

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Programme | : | **M.Tech. (Integrated) Software Engineering** | Semester | : | **Fall 20-21** |
| Course | : | **Advanced Database Management Systems** | Code | : | **SWE2014** |
| Faculty | : | **Dr.Bhuvaneswari A** | Slot | : | **L27 + L28** |
| Date | : | **18-08-2020** | Marks | : | **10 Points** |

**Reg no: 17MIS1056**

**K.NAVEEN KUMAR**

**Ex.5 ORACLE -PARALLEL EXECUTION QUERIES**

**Create table for the following schema: (Use Exercise 3 Tables)**

**SALESMAN (Salesman\_id, Name, City, Commission)**

**CUSTOMER (Customer\_id, Cust\_Name, City, Grade, Salesman\_id)**

**ORDERS (Ord\_No, Purchase\_Amt, Ord\_Date, Customer\_id, Salesman\_id)**

Sample table: Salesman

salesman\_id | name | city | commission

-------------+------------+----------+------------

5001 | James Hoog | New York | 0.15

5002 | Nail Knite | Paris | 0.13

5005 | Pit Alex | London | 0.11

5006 | Mc Lyon | Paris | 0.14

5007 | Paul Adam | Rome | 0.13

5003 | Lauson Hen | San Jose | 0.12

Sample table: Customer

customer\_id | cust\_name | city | grade | salesman\_id

-------------+----------------+------------+-------+-------------

3002 | Nick Rimando | New York | 100 | 5001

3007 | Brad Davis | New York | 200 | 5001

3005 | Graham Zusi | California | 200 | 5002

3008 | Julian Green | London | 300 | 5002

3004 | Fabian Johnson | Paris | 300 | 5006

3009 | Geoff Cameron | Berlin | 100 | 5003

3003 | Jozy Altidor | Moscow | 200 | 5007

3001 | Brad Guzan | London | | 5005

Sample table: orders

ord\_no purchase\_amt ord\_date customer\_id salesman\_id

---------- ---------- ---------- ----------- -----------

70001 150.5 2012-10-05 3005 5002

70009 270.65 2012-09-10 3001 5005

70002 65.26 2012-10-05 3002 5001

70004 110.5 2012-08-17 3009 5003

70007 948.5 2012-09-10 3005 5002

70005 2400.6 2012-07-27 3007 5001

70008 5760 2012-09-10 3002 5001

70010 1983.43 2012-10-10 3004 5006

70003 2480.4 2012-10-10 3009 5003

70012 250.45 2012-06-27 3008 5002

70011 75.29 2012-08-17 3003 5007

70013 3045.6 2012-04-25 3002 5001

**TABLE CREATION:**

**SALESMAN:**

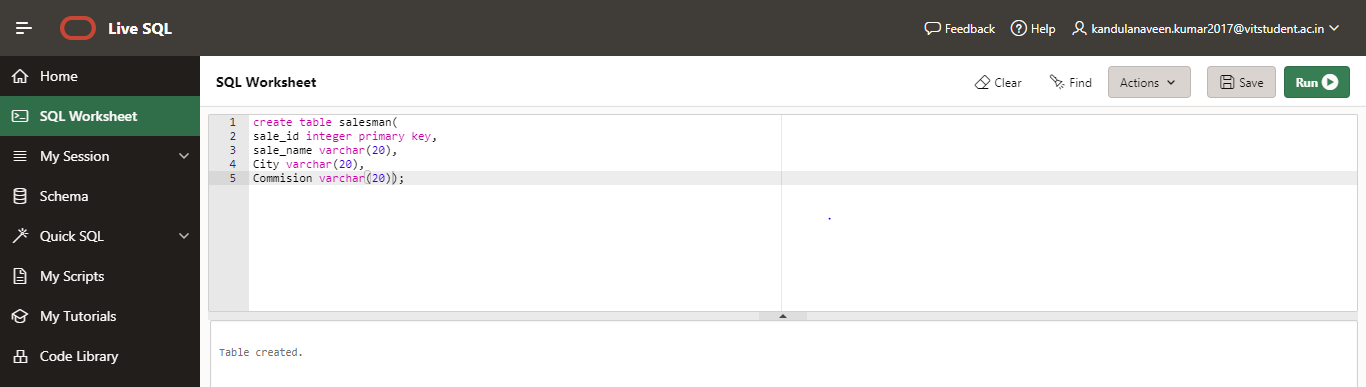
create table salesman(

sale\_id integer primary key,

sale\_name varchar(20),

City varchar(20),

Commision varchar(20) );



CUSTOMER:

