

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 | : | **28-07-2020** | Marks | : | **10 Points** |

**NAME: K. NAVEEN KUMAR**

**REG NO: 17MIS1056**

**Exercise 3: INDEX and EXECUTION PLAN**

Consider the following schema for customer-sale, where the primary keys are underlined.

* Customer (**Custid : Integer**, first\_name: VARCHAR2, last\_name: VARCHAR2, gender : char, email : VARCHAR2 NOT NULL, dob : DATE NOT NULL)
* Item(**Item\_id: Integer**, item\_name: VARCHAR2, price: integer)
* Sale(**bill\_No: Integer**, bill\_date: Date, **Custid : Integer, Item\_id: Integer**, qty\_sold: Integer)

1. First create tables with the primary Key constraints (Underlined attributes are primary keys )
2. Add the Foreign Keys wherever it is applicable.
3. Make Sales table Bill\_No as Sequence Field
4. The attributes Item\_name and cust\_name cannot be null.
5. Insert the appropriate values/records into the tables as follows.

ANS:

**Customer Table:**

create table customer(

custid integer primary key,

first\_name varchar2(30),

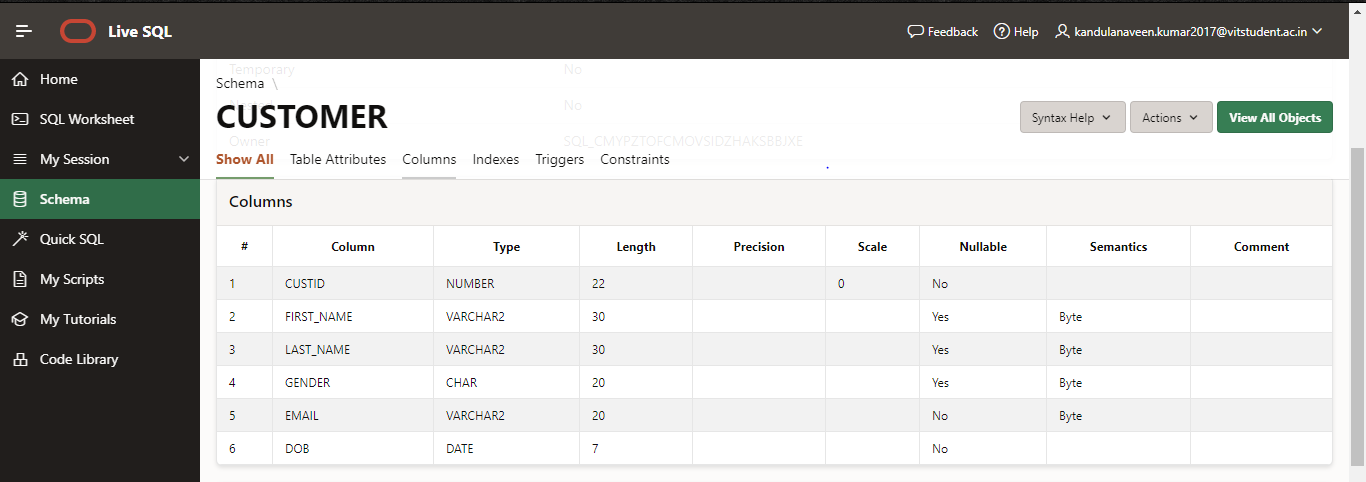
last\_name varchar2(30),

gender char(20),

email varchar(20) not null,

dob date not null);

output:



**Item table:**

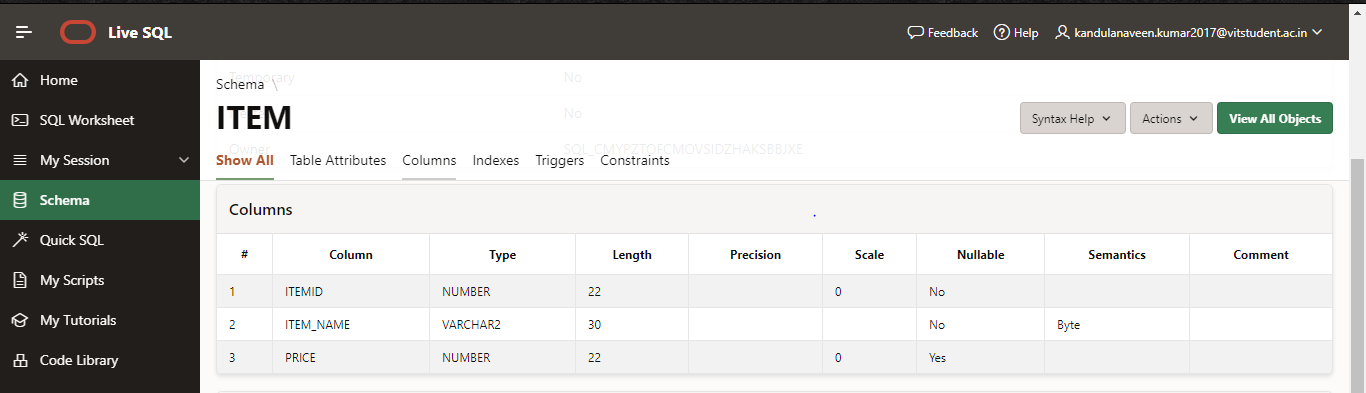
create table item(

itemid integer primary key,

item\_name varchar2(30) not null,

price integer);

