W3 SQL SUBQUERIES Exercises

1. Write a query to display all the orders from the orders table issued by the salesman 'Paul Adam'.

SELECT \*

FROM orders

WHERE salesman\_id = ( SELECT salesman\_id

FROM salesman

WHERE name = 'Paul Adam')

1. Write a query to display all the orders for the salesman who belongs to the city London.

SELECT \*

FROM orders

WHERE salesman\_id = ( SELECT salesman\_id

FROM salesman

WHERE city = 'London')

1. Write a query to find all the orders issued against the salesman who may works for customer whose id is 3007.

SELECT \*

FROM orders

WHERE salesman\_id = ( SELECT DISTINCT salesman\_id

FROM orders

WHERE customer\_id = 3007)

1. Write a query to display all the orders which values are greater than the average order value for 10th October 2012.

SELECT \*

FROM orders

WHERE purch\_amt > ( SELECT AVG(purch\_amt)

FROM orders

WHERE ord\_date = '2012-10-10')

1. Write a query to find all orders attributed to a salesman in New york.

SELECT \*

FROM orders

WHERE salesman\_id = ( SELECT salesman\_id

FROM salesman

WHERE city = 'New York')

1. Write a query to display the commission of all the salesmen servicing customers in Paris.

SELECT commission

FROM salesman

WHERE salesman\_id = ( SELECT salesman\_id

FROM customer

WHERE city = 'Paris')

1. Write a query to display all the customers whose id is 2001 bellow the salesman ID of Mc Lyon.

SELECT \*

FROM customer

WHERE customer\_id = ( SELECT salesman\_id - 2001

FROM salesman

WHERE name = 'Mc Lyon')

1. Write a query to count the customers with grades above New York's average.

SELECT grade, COUNT (\*)

FROM customer

GROUP BY grade

HAVING grade > ( SELECT AVG(grade)

FROM customer

WHERE city = 'New York');

1. Write a query to extract the data from the orders table for those salesman who earned the maximum commission.

SELECT ord\_no, purch\_amt, ord\_date, salesman\_id

FROM orders

WHERE salesman\_id = (SELECT salesman\_id

FROM salesman

WHERE commission = ( SELECT MAX(commission)

FROM salesman) )

1. Write a query to display all the customers with orders issued on date 17th August, 2012.

SELECT \*

FROM customer

WHERE customer\_id IN ( SELECT customer\_id

FROM orders

WHERE ord\_date = '2012-08-17' )

1. Write a query to find the name and numbers of all salesmen who had more than one customer.

SELECT salesman\_id, name

FROM salesman AS s

WHERE 1 < ( SELECT COUNT(\*)

FROM customer

WHERE salesman\_id=s.salesman\_id)

1. Write a query to find all orders with order amounts which are above-average amounts for their customers.

SELECT \*

FROM orders AS o

WHERE purch\_amt > ( SELECT AVG(purch\_amt)

FROM orders AS d

WHERE o.customer\_id = d.customer\_id);

1. Write a queries to find all orders with order amounts which are on or above-average amounts for their customers.

SELECT \*

FROM orders AS o

WHERE purch\_amt >= ( SELECT AVG(purch\_amt)

FROM orders AS d

WHERE o.customer\_id = d.customer\_id);

1. Write a query to find the sums of the amounts from the orders table, grouped by date, eliminating all those dates where the sum was not at least 1000.00 above the maximum order amount for that date.

SELECT o.ord\_date, SUM(o.purch\_amt)

FROM orders AS o

GROUP BY o.ord\_date

HAVING SUM(o.purch\_amt) > ( SELECT MAX(d.purch\_amt) + 1000

FROM orders AS d

WHERE d.ord\_date = o.ord\_date )

1. Write a query to extract the data from the customer table if and only if one or more of the customers in the customer table are located in London.

SELECT cust\_name, city

FROM customer

WHERE EXISTS ( SELECT \*

FROM customer

WHERE city = 'London' )

1. Write a query to find the salesmen who have multiple customers.

SELECT \*

FROM salesman

WHERE salesman\_id IN ( SELECT DISTINCT salesman\_id

FROM customer AS c

WHERE EXISTS ( SELECT \*

FROM customer AS t

WHERE c.salesman\_id = t.salesman\_id

AND c.cust\_name <> t.cust\_name ) )

1. Write a query to find all the salesmen who worked for only one customer.

SELECT \*

FROM salesman

WHERE salesman\_id IN ( SELECT DISTINCT salesman\_id

FROM customer AS c

WHERE NOT EXISTS ( SELECT \* FROM customer AS t

WHERE c.salesman\_id = t.salesman\_id

AND c.cust\_name <> t.cust\_name));

1. Write a query that extract the rows of all salesmen who have customers with more than one orders.

SELECT \*

FROM salesman AS s

WHERE 1 < ( SELECT COUNT(o.ord\_no)

FROM orders AS o

WHERE o.salesman\_id = s.salesman\_id )

1. Write a query to find salesmen with all information who lives in the city where any of the customers lives.

SELECT \*

FROM salesman

WHERE city = ANY ( SELECT city

FROM customer )

1. Write a query to find all the salesmen for whom there are customers that follow them.

SELECT \*

FROM salesman

WHERE city IN ( SELECT city

FROM customer)

1. Write a query to display the salesmen which name are alphabetically lower than the name of the customers.

SELECT \*

FROM salesman

WHERE EXISTS ( SELECT \*

FROM customer

WHERE salesman.name < customer.cust\_name )

1. Write a query to display the customers who have a greater gradation than any customer who belongs to the alphabetically lower than the city New York.