create table customer(

cust\_id integer primary key,

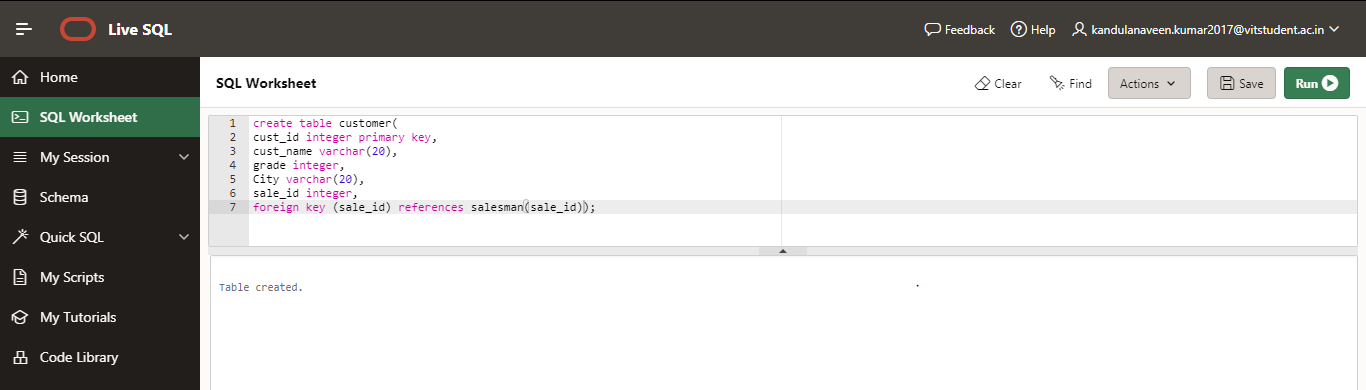
cust\_name varchar(20),

grade integer,

City varchar(20),

sale\_id integer,

foreign key (sale\_id) references salesman(sale\_id));



**ORDERS:**

create table orders(

order\_no integer primary key,

amt number(20),

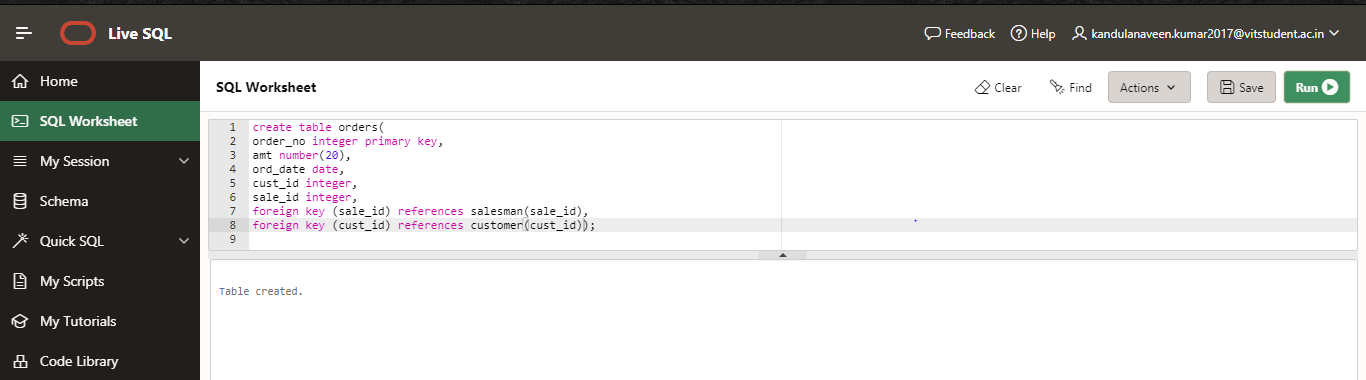
ord\_date date,

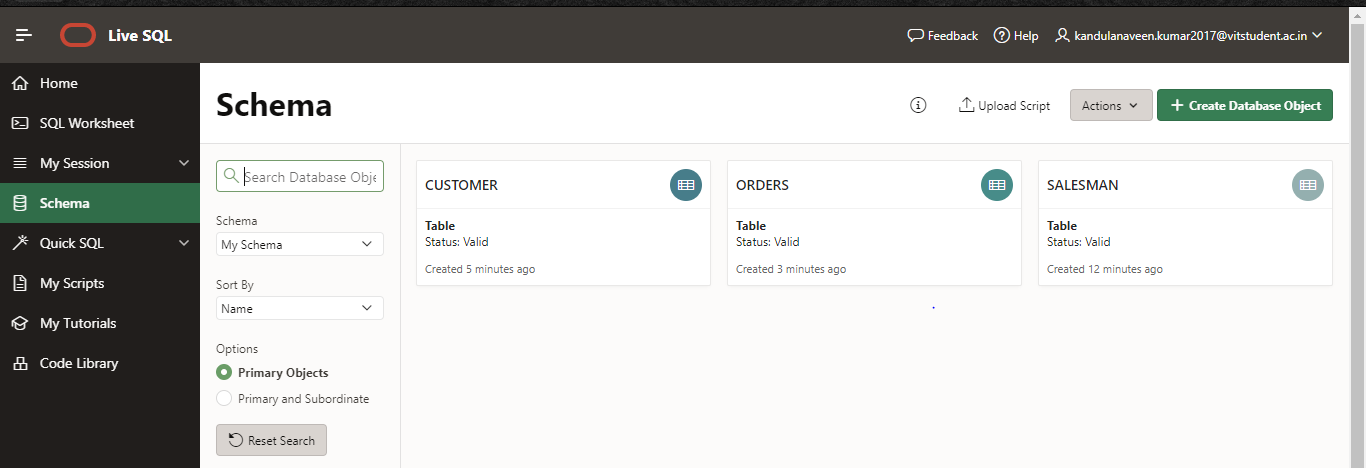
cust\_id integer,

sale\_id integer,

foreign key (sale\_id) references salesman(sale\_id),

foreign key (cust\_id) references customer(cust\_id));





**INSERTING VALUES:**

**SALESMAN:**

insert into salesman values(5001,'James Hoog','New York',0.15);

insert into salesman values(5002,'Nail Knite','Paris',0.13);