Output:



**Sale table:**

create table sale(

billno integer primary key,

bill\_date date,

custid integer,

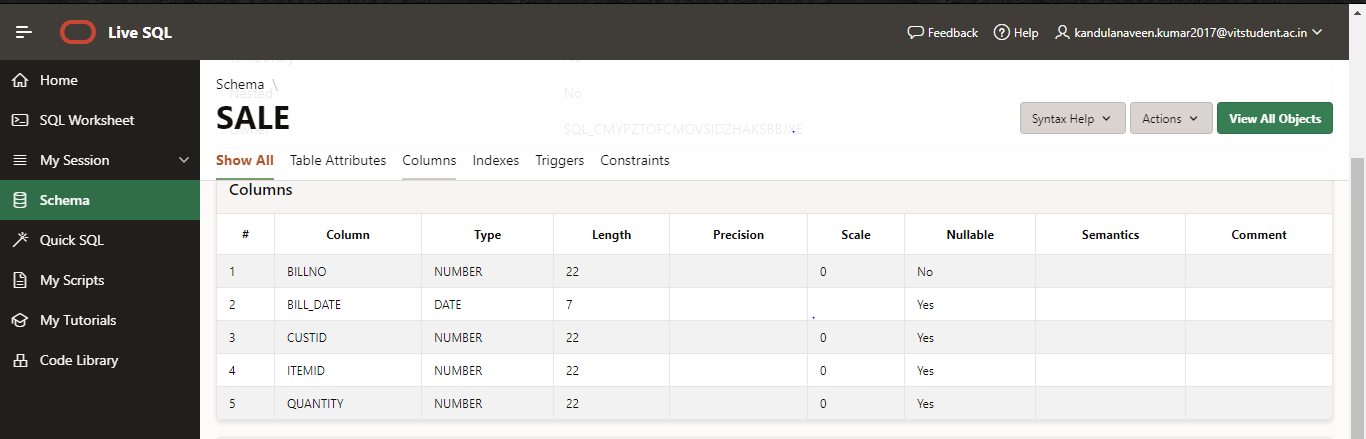
itemid integer,

quantity integer,

foreign key(custid) references customer(custid),

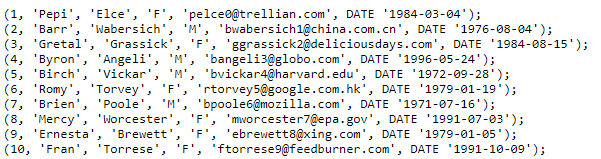
foreign key(itemid) references item(itemid));

