD\_DESCRIPT AS "PROD\_DESCRIPT(T)", T1.BRAND\_ID

FROM

(

SELECT LGINVOICE.INV\_NUM, LGLINE.LINE\_NUM, LGPRODUCT.PROD\_SKU, PROD\_DESCRIPT, BRAND\_ID

FROM LGINVOICE

JOIN LGLINE ON LGLINE.INV\_NUM=LGINVOICE.INV\_NUM

JOIN LGPRODUCT ON LGPRODUCT.PROD\_SKU=LGLINE.PROD\_SKU

WHERE PROD\_CATEGORY='Sealer'

) T1

JOIN

(

SELECT LGINVOICE.INV\_NUM, LGLINE.LINE\_NUM, LGPRODUCT.PROD\_SKU, PROD\_DESCRIPT, BRAND\_ID

FROM LGINVOICE

JOIN LGLINE ON LGLINE.INV\_NUM=LGINVOICE.INV\_NUM

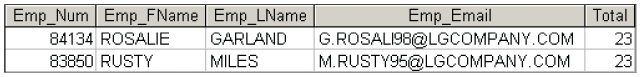
JOIN LGPRODUCT ON LGPRODUCT.PROD\_SKU=LGLINE.PROD\_SKU

WHERE PROD\_CATEGORY='Top Coat'

) T2 ON T1.INV\_NUM=T2.INV\_NUM AND T1.BRAND\_ID=T2.BRAND\_ID

ORDER BY T1.INV\_NUM ASC, T1.LINE\_NUM ASC, T2.LINE\_NUM DESC;

1. The Binder Prime Company wants to recognize the employee who sold the most of its products during a specified period. Write a query to display the employee number, employee first name, employee last name, email address, and total units sold for the employee who sold the most Binder Prime brand products between November 1, 2017, and December 5, 2017. If there is a tie for most units sold, sort the output by employee last name. (See the figure below for the output)



SELECT T.\*

FROM

(

SELECT LGEMPLOYEE.EMP\_NUM, EMP\_FNAME,EMP\_LNAME,EMP\_EMAIL, SUM(LGLINE.LINE\_QTY) AS TOTAL

FROM LGEMPLOYEE JOIN LGINVOICE ON LGEMPLOYEE.EMP\_NUM=LGINVOICE.EMPLOYEE\_ID

JOIN LGLINE ON LGLINE.INV\_NUM=LGINVOICE.INV\_NUM

JOIN LGPRODUCT ON LGPRODUCT.PROD\_SKU=LGLINE.PROD\_SKU

JOIN LGBRAND ON LGBRAND.BRAND\_ID=LGPRODUCT.BRAND\_ID

WHERE LGINVOICE.INV\_DATE BETWEEN '2017-11-01' AND '2017-12-05'

AND LGBRAND.BRAND\_NAME='BINDER PRIME'

GROUP BY LGEMPLOYEE.EMP\_NUM, EMP\_FNAME,EMP\_LNAME,EMP\_EMAIL

) T

WHERE

TOTAL=

(

SELECT MAX(TOTAL)

FROM

(

SELECT LGEMPLOYEE.EMP\_NUM, EMP\_FNAME,EMP\_LNAME,EMP\_EMAIL, SUM(LGLINE.LINE\_QTY) AS TOTAL

FROM LGEMPLOYEE JOIN LGINVOICE ON LGEMPLOYEE.EMP\_NUM=LGINVOICE.EMPLOYEE\_ID

JOIN LGLINE ON LGLINE.INV\_NUM=LGINVOICE.INV\_NUM

JOIN LGPRODUCT ON LGPRODUCT.PROD\_SKU=LGLINE.PROD\_SKU

JOIN LGBRAND ON LGBRAND.BRAND\_ID=LGPRODUCT.BRAND\_ID

WHERE LGINVOICE.INV\_DATE BETWEEN '2017-11-01' AND '2017-12-05'

AND LGBRAND.BRAND\_NAME='BINDER PRIME'

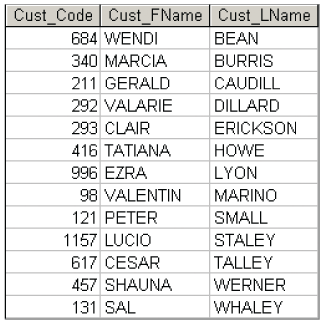
GROUP BY LGEMPLOYEE.EMP\_NUM, EMP\_FNAME,EMP\_LNAME,EMP\_EMAIL

) TT

)

ORDER BY T.EMP\_LNAME;

