ACIT 1630

Relational Database Design and SQL

Review Questions

Lesson 11

Answer each of the following questions labeling your answers clearly. Save your work in the Desire2Learn **Drop box** for **Lesson 08**.

**Part A:**

1. What is the difference between UNION and UNION ALL?

UNION keeps unique records, UNIION ALL keeps all records, even duplicate values.

1. Suppose that you have two tables, EMPLOYEE and EMPLOYEE\_1. The EMPLOYEE table contains the records for four employees: Alice Cordoza, John Cretchakov, Susan Harrison, and Anne McDonald. The EMPLOYEE\_1 table contains the records for two employees: John Cretchakov and Mary Chen. Given that information, list the query output for the UNION query.

The output is:

Alice Cordoza

John Cretchakov

Susan Harrison

Anne McDonald

Mary Chen

1. Given the employee information in question 2 above, list the query output for the UNION ALL query.

The output is:

Alice Cordoza

John Cretchakov

Susan Harrison

Anne McDonald

John Cretchakov

Mary Chen

1. What is the difference between an inner join and an outer join?

INNER JOIN selects records that have matching pairs in both tables, when given criteria.

OUTER JOIN selects records that have matching pairs in both tables but also keeps the unmatched pairs.

1. Suppose that a PRODUCT table contains two attributes, PROD\_CODE and VEND\_CODE. Those two attributes have values of ABC, 125, DEF, 124, GHI, 124, and JKL, 123, respectively. The VENDOR table contains a single attribute, VEND\_CODE, with values 123, 124, 125, and 126, respectively. The VEND\_CODE attribute in the PRODUCT table is a foreign key to the VEND\_CODE in the VENDOR table. Given that information, what would be the query output for a UNION query based on these two tables? (Because the common attribute is V\_CODE, the output will only show the V\_CODE values generated by the each query.)

It would be 125, 124, 123, 126. Duplicates won’t be shown

1. Suppose that a PRODUCT table contains two attributes, PROD\_CODE and VEND\_CODE. Those two attributes have values of ABC, 125, DEF, 124, GHI, 124, and JKL, 123, respectively. The VENDOR table contains a single attribute, VEND\_CODE, with values 123, 124, 125, and 126, respectively. The VEND\_CODE attribute in the PRODUCT table is a foreign key to the VEND\_CODE in the VENDOR table. Given that information, what would be the query output for a UNION ALL query based on these two tables? (Because the common attribute is V\_CODE, the output will only show the V\_CODE values generated by the each query.)

It would be: 125,124,124,123,123,124,125,126. Duplicates included.

1. What are the three join types included in the OUTER JOIN classification? Describe each of the types.

LEFT OUTER JOIN will yield all the matching rows in the join columns, including the unmatched rows in the Left(first) table.

RIGHT OUTER JOIN will yield all the matching rows in the join columns, including the unmatched rows in the Right(second) table.

FULL OUTER JOIN will yield all the matching rows in the join columns, including all the unmatched rows from both tables.

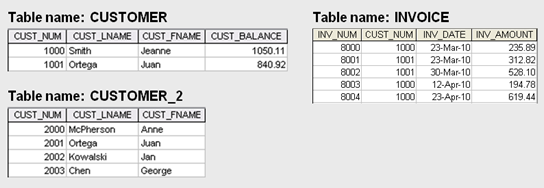
1. What string function should you use to list the first three characters of a company’s EMP\_LNAME values? Give an example, using a table named EMPLOYEE.

SELECT SUBSTRING(EMP\_LNAME, 1, 3)

FROM EMPLOYEE

1. Describe a SQL batch.

It is a set of SQL commands executed together as a group.



1. Using the INVOICE table shown above, write the query that will show the invoice number, the invoice amount, and the average invoice amount from the INVOICE table.

select inv\_num, inv\_amount, avg(inv\_amount)

from INVOICE

1. Using the CUSTOMER table shown above, modify the CUSTOMER table to included two new attributes: CUST\_DOB and CUST\_AGE.

alter table CUSTOMER

add CUST\_DOB date,

CUST\_AGE int

1. Using the CUSTOMER table shown above, modify the birth date for customer 1000 to March 15, 1979, and customer 1001 to December 22, 1988.

UPDATE CUSTOMER

SET CUST\_DOB = '1979-03-15'

WHERE CUST\_NUM = 1000

UPDATE CUSTOMER

SET CUST\_DOB = '1988-12-22'

WHERE CUST\_NUM = 1000

1. Explain why the two following commands produce different results.

SELECT DISTINCT COUNT (V\_CODE) FROM PRODUCT;

SELECT COUNT (DISTINCT V\_CODE) FROM PRODUCT;

In the first query we get the count of V\_CODE, and then we get the DISTINCT value, whereas the second query we take the DISTINCT value of V\_CODE and THEN the Count Value of V\_CODE.

1. Explain the difference between an ORDER BY clause and a GROUP BY clause.

GROUP BY is used to group results according to distinct values of a selected column.

