SUBQUERY-1

- DEFINITION
- GUIDELINES
- TYPES
  - Single row
  - Multiple row
  - Nested
  - Correlated
- SUBQUERY WITH
  - Exists and not exists
  - Insertion
  - Update
  - Delete
- LAB EXERCISE

Subquery/Inner query

- a query within another SQL query
- It can be used in a SELECT, INSERT, DELETE, or UPDATE statement
- It perform the following tasks:
  - Compare an expression to the result of the query.
  - Determine if an expression is included in the results of the query.
  - Check whether the query selects any rows.
  - The subquery executes once before the main query (outer query) executes.
  - The main query use the subquery result.

Subqueries: Guidelines

- must be enclosed in parentheses.
- must be placed on the right side of the comparison operator
- An ORDER BY command cannot be used in a subquery.
- Subqueries that return more than one row can only be used with multiple value operators such as the IN operator.
- The BETWEEN operator cannot be used with a subquery. However, the BETWEEN operator can be used within the subquery.

Types

Single row subquery : Returns zero or one row.

Multiple row subquery : Returns one or more rows.

Correlated subqueries : Reference one or more columns in the outer SQL statement. The subquery is known as a correlated subquery because the subquery is related to the outer SQL statement.

Nested subqueries : Subqueries are placed within another subquery.

```
SQL>  create table oldcafeteria(ord_no number,ord_amount number, advaance_amount number,
  2  agent_code  varchar2(12), customer_code varchar2(15));

Table created.

SQL>  insert into oldcafeteria values(1,200,2000,'oc001','cus001');

1 row created.

SQL> insert into oldcafeteria values(2,300,3000,'oc002','cus002');

1 row created.

SQL> insert into oldcafeteria values(3,3000,300000,'oc007','cus009');

1 row created.

SQL> insert into oldcafeteria values(4,7000,20000,'oc009','cus010');

1 row created.

SQL> insert into oldcafeteria values(5,600,5000,'oc010','cus011');

1 row created.
```

```
SQL> select * from oldcafeteria;

    ORD_NO ORD_AMOUNT ADVAANCE_AMOUNT AGENT_CODE   CUSTOMER_CODE
---------- ---------- --------------- ------------ --------------
         1        200            2000 oc001        cus001
         2        300            3000 oc002        cus002
         3       3000          300000 oc007        cus009
         4       7000           20000 oc009        cus010
         5        600            5000 oc010        cus011
```

Single Row Subqueries

A single row subquery returns zero or one row to the outer SQL statement.

```
SQL> select ord_no, ord_amount from oldcafeteria where
  2  ord_no=(select ord_no from oldcafeteria where agent_code='oc001');

    ORD_NO ORD_AMOUNT
---------- ----------
         1        200
```

Multiple Row Subqueries

- returns one or more rows to the outer SQL statement
- use the IN, ANY, or ALL operator in outer query to handle a subquery that returns multiple rows

```
SQL>  select ord_no, ord_amount from oldcafeteria where ord_no in
  2   (select ord_no from oldcafeteria where(advaance_amount<=20000));

    ORD_NO ORD_AMOUNT
---------- ----------
         1        200
         2        300
         4       7000
         5        600
```

Nested subqueries

```
SQL> select ord_no from oldcafeteria where exists(select *
  2  from oldcafeteria where exists
  3  (select * from oldcafeteria where ord_amount<20000));

    ORD_NO
----------
         1
         2
         3
         4
         5
```

subquery with EXISTS and NOT EXISTS

subquery returns a Boolean value of True or false

TRUE: Subquery return any rows
False : Doesn't return any rows

SELECT * FROM table_name **WHERE EXISTS**(subquery);

```
SQL> select ord_no from oldcafeteria where exists(select advaance_amount
  2  from oldcafeteria where agent_code like '%7%');

    ORD_NO
----------
         1
         2
         3
         4
         5

SQL>  select ord_no from oldcafeteria where not exists(select advaance_amount
  2  from oldcafeteria where agent_code like '%7%');

no rows selected
```

Inserting records using subqueries

INSERT INTO table_name [(column1 [, column2])] SELECT [*|column1 [, column2] FROM table1 [, table2] [WHERE VALUE OPERATOR];

Old cafeteria and newcafeteria has same attributes

```
SQL>  create table newcafeteria(ord_no number,ord_amount number, advaance_amount
  2  number,agent_code  varchar2(12), customer_code varchar2(15));

Table created.

SQL> insert into newcafeteria select * from oldcafeteria where advaance_amount
  2  in(2000,30000);

1 row created.

SQL> select * from newcafeteria;

    ORD_NO ORD_AMOUNT ADVAANCE_AMOUNT AGENT_CODE   CUSTOMER_CODE
---------- ---------- --------------- ------------ ---------------
         1        200            2000 oc001        cus001
```