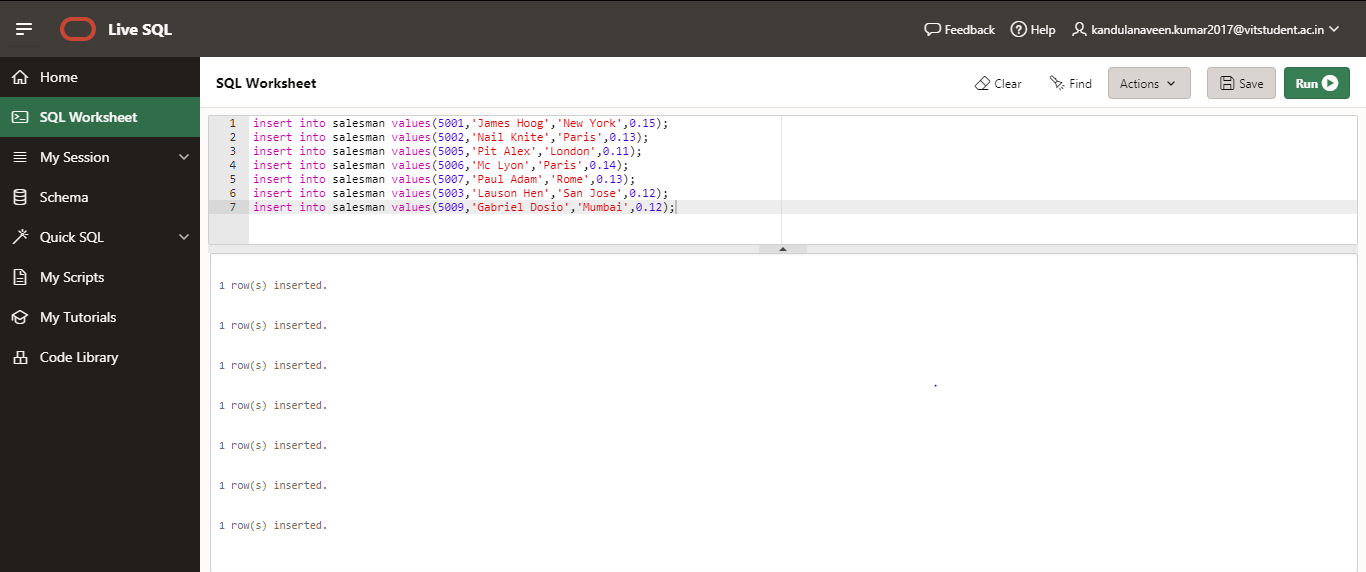
insert into salesman values(5005,'Pit Alex','London',0.11);

insert into salesman values(5006,'Mc Lyon','Paris',0.14);

insert into salesman values(5007,'Paul Adam','Rome',0.13);

insert into salesman values(5003,'Lauson Hen','San Jose',0.12);

insert into salesman values(5009,'Gabriel Dosio','Mumbai',0.12);



**CSUTOMERS:**

insert into customer values(3002,'Nick Rimando',100,'New York',5001);

insert into customer values(3007,'Brad Davis',200,'New York',5001);

insert into customer values(3005,'Graham Suzi',200,'California',5002);

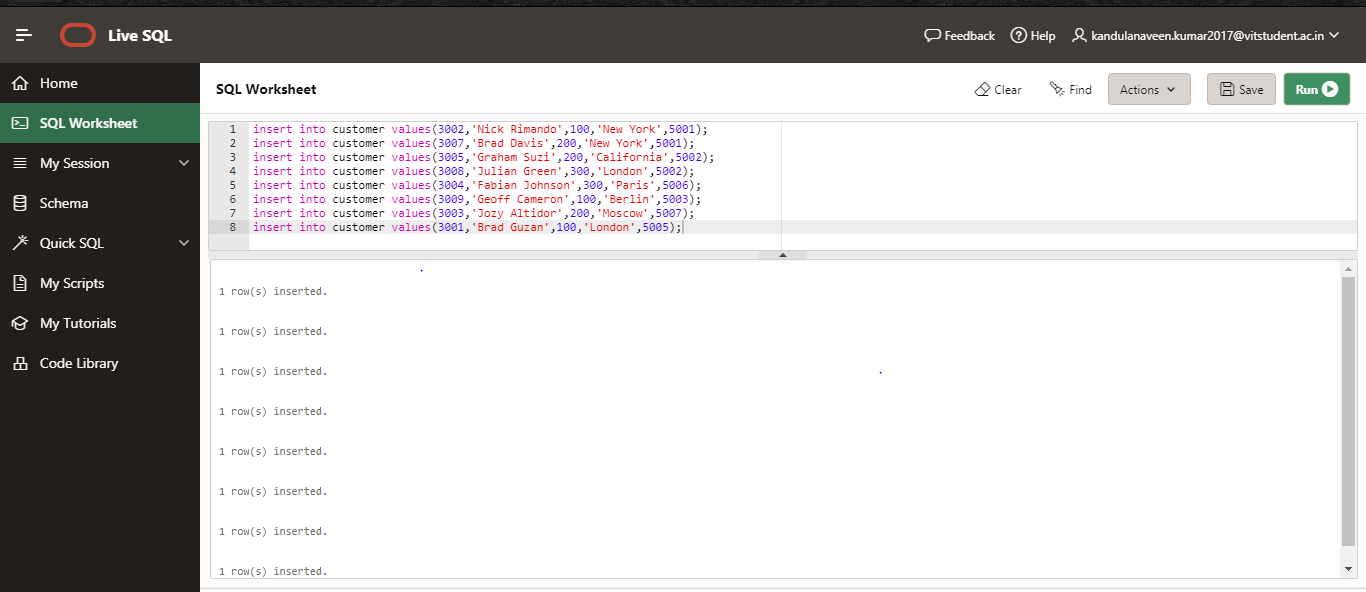
insert into customer values(3008,'Julian Green',300,'London',5002);