SELECT \*

FROM customer

WHERE grade > ANY ( SELECT grade

FROM customer

WHERE city < 'New York' )

1. Write a query to display all the orders that had amounts that were greater than at least one of the orders on September 10th 2012.

SELECT \*

FROM orders

WHERE purch\_amt > ( SELECT MIN(purch\_amt)

FROM orders

WHERE ord\_date = '2012-09-10' )

1. Write a query to find all orders with an amount smaller than any amount for a customer in London. (Using ANY keyword).

SELECT \*

FROM orders

WHERE purch\_amt < ANY ( SELECT purch\_amt

FROM orders AS o, customer AS c

WHERE o.customer\_id = c.customer\_id

AND c.city = 'London')

1. Write a query to display all orders with an amount smaller than any amount for a customer in London. (Using MAX).

SELECT \*

FROM orders

WHERE purch\_amt < ( SELECT MAX(purch\_amt)

FROM customer AS c

JOIN orders AS o

ON c.customer\_id = o.customer\_id

WHERE city = 'London' )

1. Write a query to display only those customers whose grade are, in fact, higher than every customer in New York.

SELECT \*

FROM customer

WHERE grade > ALL ( SELECT grade

FROM customer

WHERE city = 'New York' )

1. Write a query in sql to find the name, city, and the total sum of orders amount a salesman collects. Salesman should belong to the cities where any of the customer belongs.

SELECT s.name, s.city, total.total\_amt

FROM salesman AS s, ( SELECT s.salesman\_id, SUM(o.purch\_amt) AS total\_amt

FROM salesman AS s

JOIN orders AS o

ON s.salesman\_id = o.salesman\_id

GROUP BY s.salesman\_id ) AS total

WHERE s.salesman\_id = total.salesman\_id

AND s.city IN ( SELECT DISTINCT city

FROM customer )

1. Write a query to get all the information for those customers whose grade is not as the grade of customer who belongs to the city London.

SELECT \*

FROM customer

WHERE grade != ANY ( SELECT grade

FROM customer

WHERE city = 'London' )

1. Write a query to find all those customers whose grade are not as the grade, belongs to the city Paris.

SELECT \*

FROM customer

WHERE grade != ANY ( SELECT grade

FROM customer

WHERE city = 'Paris' )

1. Write a query to find all those customers who hold a different grade than any customer of the city Dallas.

SELECT \*

FROM customer

WHERE NOT grade = ANY ( SELECT grade

FROM customer

WHERE city = 'Dallas' )

1. Write a SQL query to find the average price of each manufacturer's products along with their name.

SELECT c.com\_name, AVG(m.pro\_price)

FROM item\_mast AS m, company\_mast AS c

WHERE m.pro\_com = c.com\_id

GROUP BY c.com\_name;

1. Write a SQL query to display the average price of the products which is more than or equal to 350 along with their names.

SELECT c.com\_name, AVG(m.pro\_price) AS "average"

FROM item\_mast AS m, company\_mast AS c

WHERE m.pro\_com = c.com\_id

GROUP BY c.com\_name

HAVING AVG(m.pro\_price) >= 350;

1. Write a SQL query to display the name of each company, price for their most expensive product along with their Name.

SELECT c.com\_name, m.pro\_name, m.pro\_price

FROM company\_mast AS c, item\_mast AS m

WHERE c.com\_id = m.pro\_com

AND m.pro\_price = ( SELECT MAX(i.pro\_price)

FROM item\_mast AS i

WHERE i.pro\_com = c.com\_id )

1. Write a query in SQL to find all the details of employees whose last name is Gabriel or Dosio.

SELECT \*

FROM emp\_details

WHERE emp\_lname IN ('Gabriel', 'Dosio')

1. Write a query in SQL to display all the details of employees who works in department 89 or 63.

SELECT \*

FROM emp\_details AS e, emp\_department AS d

WHERE e.emp\_dept = d.dpt\_code

AND d.dpt\_code in (89, 63)

1. Write a query in SQL to display the first name and last name of employees working for the department which allotment amount is more than Rs.50000.

SELECT \*

FROM emp\_details

WHERE emp\_dept IN ( SELECT dpt\_code

FROM emp\_department

WHERE dpt\_allotment > 50000 )

1. Write a query in SQL to find the departments which sanction amount is larger than the average sanction amount of all the departments.

SELECT \*

FROM emp\_department

WHERE dpt\_allotment > ( SELECT AVG(dpt\_allotment)

FROM emp\_department )

1. Write a query in SQL to find the names of departments with more than two employees are working.

SELECT \*

FROM emp\_department

WHERE dpt\_code IN ( SELECT emp\_dept

FROM emp\_details

GROUP BY emp\_dept

HAVING COUNT(emp\_idno) > 2 )

1. Write a query in SQL to find the first name and last name of employees working for departments which sanction amount is second lowest.

SELECT emp\_lname, emp\_fname

FROM emp\_details

WHERE emp\_dept IN ( SELECT dpt\_code

FROM emp\_department

WHERE dpt\_allotment = ( SELECT MIN(dpt\_allotment)

FROM emp\_department

WHERE dpt\_allotment > ( SELECT MIN(dpt\_allotment)

FROM emp\_department )))