ORDER BY sorts the rows in ascending or descending order, according to the selected column values.

1. Using the INVOICE table shown above, write the query to show the number of invoices that have invoice amount over $300.

Select \* From INVOICE

Where INV\_AMOUNT > 300

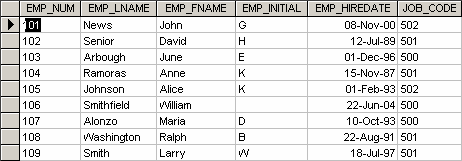
order by ASC

1. In a SELECT query, what is the difference between a WHERE clause and a HAVING clause?

The WHERE clause yields rows that match given constraints/conditions.

The HAVING clause will filter results based on an aggregate function.

**EMPLOYEE**



1. Using the EMPLOYEE table shown above, write the SQL code to insert the rows for employee numbers 106 and 107 for the table.

INSERT INTO EMPLOYEE (EMP\_NUM, EMP\_LNAME, EMP\_FNAME, EMP\_INITIAL,

EMP\_HIREDATE, JOB\_CODE) VALUES

(105, 'Johnson', 'Alice', 'K', '01-Feb-93', 502),

(106, 'Smithfield', 'William', null, '22-Jun-04', 500);

1. Using the EMPLOYEE table shown above, write the SQL code to change the job code to 500 for the person whose employee number is 101.

UPDATE EMPLOYEE

SET JOB\_CODE = 500

WHERE EMP\_NUM = 101

1. Using the EMPLOYEE table shown above, write the SQL code to delete the row for the person named William Smithfield, who was hired on June 22, 2004, and whose job code classification is 500.

DELETE FROM EMPLOYEE

WHERE (EMP\_FNAME ='William' and EMP\_LNAME = 'Smithfield' and JOB\_CODE = 500)

1. Using the EMPLOYEE table shown above, write the SQL code to change the job classification (JOB\_CODE) to 502 for all employees whose job classification (JOB\_CODE) is 501.

UPDATE EMPLOYEE

SET JOB\_CODE = 502

WHERE JOB\_CODE = 501

**Part B:**

1. List the title from the titles table, the order number and order date from the sales table, and the store name from the stores table. Display only the first 30 characters of the title column. There should be a row produced in the result set for each row in the titles table. Order the result set by the order number. The query should produce the result set listed below. (Hint: LEFT OUTER JOIN statement)

title ord\_num ord\_date stor\_name

------------------------------------------- ---------------- ------------------------------ -------------------------

Net Etiquette NULL NULL NULL

The Psychology of Computer Coo NULL NULL NULL

The Gourmet Microwave 423LL922 1994-09-14 00:00:00.000 Bookbeat

.....

Straight Talk About Computers QQ2299 1993-10-28 00:00:00.000 Fricative Bookshop

Silicon Valley Gastronomic Tre TQ456 1993-12-12 00:00:00.000 Fricative Bookshop

You Can Combat Computer Stress X999 1993-02-21 00:00:00.000 Fricative Bookshop

(23 row(s) affected)

select substring(title, 1, 30) as title, ord\_num, ord\_date, stor\_name

from titles

left outer join sales on titles.title\_id = sales.title\_id

left outer join stores on sales.stor\_id = stores.stor\_id

order by ord\_num

1. Using the INSERT command, add two books to the titles table. The first book has a title id of ZZ1234, title of Microsoft SQL Server, type of computer, publisher id of 0877, price of $89.99, and a publish date of September 29, 2008. The second book has a title id of TT2345, title of Designing Databases, type of computer, publisher id of 1389, priced of $168.98, and a publish date of January 1, 2009.

insert into titles (title\_id, title, type, pub\_id, price, pubdate) values

('ZZ1234','Microsoft SQL Server','computer','0877',89.99, '2008-09-29'),

('TT2345','Designing Databases','computer','1389',168.98, '2009-01-01')

1. List the sum of the order cost (quantity \* price) for each order by store id, store name, order number, quantity, title, and price where the sum of the order cost is between $150.00 and $500.00. Display the order number, store id, and quantity from the sales table, the store name from the stores table, the price from the titles table, and the sum of the order cost. The query should produce the result set listed below.

ord\_num stor\_id stor\_name qty price order\_cost

-------------- ------- -------------------------------------------------- ------ --------- ------------

P2121 7067 News & Brews 20 11.95 239.00

P2121 7067 News & Brews 20 14.99 299.80

N914008 7131 Doc-U-Mat: Quality Laundry and Books 20 10.95 219.00

…

TQ456 7896 Fricative Bookshop 10 19.99 199.90

423LL930 8042 Bookbeat 10 19.99 199.90

P723 8042 Bookbeat 25 11.95 298.75

(11 row(s) affected)

select sales.ord\_num, sales.stor\_id, stores.stor\_name, sales.qty, titles.price, (sales.qty \* titles.price) as order\_cost

from sales

left outer join titles on titles.title\_id = sales.title\_id

left OUTER join stores on sales.stor\_id = stores.stor\_id

WHERE ((sales.qty\*titles.price) > 150.00) AND ((sales.qty\*titles.price) < 500.00)