**output:**

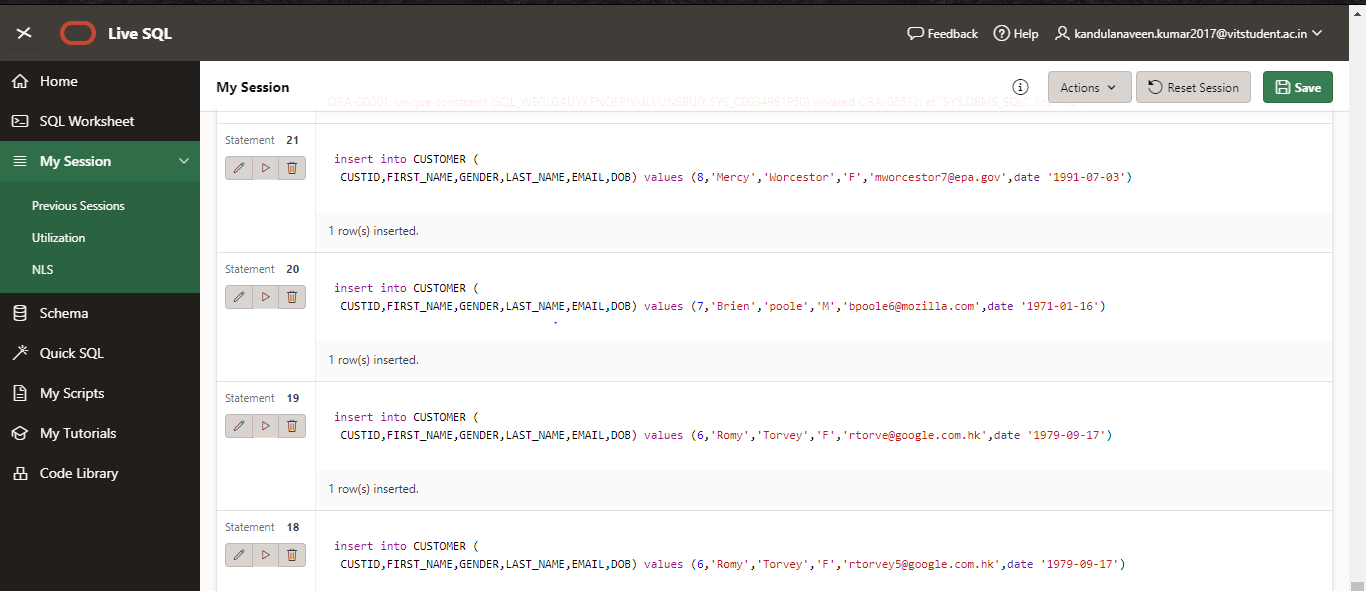


1. Add the Foreign Keys wherever it is applicable.
2. Make Sales table Bill\_No as Sequence Field
3. The attributes Item\_name and cust\_name cannot be null.
4. Insert the appropriate values/records into the tables as follows.

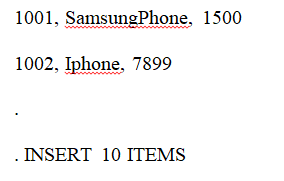
Insert into Customer table :



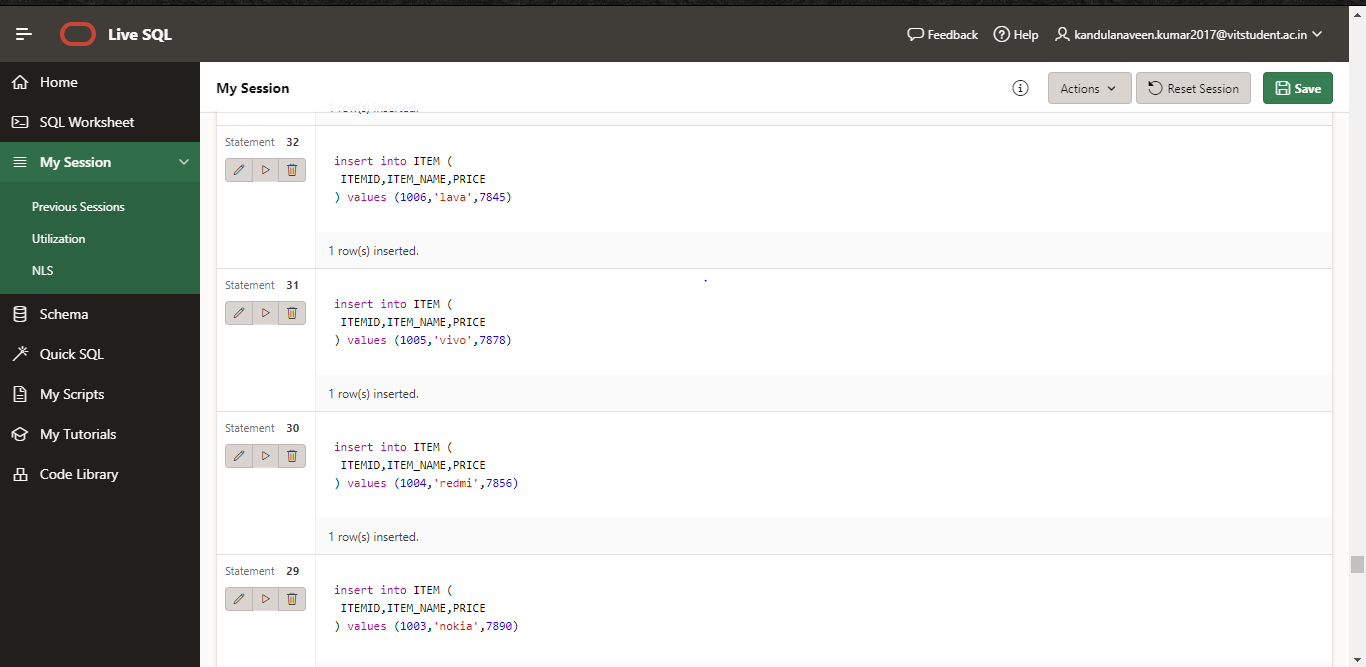
ALL 10 VALUES ARE INSERTED THIS IS SAMPLE SCREENSHOT OF 4 VALUES



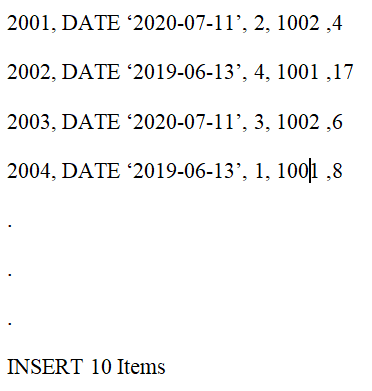
Insert into Items table :



