**Activity 1:**

-- Create salesman table

CREATE TABLE salesman (

salesman\_id int,

salesman\_name varchar2(32),

salesman\_city varchar2(32),

commission int

);

**Activity 2:**

-- Insert one row

INSERT INTO salesman VALUES(5002, 'Nail Knite', 'Paris', 13);

-- Insert multiple rows at once

INSERT ALL

INTO salesman VALUES(5005, 'Pit Alex', 'London', 11)

INTO salesman VALUES(5006, 'McLyon', 'Paris', 14)

INTO salesman VALUES(5007, 'Paul Adam', 'Rome', 13)

INTO salesman VALUES(5003, 'Lauson Hen', 'San Jose', 12)

SELECT 1 FROM DUAL;

-- View data from all columns

SELECT \* FROM salesman;

**Activity 3:**

-- Show data from the salesman\_id and city columns

SELECT salesman\_id, salesman\_city FROM salesman;

-- Show data of salesman from Paris

SELECT \* FROM salesman WHERE salesman\_city='Paris';

-- Show salesman\_id and commission of Paul Adam

SELECT salesman\_id, commission FROM salesman WHERE salesman\_name='Paul Adam';

**Activity 4:**

-- Add the grade column

ALTER TABLE salesman ADD grade int;

-- Update the values in the grade column

UPDATE salesman SET grade=100;

-- Display data

SELECT \* FROM salesman;

**Activity 5:**

-- Update the grade score of salesmen from Rome to 200.

UPDATE salesman SET grade=200 WHERE salesman\_city='Rome';

-- Update the grade score of James Hoog to 300.

UPDATE salesman SET grade=300 WHERE salesman\_name='James Hoog';

-- Update the name McLyon to Pierre.

UPDATE salesman SET salesman\_name='Pierre' WHERE salesman\_name='McLyon';

-- Display modified data

SELECT \* FROM salesman;

**Activity 6:**

-- Get all salesman ids without any repeated values

select distinct salesman\_id from orders;

-- Display the order number ordered by date in ascending order

select order\_no, order\_date from orders order by order\_date;

-- Display the order number ordered by purchase amount in descending order

select order\_no, purchase\_amount from orders order by purchase\_amount DESC;

-- Display the full data of orders that have purchase amount less than 500.

select \* from orders where purchase\_amount < 500;

-- Display the full data of orders that have purchase amount between 1000 and 2000.

select \* from orders where purchase\_amount between 1000 and 2000;

**Activity 7:**

-- Write an SQL statement to find the total purchase amount of all orders.

select SUM(purchase\_amount) AS "Total sum" from orders;

-- Write an SQL statement to find the average purchase amount of all orders.

select AVG(purchase\_amount) AS "Average" from orders;

-- Write an SQL statement to get the maximum purchase amount of all the orders.

select MAX(purchase\_amount) AS "Maximum" from orders;

-- Write an SQL statement to get the minimum purchase amount of all the orders.

select MIN(purchase\_amount) AS "Minumum" from orders;

-- Write an SQL statement to find the number of salesmen listed in the table.

select COUNT(distinct salesman\_id) AS "Total count" from orders;

**Activity 8:**

-- Write an SQL statement to find the highest purchase amount ordered by the each customer with their ID and highest purchase amount.

SELECT customer\_id, MAX(purchase\_amount) AS "Max Amount" FROM orders GROUP BY customer\_id;

-- Write an SQL statement to find the highest purchase amount on '2012-08-17' for each salesman with their ID.

SELECT salesman\_id, order\_date, MAX(purchase\_amount) AS "Max Amount" FROM orders

WHERE order\_date=To\_DATE('2012/08/17', 'YYYY/MM/DD') GROUP BY salesman\_id, order\_date;

-- Write an SQL statement to find the highest purchase amount with their ID and order date, for only those customers who have a higher purchase amount within the list 2030, 3450, 5760, and 6000.

SELECT customer\_id, order\_date, MAX(purchase\_amount) AS "Max Amount" FROM orders

GROUP BY customer\_id, order\_date

HAVING MAX(purchase\_amount) IN(2030, 3450, 5760, 6000);

**Activity 9:**

-- Write an SQL statement to know which salesman are working for which customer.

SELECT a.customer\_name AS "Customer Name", a.city, b.name AS "Salesman", b.commission FROM customers a

INNER JOIN salesman b ON a.salesman\_id=b.salesman\_id;

-- Write an 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

SELECT a.customer\_name, a.city, a.grade, b.name AS "Salesman", b.city FROM customers a

LEFT OUTER JOIN salesman b ON a.salesman\_id=b.salesman\_id WHERE a.grade<300

ORDER BY a.customer\_id;

**Activity 10:**

-- 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);

-- Write a query to find all orders attributed to a salesman in New York.

SELECT \* FROM orders

WHERE salesman\_id IN (SELECT salesman\_id FROM salesman WHERE city='New York');

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

SELECT grade, COUNT(\*) FROM customers

GROUP BY grade HAVING grade>(SELECT AVG(grade) FROM customers WHERE city='New York');

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

SELECT order\_no, purchase\_amount, order\_date, salesman\_id FROM orders

WHERE salesman\_id IN( SELECT salesman\_id FROM salesman

WHERE commission=( SELECT MAX(commission) FROM salesman));

**Activity 11:**

-- Write a query that produces the name and number of each salesman and each customer with more than one current order. Put the results in alphabetical order

SELECT customer\_id, customer\_name FROM customers a

WHERE 1<(SELECT COUNT(\*) FROM orders b WHERE a.customer\_id = b.customer\_id)

UNION

SELECT salesman\_id, name FROM salesman a

WHERE 1<(SELECT COUNT(\*) FROM orders b WHERE a.salesman\_id = b.salesman\_id)

ORDER BY customer\_name;

-- Write a query to make a report of which salesman produce the largest and smallest orders on each date.

SELECT a.salesman\_id, name, order\_no, 'highest on', order\_date FROM salesman a, orders b

WHERE a.salesman\_id=b.salesman\_id

AND b.purchase\_amount=(SELECT MAX(purchase\_amount) FROM orders c WHERE c.order\_date = b.order\_date)

UNION

SELECT a.salesman\_id, name, order\_no, 'lowest on', order\_date FROM salesman a, orders b

WHERE a.salesman\_id=b.salesman\_id

AND b.purchase\_amount=(SELECT MIN(purchase\_amount) FROM orders c WHERE c.order\_date = b.order\_date);