**W3 SQL JOINS Exercises**

1. Write a SQL statement to prepare a list with salesman name, customer name and their cities for the salesmen and customer who belongs to the same city.

SELECT s.name, c.cust\_name, s.city

FROM salesman AS s, customer AS c

WHERE s.city = c.city

1. Write a SQL statement to make a list with order no, purchase amount, customer name and their cities for those orders which order amount between 500 and 2000.

SELECT o.ord\_no, o.purch\_amt, c.cust\_name, c.city

FROM orders AS o

JOIN customer AS c

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

WHERE o.purch\_amt BETWEEN 500 AND 2000

1. Write a SQL statement to know which salesman are working for which customer.

SELECT c.cust\_name, s.name

FROM customer AS c

INNER JOIN salesman AS s

ON c.salesman\_id = s.salesman\_id;

1. Write a SQL statement to find the list of customers who appointed a salesman for their jobs who gets a commission from the company is more than 12%.

SELECT c.cust\_name, s.name

FROM customer AS c

JOIN salesman AS s

ON c.salesman\_id = s.salesman\_id

WHERE s.commission > 0.12

1. Write a SQL statement to find the list of customers who appointed a salesman for their jobs who does not live in the same city where their customer lives, and gets a commission is above 12%.

SELECT c.cust\_name, s.name

FROM customer AS c

JOIN salesman AS s

ON c.salesman\_id = s.salesman\_id

WHERE c.city != s.city

AND s.commission > 0.12

1. Write a SQL statement to find the details of an order i.e. order number, order date, amount of order, which customer gives the order and which salesman works for that customer and how much commission he gets for an order.

SELECT o.ord\_no, o.ord\_date, o.purch\_amt, c.cust\_name, s.name, s.commission

FROM orders AS o

JOIN customer AS c

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

JOIN salesman AS s

ON c.salesman\_id = s.salesman\_id

1. Write a SQL statement to make a join on the tables salesman, customer and orders in such a form that the same column of each table will appear once and only the relational rows will come.

SELECT \*

FROM orders

NATURAL JOIN customer

NATURAL JOIN salesman

1. Write a SQL statement to make a list in ascending order for the customer who works either through a salesman or by own.

SELECT c.cust\_name, c.city, c.grade, s.name

FROM customer AS c

LEFT JOIN salesman AS s

ON c.salesman\_id = s.salesman\_id

ORDER BY c.customer\_id

1. Write a SQL statement to make a list in ascending order for the customer who holds a grade less than 300 and works either through a salesman or by own.

SELECT c.cust\_name, c.city, c.grade, s.name

FROM customer AS c

LEFT JOIN salesman AS s

ON c.salesman\_id = s.salesman\_id

WHERE c.grade < 300

ORDER BY c.customer\_id

1. Write a SQL statement to make a report with customer name, city, order number, order date, and order amount in ascending order according to the order date to find that either any of the existing customers have placed no order or placed one or more orders.

SELECT c.cust\_name, c.city, o.ord\_no, o.ord\_date, o.purch\_amt

FROM customer AS c

LEFT JOIN orders AS o

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

ORDER BY o.ord\_date;

1. Write a SQL statement to make a report with customer name, city, order number, order date, order amount salesman name and commission to find that either any of the existing customers have placed no order or placed one or more orders by their salesman or by own.

SELECT c.cust\_name, c.city, o.ord\_no, o.ord\_date, o.purch\_amt,

s.name, s.commission

FROM customer AS c

LEFT JOIN orders AS o

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

LEFT JOIN salesman AS s

ON c.salesman\_id = s.salesman\_id

1. Write a SQL statement to make a list in ascending order for the salesmen who works either for one or more customer or not yet join under any of the customers.

SELECT s.name, s.city

FROM salesman AS s

LEFT JOIN customer AS c

ON s.salesman\_id = c.salesman\_id

ORDER BY s.salesman\_id

1. Write a SQL statement to make a list for the salesmen who works either for one or more customer or not yet join under any of the customers who placed either one or more orders or no order to their supplier.