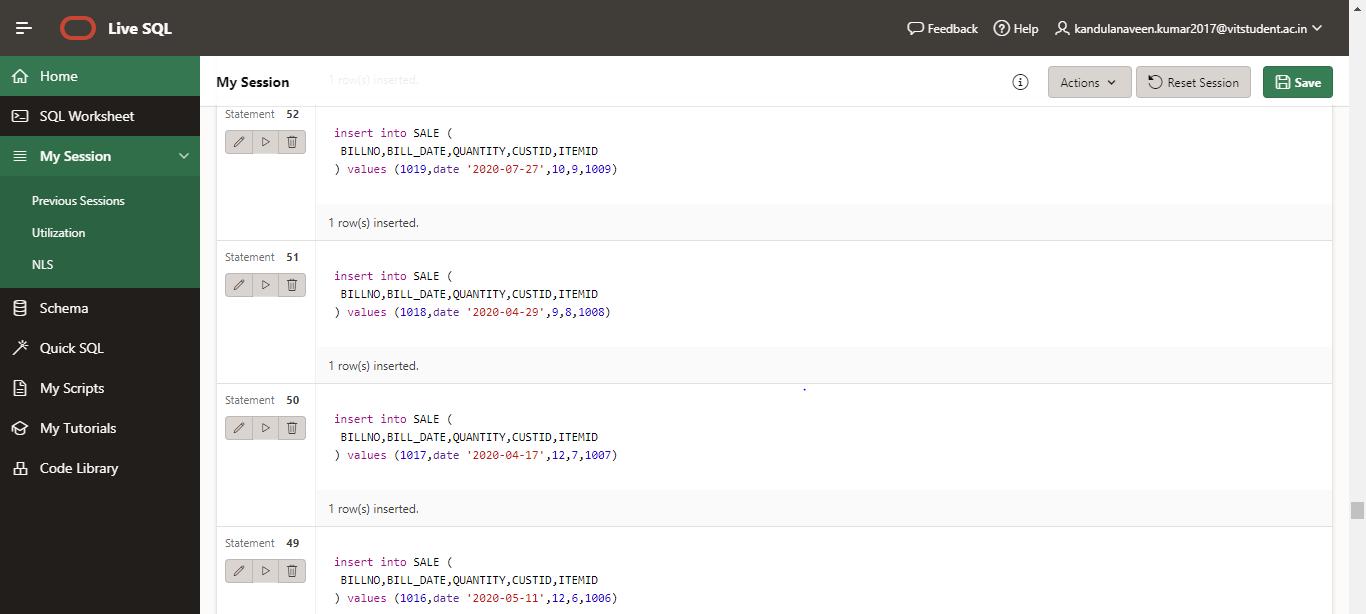
ALL 10 VALUES ARE INSERTED THIS IS SAMPLE SCREENSHOT OF 4 VALUES



Insert into Sales table :



ALL 10 VALUES ARE INSERTED THIS IS SAMPLE SCREENSHOT OF 4 VALUES



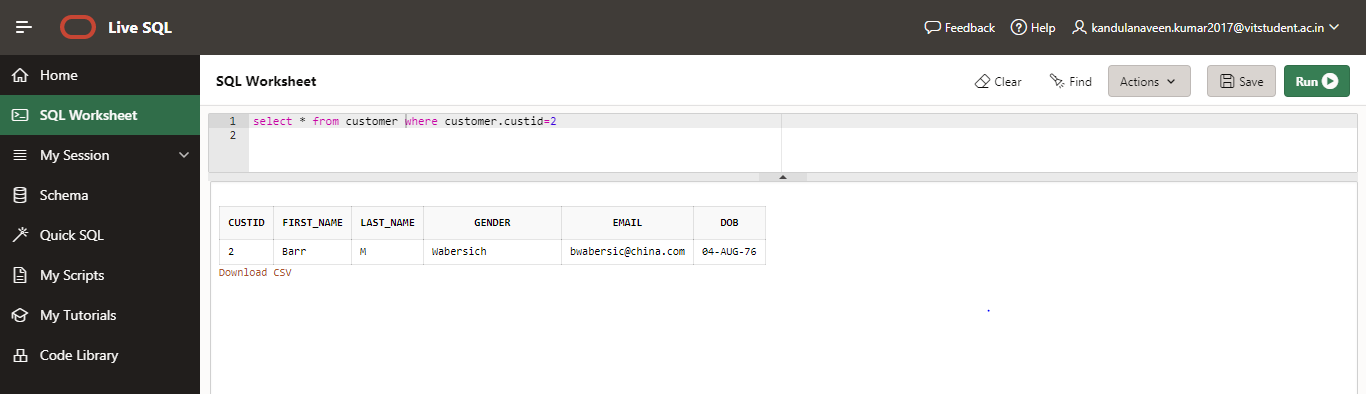
QUESTIONS:

1. WITH/WITHOUT Index
   1. Query 1: Display all the records in Customer table where cust id=2 .
      1. Explain plan for Query1 (Mention what scan is chosen by optimizer)

ANS:

select \* from customer where customer.custid=2

**output:**



* 1. Create Index on Customer ID in Customer table

QUERY :

create index id on customer(custid)