insert into customer values(3004,'Fabian Johnson',300,'Paris',5006);

insert into customer values(3009,'Geoff Cameron',100,'Berlin',5003);

insert into customer values(3003,'Jozy Altidor',200,'Moscow',5007);

insert into customer values(3001,'Brad Guzan',100,'London',5005);



**ORDERS:**

insert into orders values(70001,150,date'10-03-12',3005,5002);

insert into orders values(70009,270.65,date'10-03-12',3001,5005);

insert into orders values(70002,65.26,date'10-03-12',3002,5001);

insert into orders values(70004,110.5,date'10-03-12',3009,5003);

insert into orders values(70007,948.5,date'10-03-12',3005,5002);

insert into orders values(70005,2400.6,date'10-03-12',3007,5001);

insert into orders values(70008,5760,date'10-03-12',3002,5001);

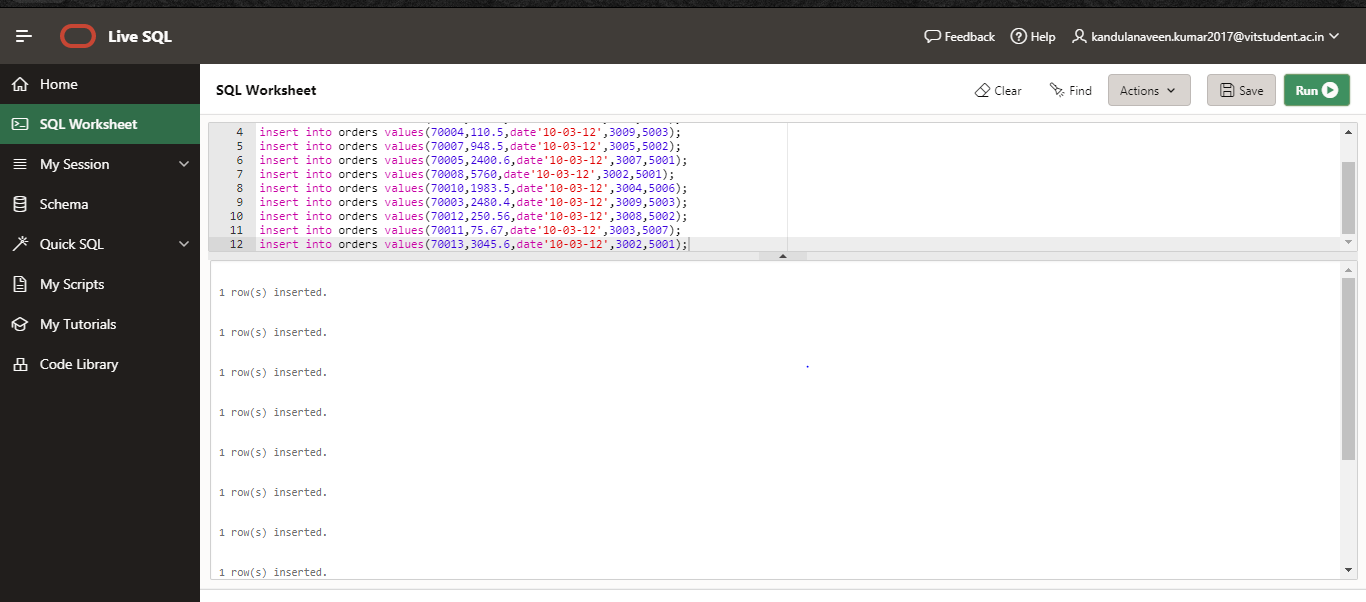
insert into orders values(70010,1983.5,date'10-03-12',3004,5006);

insert into orders values(70003,2480.4,date'10-03-12',3009,5003);

insert into orders values(70012,250.56,date'10-03-12',3008,5002);

insert into orders values(70011,75.67,date'10-03-12',3003,5007);

insert into orders values(70013,3045.6,date'10-03-12',3002,5001);



**Write SQL queries in parallel using ORACLE (Specify DOP)**

ALTER SESSION ENABLE PARALLEL DML;

ALTER TABLE customer PARALLEL 8;

ALTER TABLE salesman PARALLEL 8;

ALTER TABLE orders PARALLEL 8;