1. Using the UPDATE command, increase the price by 10% for the book Microsoft SQL Server with a title id of ZZ1234.

UPDATE titles

SET price = (price \* 1.1)

WHERE title\_id = 'ZZ1234'

1. Create a new table called BusinessBooks containing the title id, title, and price columns along with the data from the titles table which have a type of business. There should be 4 rows inserted into the new table.

select title\_id, title, price, type

into BusinessBooks

from titles

where type = 'business'

1. Delete the books with a title id of ZZ1234 and TT2345 from the titles.

delete from titles

where (title\_id = 'ZZ1234') or (title\_id = 'TT2345')

1. List the average price and the sum of the price of the books for each type and the running total of all book prices from the titles table. The query should produce the result set listed below.

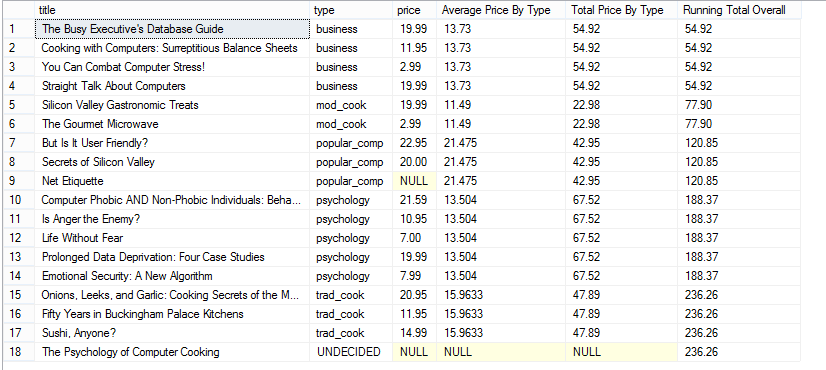
SELECT title, type, price,

AVG(price) OVER (PARTITION BY type) AS 'Average Price By Type',

SUM(price) OVER (PARTITION BY type) AS 'Total Price by Type',

SUM(price) OVER (ORDER BY type) AS 'Running Total Overall'

FROM titles



1. Using the UNION command, list the authors with a state of CA and the publishers with a state that does not contain NULL values. Display the author id, first name, last name, city, and state from the authors table, and the publisher id, publisher name, city, and state from the publishers table. Format the name of the author as the first name followed by a space followed by the last name. Order the result set by the four column - state. The query should produce the result set listed below.

ID Name city state

----------- ---- ---------------------------------------- ------------------- --------------

1389 Algodata Infosystems Berkeley CA

172-32-1176 JohnsonWhite Menlo Park CA

213-46-8915 MarjorieGreen Oakland CA

…..

0736 New Moon Books Boston MA

9952 Scootney Books New York NY

1756 Ramona Publishers Dallas TX

(21 row(s) affected)

SELECT au\_ID, concat(au\_fname,' ',au\_lname) AS Name, city, state

FROM authors

WHERE state = 'CA'

UNION

SELECT pub\_id, pub\_name, city, state

FROM publishers

WHERE state IS NOT NULL

ORDER BY state

1. List the total of books by the type and publisher name. Display the type and minimum price from the titles table, the publisher name from the publishers table, and the sum of the quantity from the sales table. (Hint: Use a GROUP BY statement.)

type pub\_name quantity price

------------------- ----------------------------- ---------- ---------------------

business Algodata Infosystems 55 11.95

popular\_comp Algodata Infosystems 80 20.00

mod\_cook Binnet & Hardley 50 2.99

psychology Binnet & Hardley 20 21.59

trad\_cook Binnet & Hardley 80 11.95

business New Moon Books 35 2.99

psychology New Moon Books 173 7.00

(7 row(s) affected)

SELECT type, pub\_name, SUM(qty), MIN(price)

FROM titles

JOIN publishers ON titles.pub\_id = publishers.pub\_id

JOIN sales ON titles.title\_id = sales.title\_id

GROUP BY type, pub\_name

1. List the sum of the quantity for each order by store id, order number, and order date where the sum of the quantity is between 10 and 50. Display the store id, order number, order date, and the sum of the quantity from the sales table. The query should produce the result set listed below.

stor\_id ord\_num ord\_date sum\_quantity

--------- ------------------ ------------------------------- ----------------

7066 A2976 1993-05-24 00:00:00.000 50

7067 D4482 1994-09-14 00:00:00.000 10

7131 N914008 1994-09-14 00:00:00.000 20

…..

8042 423LL930 1994-09-14 00:00:00.000 10

8042 P723 1993-03-11 00:00:00.000 25

8042 QA879.1 1993-05-22 00:00:00.000 30

(11 row(s) affected)

select stor\_id, ord\_num, ord\_date, sum(qty) as sum\_quantity

from sales

group by stor\_id, ord\_date, ord\_num

having sum(qty) BETWEEN 10 and 50