SE COMP - B	Roll number : 8942
Experiment no. : 7	Date of Implementation : 15/04/2021

Aim : To implement Nested Sub-queries in SQL

Tool Used : PostgreSQL/ Mysql

Related Course outcome : At the end of the course, Students will be able to Use SQL : Standard language of relational database

Rubrics for assessment of Experiment:

Indicator	Poor	Average	Good
Timeliness • Maintains assignment deadline (3)	Assignment not done (0)	One or More than One week late (1-2)	Maintains deadline (3)
Completeness and neatness • Complete all parts of assignment(3)	N/A	< 80% complete (1-2)	100% complete (3)
Originality • Extent of plagiarism(2)	Copied it from someone else(0)	At least few questions have been done without copying(1)	Assignment has been solved completely without copying (2)
Knowledge • In depth knowledge of the assignment(2)	Unable to answer 2 questions(0)	Unable to answer 1 question (1)	Able to answer 2 questions (2)

Assessment Marks :

Timeliness	
Completeness and neatness	
Originality	
Knowledge	
Total	

Total : (Out of 10)

Teacher's Sign :	
EXPERIMENT 7	Nested subqueries in SQL
Aim	To implement nested sub-queries in SQL
Tools	PostgreSQL/Mysql
Procedure	Use the tables created in the previous experiments and Perform the following queries using nested sub-queries. **Client_master** (client_no, name, address, city, pincode, state, bal_due) **Product_master** (product_no, description, profit_percentage, unit_measure, qty_on_hand, reorder_level, sell_price, cost_price) **Sales_order**(order_no, order_date, client_no, dely_Addr, salesman_no, dely_type, billed_yn, dely_date, order_status) **Sales_order_details**(order_no, product_no, qty_ordered, qty_disp, product_rate) 1. Find the product no. and description of non-moving products i.e. products not being sold. 2. Find the customer name, address for the client who has placed order no 'O191' 3. Find the clients names who have placed orders before the month of May'96 4. Find out if the product '1.44 Drive' has been ordered by any client and print the client_no, name to whom it was sold 5. Find the names of clients who have placed orders worth Rs. 10000 or more 6. Retrieve all the orders placed by a client named 'Rahul Desai' from the sales_order table. 7. Retrieve name, address, city of all the clients who have placed an order through salesman no 's001'.
Post Lab Questions:	1. What is incremental Update? 2. Explain is use of **on delete cascade** and **on update cascade** with suitable example?

1. Creating a database as exp_7

Query:

create database exp_7;

Create following table:

We have already created three tables:

client_master, product_master and sales_order in previous experiments.

A.] Client_master

1.] Creating table

Query:

```
create table client_master(
Client_no varchar(6), PRIMARY KEY(Client_no),
Name varchar(20) not null,
Address varchar(30),
City varchar(15),
Pincode numeric(8),
State varchar(15),
Bal_due numeric(10,2));
```

Screenshot:

2.] Inserting values

Query:

Insert into client_master value('C0001', 'Praisy', 'Mira Road', 'Mumbai', 401107, 'Maharashtra', 400);

Insert into client_master value('C0002', 'Ashley', 'Ashok Nagar', 'Chennai', 401107, 'Tamil Nadu', 400);

Insert into client_master value('C0003', 'Prejith', 'Adyar', 'Chennai', 401107, 'Tamil Nadu', 400);

Insert into client_master value('C0004', 'Elwin', 'Bandra', 'Mumbai', 401107, 'Maharashtra', 400);
Insert into client_master value('C0005', 'Elaine', 'Pallom', 'Kottayam', 401107, 'Kerala', 400);
Insert into client_master value('C0006', 'Jerry', 'Borivali', 'Mumbai', 401107, 'Maharashtra', 400);
Insert into client_master value('C0007', 'Gini', 'Adyar', 'Chennai', 401107, 'Tamil Nadu', 400);
Insert into client_master value('C0008', 'Rahul Desai', 'Bandra', 'Mumbai', 401107, 'Maharashtra', 400);
Insert into client_master value('C0009', 'Ram', 'Adyar', 'Madras', 401107, 'Tamil Nadu', 400);
Insert into client_master value('C0010', 'Ivan Bayross', 'Bandra', 'Mumbai', 401107, 'Maharashtra', 400);

Screenshot:

Result Grid | Filter Rows: | Edit: | Export/Import:

Client_no	Name	Address	City	Pincode	State	Bal_due
C0001	Praisy	Mira Road	Mumbai	401107	Maharashtra	400.00
C0002	Ashley	Ashok Nagar	Chennai	401107	Tamil Nadu	400.00
C0003	Prejith	Adyar	Chennai	401107	Tamil Nadu	400.00
C0004	Elwin	Bandra	Mumbai	401107	Maharashtra	400.00
C0005	Elaine	Pallom	Kottayam	401107	Kerala	400.00
C0006	Jerry	Borivali	Mumbai	401107	Maharashtra	400.00
C0007	Gini	Adyar	Chennai	401107	Tamil Nadu	400.00
C0008	Rahul Desai	Bandra	Mumbai	401107	Maharashtra	400.00
C0009	Ram	Adyar	Madras	401107	Tamil Nadu	400.00
C0010	Ivan Bayross	Bandra	Mumbai	401107	Maharashtra	400.00
NULL	NULL	NULL	NULL	NULL	NULL	NULL

B.]Product_master

1.] Creating Table

Query:

create table Product_master(
product_no varchar(6), PRIMARY KEY(product_no),
description varchar(15) not null,
profit_percent numeric(4,2),

unit_measure varchar(10),
qty_on_hand numeric(8),
reorder_level numeric(8),
sell_price numeric(8,2),
cost_price numeric(8,2));

Screenshot:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

product_no	description	profit_percent	unit_measure	qty_on_hand	reorder_level	sell_price	cost_price
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2.] Inserting values

Query:

Insert into product_master value('P0001', 'CD Drive', '10', '10GB',20, 30, 80, 50);
Insert into product_master value('P0002', 'Mouse', '20', '4GB',50, 60,100, 60);
Insert into product_master value('P0003', '1.44 Drive', '1', '146',70, 50, 200, 150);
Insert into product_master value('P0004', 'Keyborad', '50', '105 ',50, 45, 300, 150);
Insert into product_master value('P0005', 'HDD', '30', '45GB',20, 30, 100, 50);
Insert into product_master value('P0006', 'CPU', '20', '3GHz" ',80, 90, 350, 200);

Screenshot:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

product_no	description	profit_percent	unit_measure	qty_on_hand	reorder_level	sell_price	cost_price
P0001	CD Drive	10.00	10GB	20	30	80.00	50.00
P0002	Mouse	20.00	4GB	50	60	100.00	60.00
P0003	1.44 Drive	1.00	146	70	50	200.00	150.00
P0004	Keyborad	50.00	105	50	45	300.00	150.00
P0005	HDD	30.00	45GB	20	30	100.00	50.00
P0006	CPU	20.00	3GHz"	80	90	350.00	200.00
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

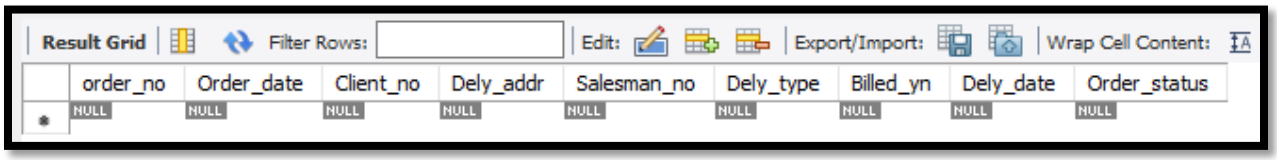
C.] Sales_order

1.] Creating table

Query:

```
create table sales_order(
order_no varchar(6), PRIMARY KEY(order_no),
Order_date date NOT NULL,
Client_no varchar(6) NOT NULL,
Dely_addr varchar(25),
Salesman_no varchar(6),
Dely_type char(1),
Billed_yn char(1),
Dely_date date,
Order_status varchar(10));
```

Screenshot:



order_no	Order_date	Client_no	Dely_addr	Salesman_no	Dely_type	Billed_yn	Dely_date	Order_status
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2.] Inserting values

Query:

```
Insert into sales_order value('O190','2021-02-26', 'C0001', 'Mumbai', 'S001', 's',
'y', '2021-01-1', 'delivered');
Insert into sales_order value('O191','2021-02-21', 'C0002', 'Chennai', 'S002', 's',
'y', '2021-02-15', 'shipped');
Insert into sales_order value('O192','2021-02-02', 'C0003', 'Chennai', 'S003', 'r',
'n', '2021-03-09', 'delivered');
Insert into sales_order value('O193','2021-02-06', 'C0004', 'Mumbai', 'S001', 's',
'y', '2021-04-13', 'delivered');
Insert into sales_order value('O194','2021-02-15', 'C0005', 'Kottayam', 'S005',
'r', 'n', '2021-05-2', 'shipped');
Insert into sales_order value('O195','2021-02-10', 'C0006', 'Mumbai', 'S001', 's',
'y', '2021-06-16', 'delivered');
```