1. Write a query to display the customer code, first name, and last name of all customers who have had at least one invoice completed by employee 83649 and at least one invoice completed by employee 83677. Sort the output by customer last name and then first name. (See the figure below for the output)



SELECT DISTINCT T1.\*

FROM

(

SELECT LGCUSTOMER.CUST\_CODE, CUST\_FNAME, CUST\_LNAME

FROM LGCUSTOMER

JOIN LGINVOICE ON LGCUSTOMER.CUST\_CODE= LGINVOICE.CUST\_CODE

WHERE LGINVOICE.EMPLOYEE\_ID='83649'

) T1

JOIN

(

SELECT LGCUSTOMER.CUST\_CODE, CUST\_FNAME, CUST\_LNAME

FROM LGCUSTOMER

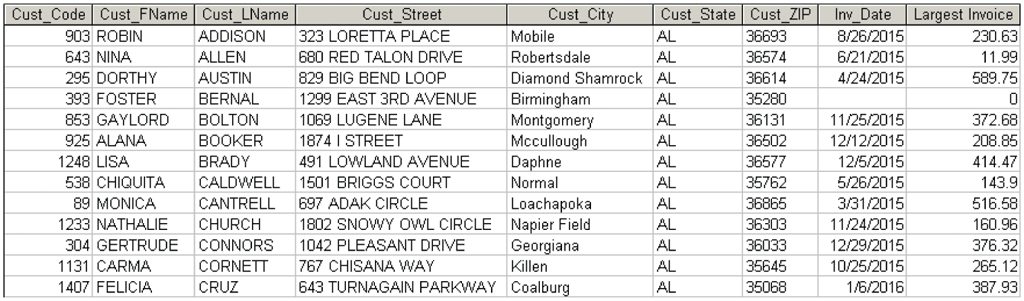
JOIN LGINVOICE ON LGCUSTOMER.CUST\_CODE= LGINVOICE.CUST\_CODE

WHERE LGINVOICE.EMPLOYEE\_ID='83677'

) T2 ON T1.CUST\_CODE=T2.CUST\_CODE

ORDER BY T1.CUST\_LNAME, T1.CUST\_FNAME;

1. LargeCo is planning a new promotion in Alabama (AL) and wants to know about the largest purchases made by customers in that state. Write a query to display the customer code, customer first name, last name, full address, invoice date, and invoice total of the largest purchases made by each customer in Alabama. Be certain to include any customers in Alabama who have never made a purchase; their invoice dates should be NULL and the invoice totals should display as 0. Sort the result by customer last name and then first name. (See the figure below for the first 13 rows of the output. Actual output has 50 rows.)



SELECT T1.\*

FROM

(

SELECT DISTINCT LGCUSTOMER.CUST\_CODE, CUST\_FNAME, CUST\_LNAME, CUST\_STREET, CUST\_CITY,

CUST\_STATE, CUST\_ZIP, LGINVOICE.INV\_DATE, IFNULL(MAX(INV\_TOTAL),0) AS LARGEST\_INVOICE

FROM LGCUSTOMER

LEFT JOIN LGINVOICE ON LGCUSTOMER.CUST\_CODE= LGINVOICE.CUST\_CODE

WHERE CUST\_STATE='AL'

GROUP BY LGCUSTOMER.CUST\_CODE, CUST\_FNAME, CUST\_LNAME, CUST\_STREET, CUST\_CITY,

CUST\_STATE, CUST\_ZIP, LGINVOICE.INV\_DATE

ORDER BY CUST\_LNAME, CUST\_FNAME

) T1

JOIN

(

SELECT DISTINCT LGCUSTOMER.CUST\_CODE, CUST\_FNAME, CUST\_LNAME, CUST\_STREET, CUST\_CITY,

CUST\_STATE, CUST\_ZIP, IFNULL(MAX(INV\_TOTAL),0) AS LARGEST\_INVOICE

FROM LGCUSTOMER

LEFT JOIN LGINVOICE ON LGCUSTOMER.CUST\_CODE= LGINVOICE.CUST\_CODE

WHERE CUST\_STATE='AL'

GROUP BY LGCUSTOMER.CUST\_CODE, CUST\_FNAME, CUST\_LNAME, CUST\_STREET, CUST\_CITY,

CUST\_STATE, CUST\_ZIP

ORDER BY CUST\_LNAME, CUST\_FNAME

) T2 ON T1.CUST\_CODE=T2.CUST\_CODE AND T1.LARGEST\_INVOICE=T2.LARGEST\_INVOICE;

This is the end of Prac #9 Lab.

You are required to submit one document having all SQL queries you composed for this prac.