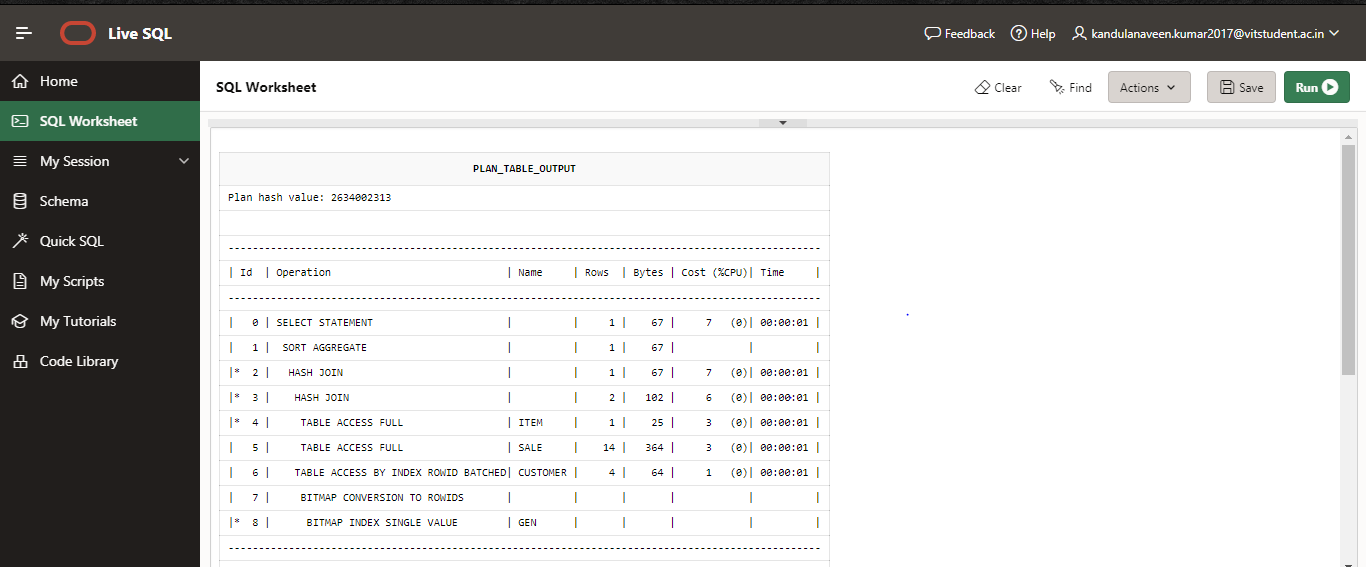
* + 1. Query 1A: Display all the records in Customer table where cust id=2 .
    2. Explain plan for Query1A(Mention what scan is chosen by optimizer)

ANS:

SELECT \* FROM customer WHERE custid=2;

SELECT plan\_table\_output

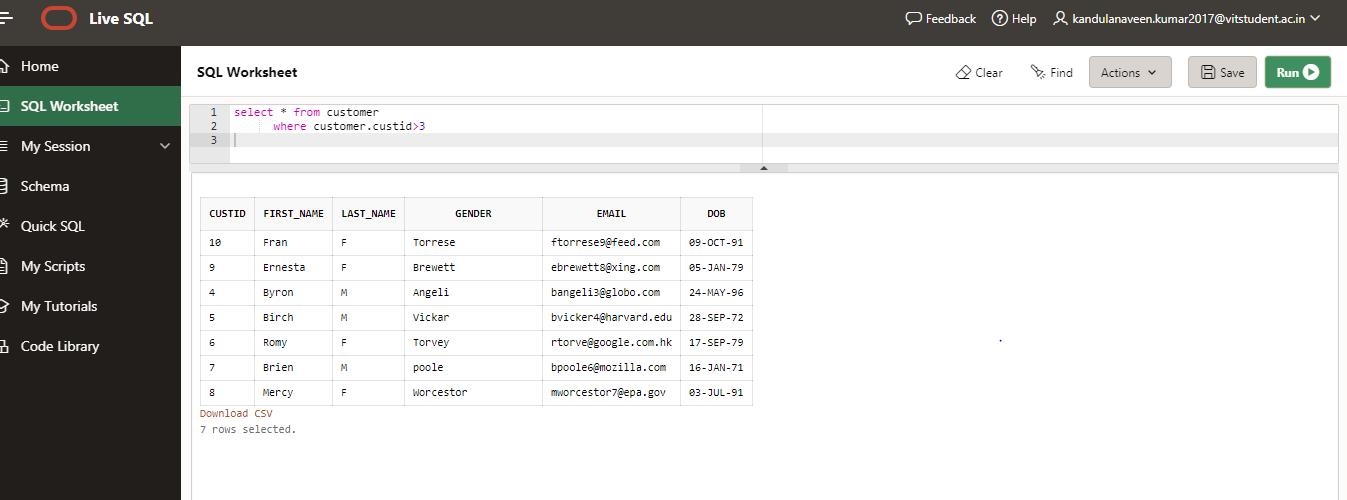
FROM TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



* 1. Query 1B: Display all the records in Customer table greater than 3 .