1. **Find the number of customers.**

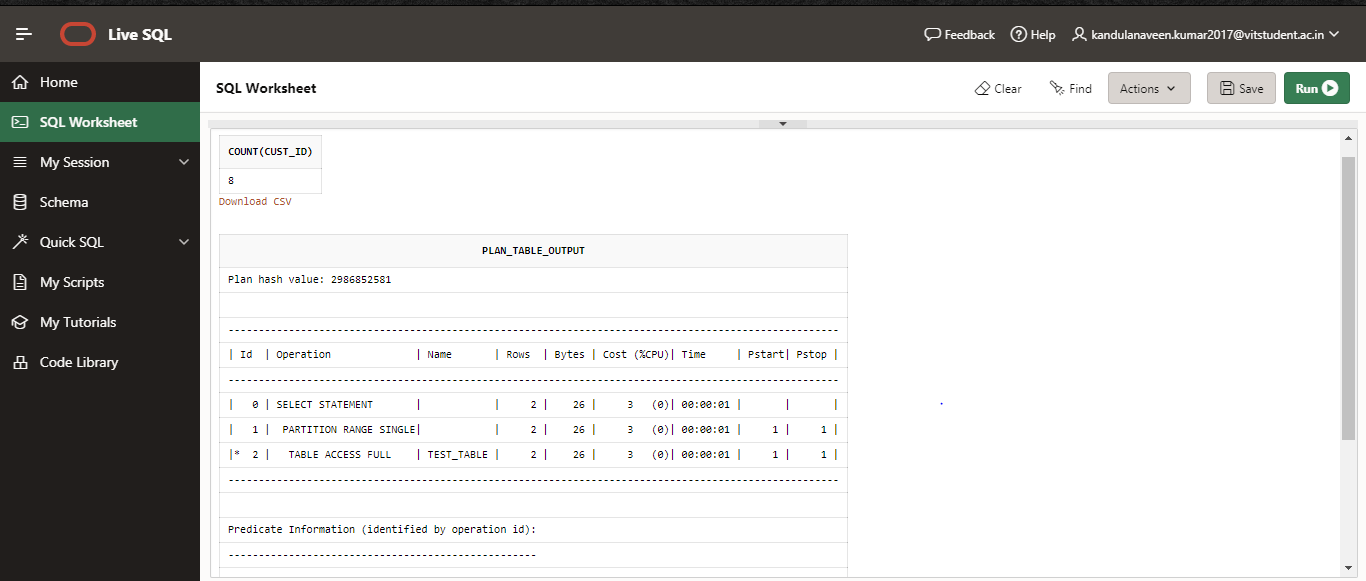
**ANS:**

select /\* +PARALLEL(default)\*/count(cust\_id)

from customer;

SELECT plan\_table\_output

FROM TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



1. **Find the number of orders with purchase amount more than 1000.**

**ANS:**

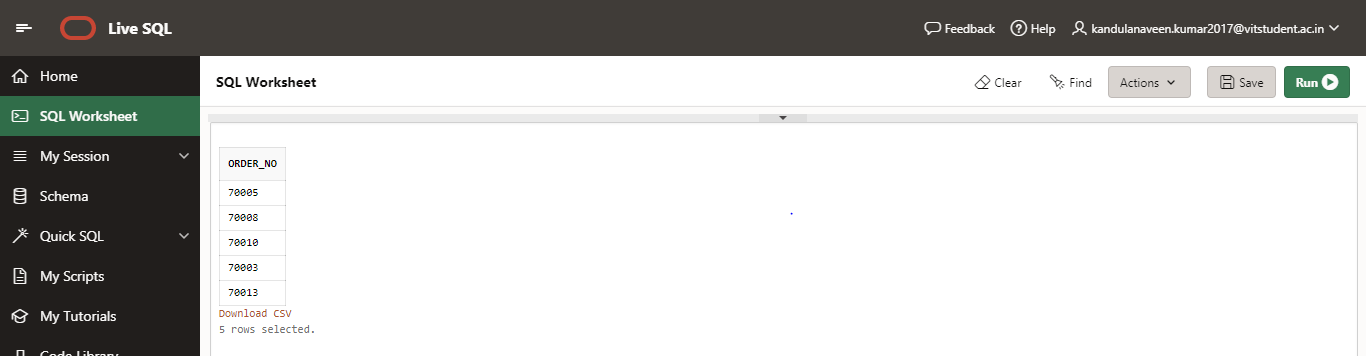
select/\* +PARALLEL(default)\*/ distinct(order\_no)