SELECT s.name, s.city

FROM customer AS c

RIGHT JOIN salesman AS s

ON c.salesman\_id = s.salesman\_id

RIGHT JOIN orders AS o

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

1. Write a SQL statement to make a list for the salesmen who either work for one or more customers or yet to join any of the customer. The customer may have placed, either one or more orders on or above order amount 2000 and must have a grade, or he may not have placed any order to the associated supplier.

SELECT c.cust\_name, s.name

FROM customer AS c

RIGHT JOIN salesman AS s

ON c.salesman\_id = s.salesman\_id

LEFT JOIN orders AS o

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

WHERE o.purch\_amt >= 2000

AND c.grade IS NOT NULL

1. Write a SQL statement to make a report with customer name, city, order no. order date, purchase amount for those customers from the existing list who placed one or more orders or which order(s) have been placed by the customer who is not on the list.

SELECT c.cust\_name, c.city, o.ord\_no, o.ord\_date, o.purch\_amt

FROM customer AS c

FULL JOIN orders AS o

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

1. Write a SQL statement to make a report with customer name, city, order no. order date, purchase amount for only those customers on the list who must have a grade and placed one or more orders or which order(s) have been placed by the customer who is neither in the list not have a grade.

SELECT c.cust\_name, c.city, o.ord\_no, o.ord\_date, o.purch\_amt

FROM customer AS c

FULL JOIN orders AS o

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

WHERE c.grade IS NOT NULL

1. Write a SQL statement to make a cartesian product between salesman and customer i.e. each salesman will appear for all customer and vice versa.

SELECT \*

FROM customer

CROSS JOIN salesman

1. Write a SQL statement to make a cartesian product between salesman and customer i.e. each salesman will appear for all customer and vice versa for that salesman who belongs to a city.

SELECT \*

FROM customer

CROSS JOIN salesman

WHERE salesman.city IS NOT NULL

1. Write a SQL statement to make a cartesian product between salesman and customer i.e. each salesman will appear for all customer and vice versa for those salesmen who belongs to a city and the customers who must have a grade.

SELECT \*

FROM customer

CROSS JOIN salesman

WHERE salesman.city IS NOT NULL

AND customer.grade IS NOT NULL

1. Write a SQL statement to make a cartesian product between salesman and customer i.e. each salesman will appear for all customer and vice versa for those salesmen who must belong a city which is not the same as his customer and the customers should have an own grade.

SELECT \*

FROM customer

CROSS JOIN salesman

WHERE salesman.city IS NOT NULL

AND customer.grade IS NOT NULL

AND salesman.city != customer.city

1. Write a SQL query to display all the data from the item\_mast, including all the data for each item's producer company.

SELECT \*

FROM item\_mast AS m

JOIN company\_mast AS c

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

1. Write a SQL query to display the item name, price, and company name of all the products.

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

FROM item\_mast AS m

JOIN company\_mast AS c

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

1. Write a SQL query to display the average price of items of each company, showing the name of the company.

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

FROM item\_mast AS m

JOIN company\_mast AS c

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

GROUP BY c.com\_name

1. Write a SQL query to display the names of the company whose products have an average price larger than or equal to Rs. 350.

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

FROM company\_mast AS c

JOIN item\_mast AS m

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

GROUP BY c.com\_name

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

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

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

FROM company\_mast AS c

JOIN item\_mast AS m

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

AND m.pro\_price = (

SELECT MAX(m.pro\_price)

FROM item\_mast AS m

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

1. Write a query in SQL to display all the data of employees including their department.

SELECT d.\*, e.\*

FROM emp\_details AS e

JOIN emp\_department AS d

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

1. Write a query in SQL to display the first name and last name of each employee, along with the name and sanction amount for their department.

SELECT e.emp\_fname, e.emp\_lname, d.dpt\_name, d.dpt\_allotment

FROM emp\_details AS e

JOIN emp\_department AS d

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

1. Write a query in SQL to find the first name and last name of employees working for departments with a budget more than Rs. 50000.

SELECT e.emp\_fname, e.emp\_lname, d.dpt\_name

FROM emp\_details AS e

JOIN emp\_department AS d

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

WHERE d.dpt\_allotment > 50000

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

SELECT d.dpt\_name, COUNT(e.emp\_idno)

FROM emp\_department AS d

JOIN emp\_details AS e

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

GROUP BY d.dpt\_name

HAVING COUNT(e.emp\_idno) > 2