ANS:

select \* from customer where customer.custid>3



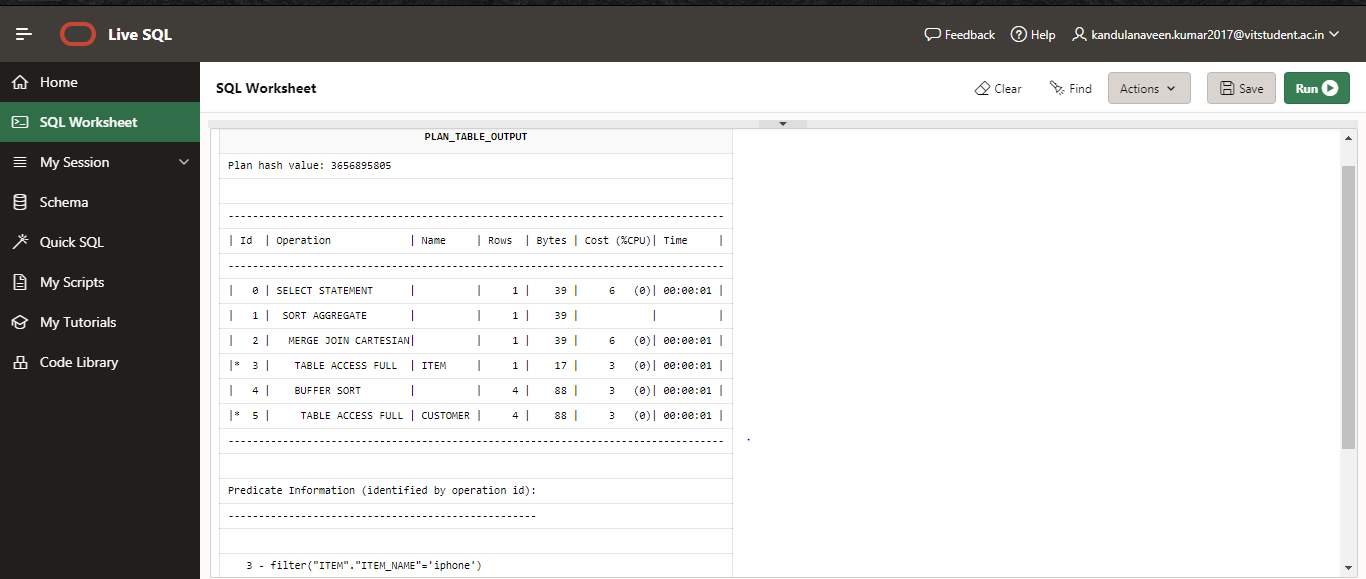
1.Explain plan for Query1B(Mention what scan is chosen by optimizer

select \* from customer

where customer.custid>3

SELECT plan\_table\_output

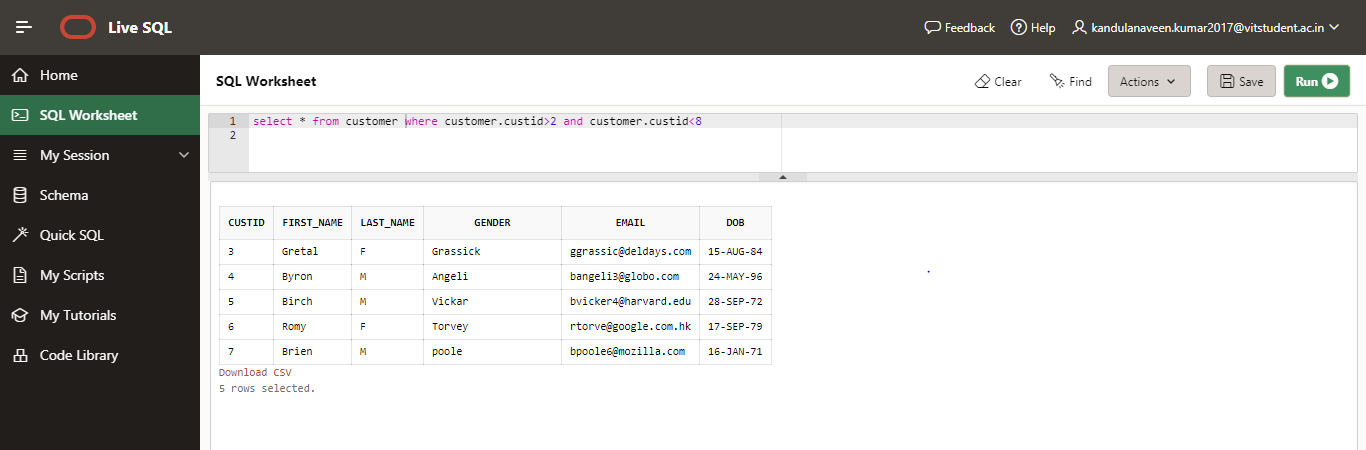
FROM TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



* 1. Query 1C: Display all the records in Customer table between 2 and 8.