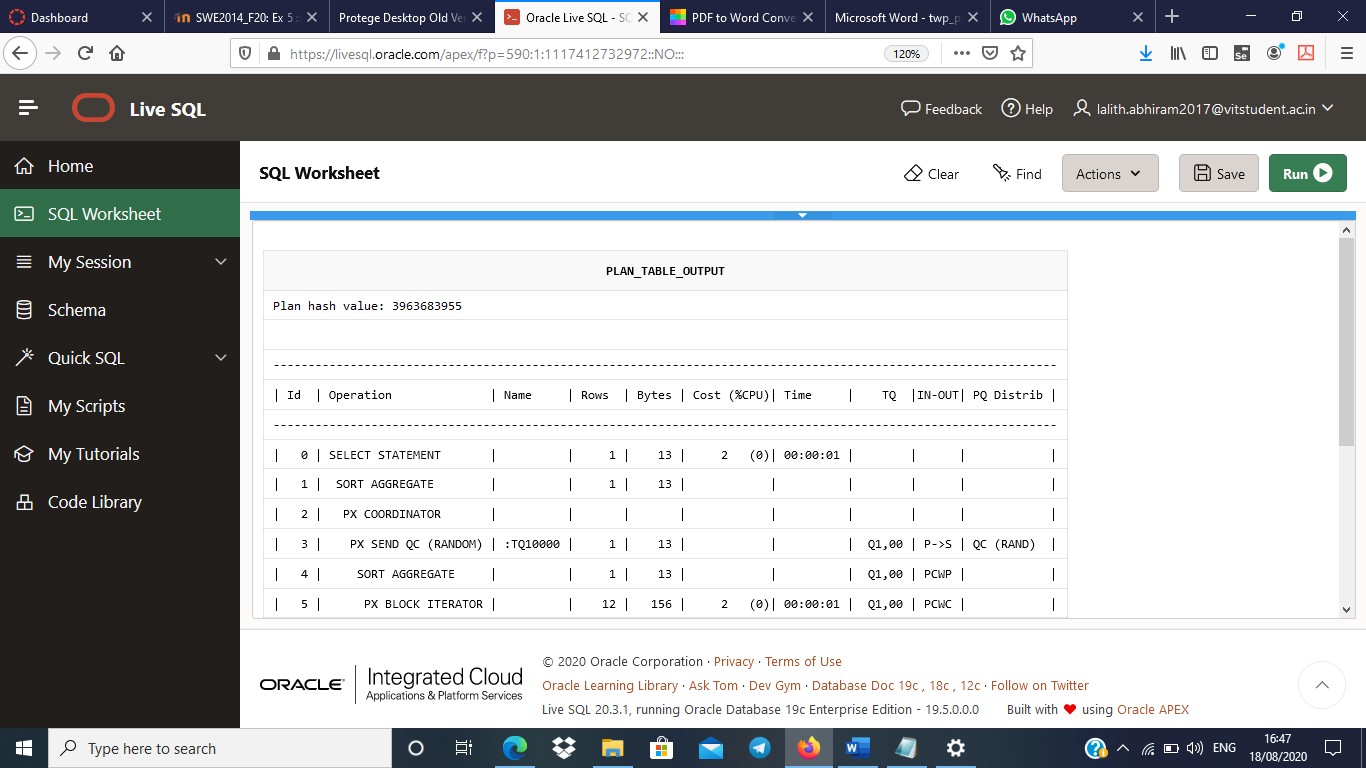
from orders

where amt>1000;

SELECT plan\_table\_output

FROM TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));





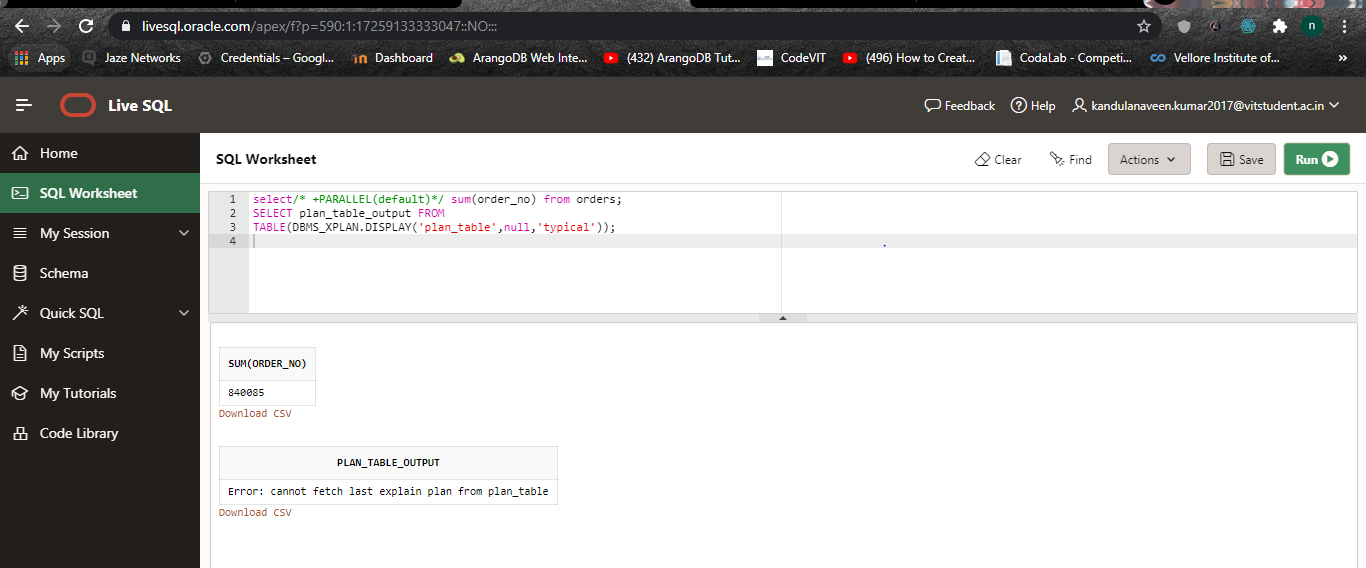
1. **Write a SQL statement to find the total purchase amount of all orders.**

**ANS:**

select/\* +PARALLEL(default)\*/ sum(order\_no) from orders;