Insert into sales_order value('O196','2021-01-10', 'C0007', 'Chennai', 'S004', 's', 'y', '2021-07-18', 'delivered');
Insert into sales_order value('O197','2021-01-10', 'C0008', 'Mumbai', 'S001', 's', 'y', '2021-08-14', 'shipped');
Insert into sales_order value('O198','2021-01-10', 'C0009', 'Madras', 'S002', 's', 'y', '2021-09-11', 'shipped');
Insert into sales_order value('O199','2021-01-10', 'C0010', 'Mumbai', 'S004', 's', 'y', '2021-10-19', 'delivered');

Screenshot:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

order_no	Order_date	Client_no	Dely_addr	Salesman_no	Dely_type	Billed_yn	Dely_date	Order_status
O190	2021-02-26	C0001	Mumbai	S001	s	y	2021-01-01	delivered
O191	2021-02-21	C0002	Chennai	S002	s	y	2021-02-15	shipped
O192	2021-02-02	C0003	Chennai	S003	r	n	2021-03-09	delivered
O193	2021-02-06	C0004	Mumbai	S001	s	y	2021-04-13	delivered
O194	2021-02-15	C0005	Kottayam	S005	r	n	2021-05-02	shipped
O195	2021-02-10	C0006	Mumbai	S001	s	y	2021-06-16	delivered
O196	2021-01-10	C0007	Chennai	S004	s	y	2021-07-18	delivered
O197	2021-01-10	C0008	Mumbai	S001	s	y	2021-08-14	shipped
O198	2021-01-10	C0009	Madras	S002	s	y	2021-09-11	shipped
O199	2021-01-10	C0010	Mumbai	S004	s	y	2021-10-19	delivered
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

D.] Sales_order_details

1.] Creating Table

Query:

Create table sales_order_details(
order_no varchar(6) references sales_order,
product_no char(6) references product_master,
qtyordered numeric(8),
qtydisp numeric(8),
productrate numeric(10,2));

Screenshot:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

order_no	product_no	qtyordered	qtydisp	productrate

2.] Inserting values

Query:

Insert into sales_order_details value('O190', 'P0001', 10, 5, 10000);
Insert into sales_order_details value('O191', 'P0002', 23, 20, 2000);
Insert into sales_order_details value('O192', 'P0003', 4, 3, 11000);
Insert into sales_order_details value('O193', 'P0003', 2, 2, 5000);
Insert into sales_order_details value('O194', 'P0004', 10, 2, 15000);
Insert into sales_order_details value('O195', 'P0005', 18, 15, 5500);
Insert into sales_order_details value('O196', 'P0006', 10, 0, 20500);

Screenshot:

Result Grid | Filter Rows: | Export:

order_no	product_no	qtyordered	qtydisp	productrate
O190	P0001	10	5	10000.00
O191	P0002	23	20	2000.00
O192	P0003	4	3	11000.00
O193	P0003	2	2	5000.00
O194	P0004	10	2	15000.00
O195	P0005	18	15	5500.00
O196	P0006	10	0	20500.00

1. Find the product no. and description of non-moving products i.e. products not being sold.

Query:

select sales_order_details.product_no, description
from sales_order_details,product_master
where qtydisp = 0 and sales_order_details.product_no = product_master.product_no;

Screenshot:

Result Grid | Filter Rows:

product_no	description
P0006	CPU

2. Find the customer name, address for the client who has placed order no 'O191'

Query:

```
select order_no, name, Address, City, Pincode
from client_master,sales_order
where order_no = 'O191' and sales_order.client_no = client_master.client_no;
```

Screenshot:

Result Grid | Filter Rows: | Export:

order_no	name	Address	City	Pincode
O191	Ashley	Ashok Nagar	Chennai	401107

3. Find the clients names who have placed orders before the month of May'09

Query:

```
select name, order_date
from client_master,sales_order
where order_date < '2021-05-09' and sales_order.client_no =
client_master.client_no;
```

Screenshot:

Result Grid | Filter Rows:

name	order_date
Praisy	2021-02-26
Ashley	2021-02-21
Prejith	2021-02-02
Elwin	2021-02-06
Elaine	2021-02-15
Jerry	2021-02-10
Gini	2021-01-10
Rahul Desai	2021-01-10
Ram	2021-01-10
Ivan Bayross	2021-01-10

4. Find out if the product '1.44 Drive' has been ordered by any client and print the client_no, name to whom it was sold

Query:

```
select description,sales_order_details.order_no,client_master.client_no,name
from product_master, client_master,sales_order,sales_order_details
where description = '1.44 Drive'
and product_master.product_no = sales_order_details.product_no
and sales_order_details.order_no = sales_order.order_no
and sales_order.client_no = client_master.client_no;
```