ANS:

select \* from customer where customer.custid>2 and customer.custid<8



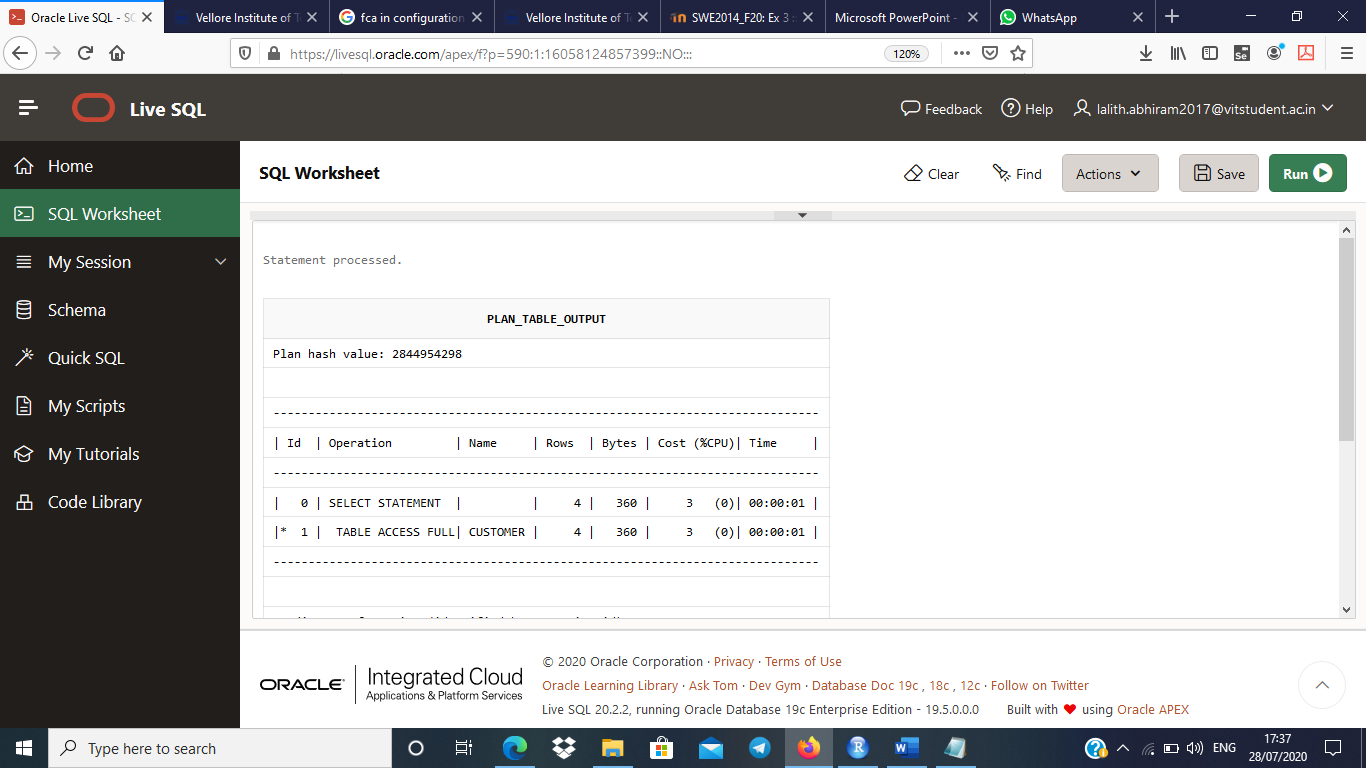
* + 1. Explain plan for Query1C(Mention what scan is chosen by optimizer)

ANS:

select \* from customer where customer.custid>2 and customer.custid<8;

SELECT plan\_table\_output

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



* 1. Create a composite Index on first\_name, last\_name, email id

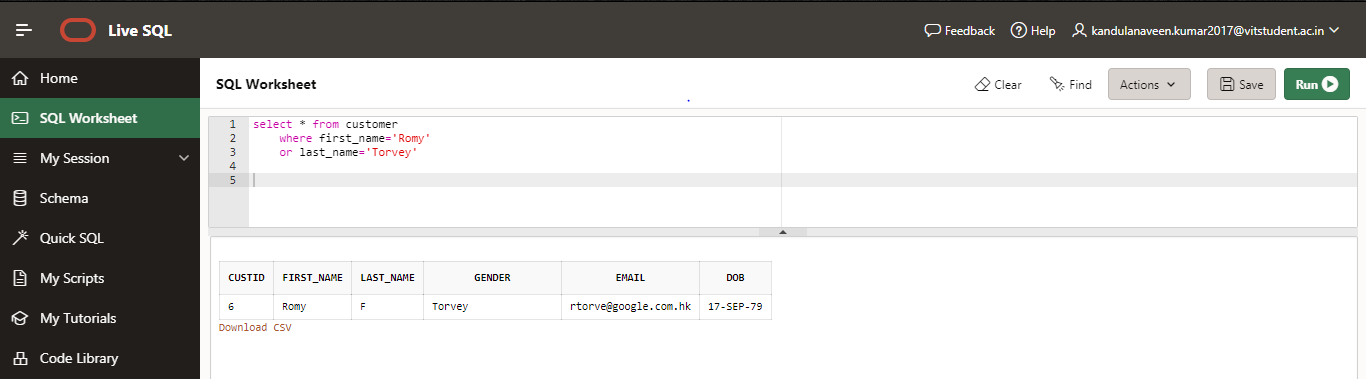
ANS:

create index tr on customer(first\_name,last\_name,email)