Subqueries with UPDATE statement

UPDATE table SET column_name = new_value [WHERE OPERATOR [VALUE] (SELECT COLUMN_NAME FROM TABLE_NAME) [WHERE)]

```
SQL> update newcafeteria set ord_no=3 where advaance_amount-ord_amount
  2  >(select min(ord_amount) from oldcafeteria);

1 row updated.

SQL> select * from newcafeteria;

    ORD_NO ORD_AMOUNT ADVAANCE_AMOUNT AGENT_CODE   CUSTOMER_CODE
---------- ---------- --------------- ------------ --------------
         3        200            2000 oc001        cus001
```

Update the rows of newcafeteria whose (advance_amount-ord_amount) is greater than minimum ord_amount from old cafeteria

Result as table

```
SQL> create table newcafe as select * from oldcafeteria;

Table created.

SQL> select * from newcafe;

    ORD_NO ORD_AMOUNT ADVAANCE_AMOUNT AGENT_CODE   CUSTOMER_CODE
---------- ---------- --------------- ------------ ---------------
         1        200            2000 oc001        cus001
         2        300            3000 oc002        cus002
         3       3000          300000 oc007        cus009
         4       7000           20000 oc009        cus010
         5        600            5000 oc010        cus011
```

```
SQL>  create table new1 as select ord_no from oldcafeteria where advaance_amount<=30000;

Table created.

SQL> select * from new1;

    ORD_NO
----------
         1
         2
         4
         5
```

Subqueries with DELETE statement

DELETE FROM TABLE_NAME [WHERE OPERATOR [VALUE] (SELECT COLUMN_NAME FROM TABLE_NAME) [WHERE)]

delete those orders from 'newcafeteria' table which advance_amount are less than the maximum advance_amount of ‘oldcafeteria' table

```
SQL> delete from newcafeteria where advaance_amount<(select
  2  max(advaance_amount) from oldcafeteria);

1 row deleted.

SQL> select * from newcafeteria;
```

LAB EXERCISE

Ord_num	Ord_amount	Advance_amount	Ord_date	Cust_code	Agent_code	Description
004	200	3000	15-aug-2020	C004	Ac001	Masala kulcha
007	600	5000	17-sept-2020	C006	Ac003	Biriyani
008	700	100	19-feb-2019	C007	Ac005	
009	10000	600	21-march-2010	C009	Ac008	Masala dosa
010	20	600	21-april - 2012	C006	Ac005	

Table: orders

Agent_code	Agent_name	Working_area	commision	Phone_no	country
Ac001	Ramesh	Bangalore	.15	0331234567	India
Ac002	Dinesh	Bangalore	.25	0331234568	
Ac003	Suresh	Mumbai	.35	0331234569	London
Ac004	Kamlesh	New jersey	.68	0331234564	
Ac005	Kartik	Chennai	.73	0331234563	India

Table: Agent

1. Consider the following table Agent(AGENT_CODE, AGENT_NAME, WORKING_AREA ,|COMMISSION , PHONE_NO, COUNTRY) and Orders(ORD_NUM, ORD_AMOUNT, ADVANCE_AMOUNT, ORD_DATE,CUST_CODE,AGENT_CODE,ORD_DESCRIPTION)
   a. Find ord_num, ord_amount, ord_date, cust_code and agent_code from the table Orders working_area of Agent table must be Bangalore.
   b. Retrive ord_num, ord_amount, cust_code and agent_code from the table orders where the agent_code of orders table must be the same agent_code of agents table and agent_name of agents table must be Ramesh.

LAB EXERCISE

Table: employees

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	manager_id	department_id
700	Hasmukh	Patel	hp@gmail.com	7003216160	15-aug-2020	Hp003	7000		90
800	Kamlesh	Paul	kp@gmail.com	7003216170	17-feb-2020	Kp 004	8000	506	90
900	Dinesh	Gandhi	dp@yahoo	9136278563	19-march-2101	Dg006	20000	508	80
701	Suresh	Modi	sm@dg.com	9187653294	20-april-2015	Sm009	15000		80

2. Consider the table employees(employee_id, first_name, last_name, email, phone_number, hire_date, job_id, salary, manager_id, department_id)
   a. Display the employee_id, manager_id, first_name and last_name of those employees who manage other employees.
   b. Display the employee_id, manager_id, first_name and last_name of those employees who have no manager status

Lab Exercise

Salesman_id	Name	City	commission
si123@06	Lakshmi	Kolkata	.5
si123@09	Ganesh	London	.6
si123@90	Dinesh	London	.3
si123@10	Joseph	Chennai	.6
si123@19	Mahesh	Hyderabad	.65
si123@26	Paul Adam	London	.1
si123@67	Rahul	Delhi	.4

Table: salesman

ord_no	Purch_amt	Ord_date	Customer_id	Salesman_id
123	600	20-aug-2010	003cd	si123@19
576	750	20-feb-2018	004cd	si123@19
579	800	20-may-20120	004cd	si123@26
600	60000	20-jan-2021	006cd	si123@10
700	745	26-jan-2021	007cd	si123@09
800	860	29-jan-2019	007cd	si123@26

Table: orders

3. Consider the tables salesman(salesman_id, name ,city ,commission) and Orders(ord_no, purch_amt, ord_date, customer_id, salesman_id)
   a. Display all the orders from the orders table issued by the salesman 'Paul Adam'.
   b. Display all the orders for the salesman who belongs to the city London