Screenshot:

Result Grid | Filter Rows:

description	order_no	client_no	name
1.44 Drive	O192	C0003	Prejith
1.44 Drive	O193	C0004	Elwin

5. Find the names of clients who have placed orders worth Rs. 10000 or more

Query:

```
select name,product_master.product_no,productrate
from client_master,product_master,sales_order, sales_order_details
where productrate >= 10000
and product_master.product_no = sales_order_details.product_no
and sales_order_details.order_no = sales_order.order_no
and sales_order.client_no = client_master.client_no;
```

Screenshot:

Result Grid | Filter Rows:

name	product_no	productrate
Praisy	P0001	10000.00
Prejith	P0003	11000.00
Elaine	P0004	15000.00
Gini	P0006	20500.00

6. Retrieve all the orders placed by a client named 'Rahul Desai' from the sales_order table.

Query:

```
select *
from sales_order
where client_no=(select client_no from client_master where name='Rahul Desai');
```

Screenshot:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

order_no	Order_date	Client_no	Dely_addr	Salesman_no	Dely_type	Billed_yn	Dely_date	Order_status
O197	2021-01-10	C0008	Mumbai	S001	s	y	2021-08-14	shipped
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

7. Retrieve name, address, city of all the clients who have placed an order through salesman no 's001'.

Query:

```
select name, Address, City
from client_master,sales_order
where Salesman_no = 'S001'
and sales_order.client_no = client_master.client_no;
```

Screenshot:

Result Grid | Filter Rows:

name	Address	City
Praisy	Mira Road	Mumbai
Elwin	Bandra	Mumbai
Jerry	Borivali	Mumbai
Rahul Desai	Bandra	Mumbai

POSTLAB QUESTIONS:

1. **What is incremental Update?**

Ans: An Incremental update adds new records to a project data set from a source Hive table.

The DP CLI --incrementalUpdate flag (abbreviated as -incremental) performs a partial update of a project data set by selecting or adding new and modified records. The data set should be a project data set that is a full data set (i.e., is not a sample data set) and has been configured for incremental updates.

The Incremental update operation fetches a subset of the records in the source Hive table. The subset is determined by using a filtering predicate that specifies the Hive table column that holds the records and the value of the records to fetch. The records in the subset batch are ingested as follows:

- If a record is brand new (does not exist in the data set), it is added to the data set.

- If a record already exists in the data set but its content has been changed, it replaces the record in the data set.

The record identifier determines if a record already exists or is new.

Unlike a Refresh update, an Incremental update has these limitations:

- An Incremental update cannot make schema changes to the data set. This means that no attributes in the data set will be deleted or added.

- An Incremental update cannot use the --disableSearch flag. This means that the searchability of the data set cannot be changed.

2. **Explain is use of on delete cascade and on update cascade with suitable example?**

Ans:

- **ON DELETE CASCADE** means that if the parent record is deleted, any child records are also deleted. This is **not** a good idea in my opinion. You should keep track of all data that's ever been in a database, although this can be done using TRIGGERs. (However, see caveat in comments below).

- **ON UPDATE CASCADE** means that if the parent primary key is changed, the child value will also change to reflect that. Again in my opinion, not a great idea. If you're changing PRIMARY KEYs with any regularity (or even at all!), there is something wrong with your design. Again, see comments.

- **ON UPDATE CASCADE ON DELETE CASCADE** means that if you UPDATE **OR** DELETE the parent, the change is cascaded to the child. This is the equivalent of ANDing the outcomes of first two statements.

EXAMPLE:

ON DELETE CASCADE

```
CREATE TABLE child (

 id INT,
 parent_id INT,
 INDEX par_ind (parent_id),
 FOREIGN KEY (parent_id)
    REFERENCES parent(id)
    ON DELETE CASCADE
) ENGINE=INNODB;
```

ON UPDATE CASCADE

```
CREATE TABLE child (
 id INT, parent_id INT,
 INDEX par_ind (parent_id),
 FOREIGN KEY (parent_id)
    REFERENCES parent(id)
    ON UPDATE CASCADE
) ENGINE=INNODB;
```

ON UPDATE CASCADE ON DELETE CASCADE

```
CREATE TABLE child (
        id INT,  parent_id INT,
    INDEX par_ind (parent_id),
    FOREIGN KEY (parent_id)
        REFERENCES parent(id)
        ON UPDATE CASCADE ON DELETE CASCADE
 ) ENGINE=INNODB;
```