SELECT plan\_table\_output FROM

TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



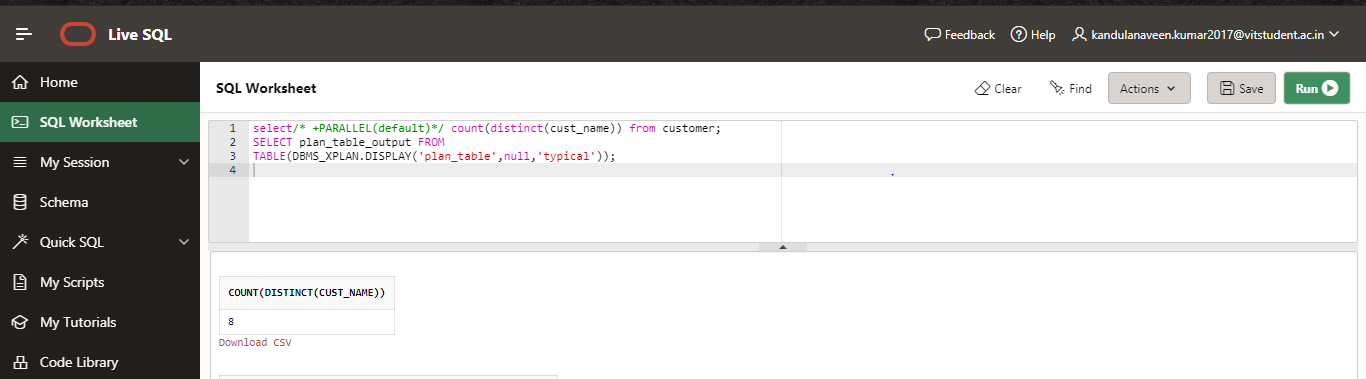
1. **Write a SQL statement know how many customer have listed their names.**

**ANS:**

select/\* +PARALLEL(default)\*/ count(distinct(cust\_name)) from customer;

SELECT plan\_table\_output FROM

TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



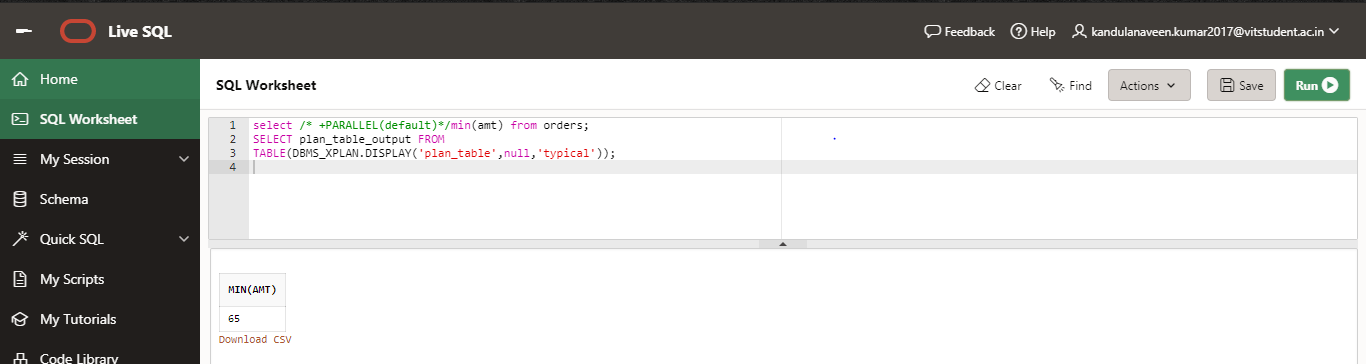
1. **Write a SQL statement to get the minimum purchase amount of all the orders.**

**ANS:**

select /\* +PARALLEL(default)\*/min(amt) from orders;

SELECT plan\_table\_output FROM

TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



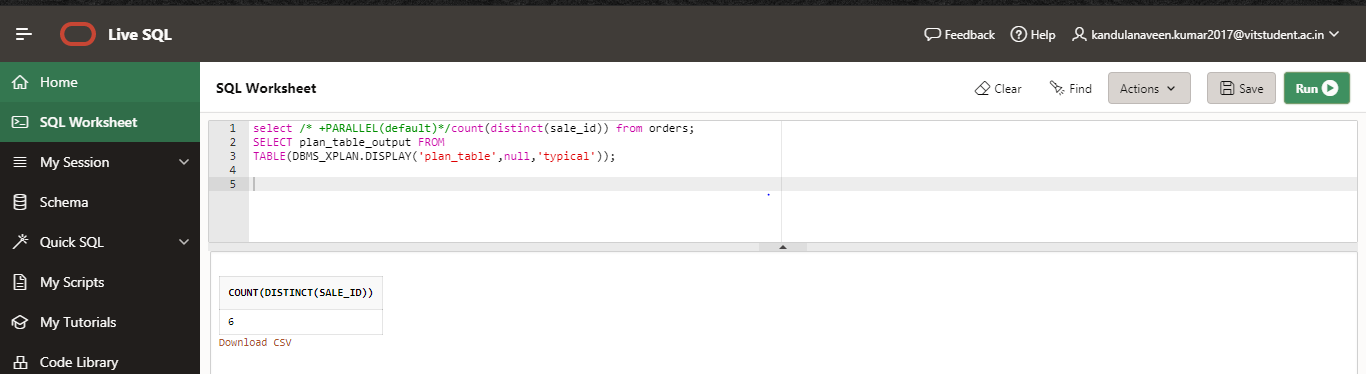
1. **Write a SQL statement to find the number of salesmen currently listing for all of their customers.**

**ANS:**

select /\* +PARALLEL(default)\*/count(distinct(sale\_id)) from orders;

SELECT plan\_table\_output FROM

TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



1. **Display all the orders from the orders table issued by the salesman 'Paul Adam'.**

**ANS:**

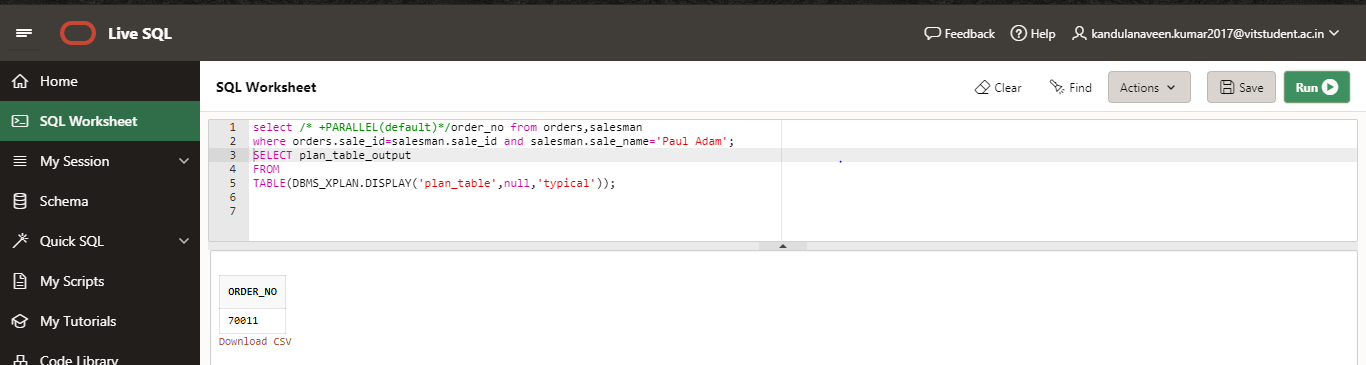
select /\* +PARALLEL(default)\*/order\_no from orders,salesman

where orders.sale\_id=salesman.sale\_id and salesman.sale\_name='Paul Adam';

SELECT plan\_table\_output

FROM

TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



1. **Display all the orders for the salesman who belongs to the city Paris**

**ANS:**

select /\* +PARALLEL(default)\*/order\_no from orders,customer

where orders.cust\_id=customer.cust\_id and customer.city='Paris';

SELECT plan\_table\_output

FROM

TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



1. **Display the commission of all the salesmen servicing customers in London**

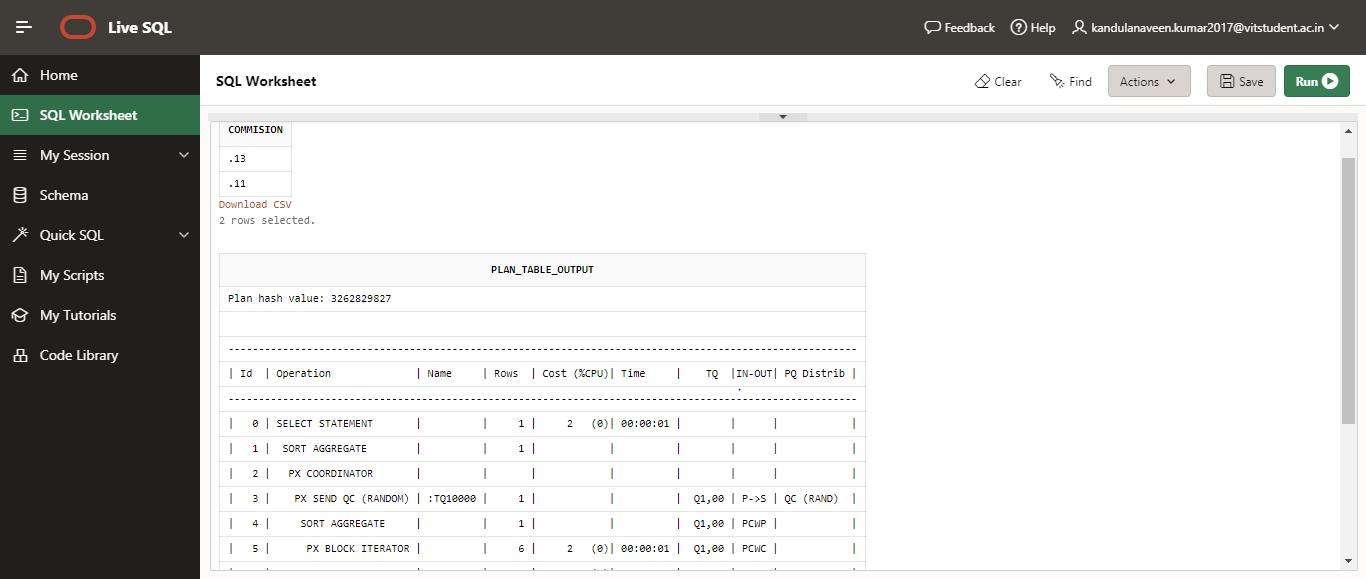
**ANS:**

select /\* +PARALLEL(default)\*/Commision from salesman,customer

where salesman.sale\_id=customer.sale\_id and customer.city='London';

SELECT plan\_table\_output FROM

TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



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

**ANS:**

select /\* +PARALLEL(default)\*/ \* from salesman

where sale\_name In ('James Hoog','Gabriel Dosio');

SELECT plan\_table\_output FROM

TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));