* + 1. Query 2: Find the customers whose first\_name, last\_name is Romy and Torvey

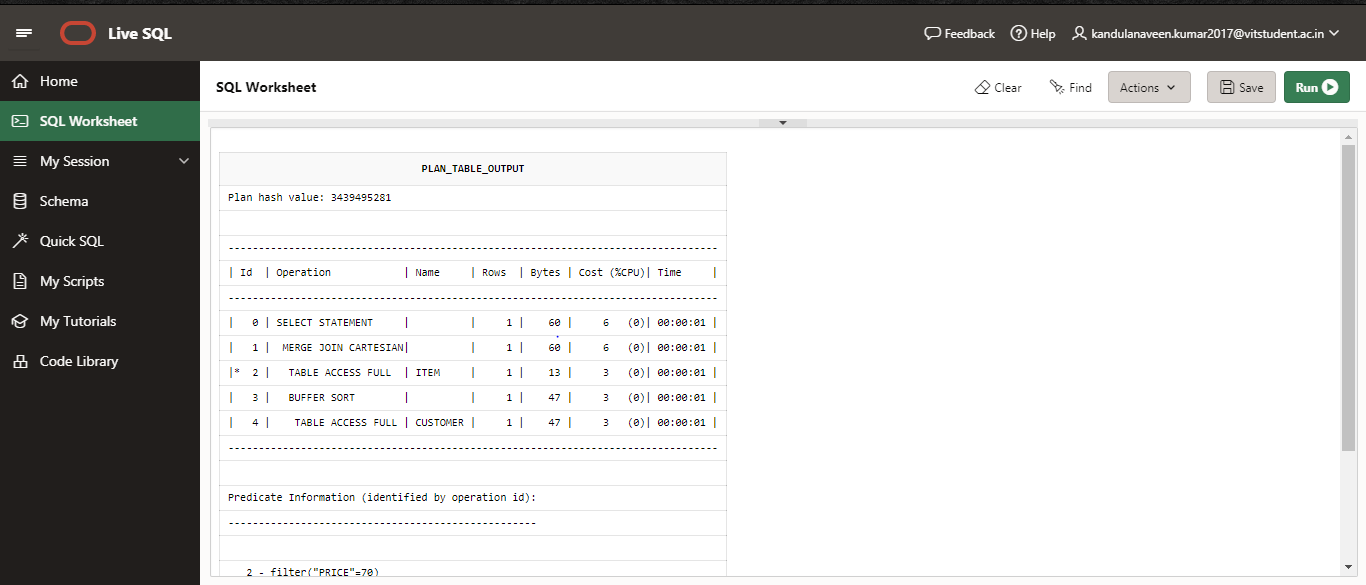
ANS:

select \* from customer where first\_name='Romy or last\_name='Torvey'



* + 1. Explain plan for Query2 (Mention what scan is chosen by optimizer)

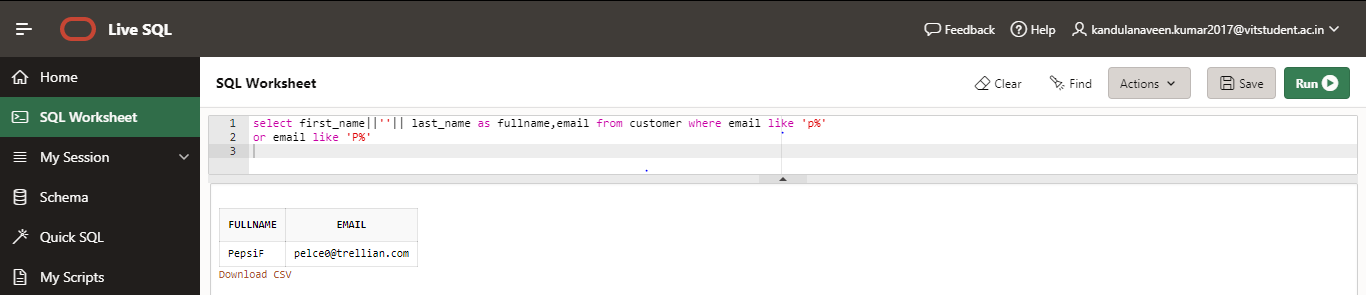
ANS:



* + 1. Query 3: Find the customers whose mail id starts with ‘p’

ANS: select first\_name||''|| last\_name as fullname,email from customer where email like 'p%'

or email like 'P%'

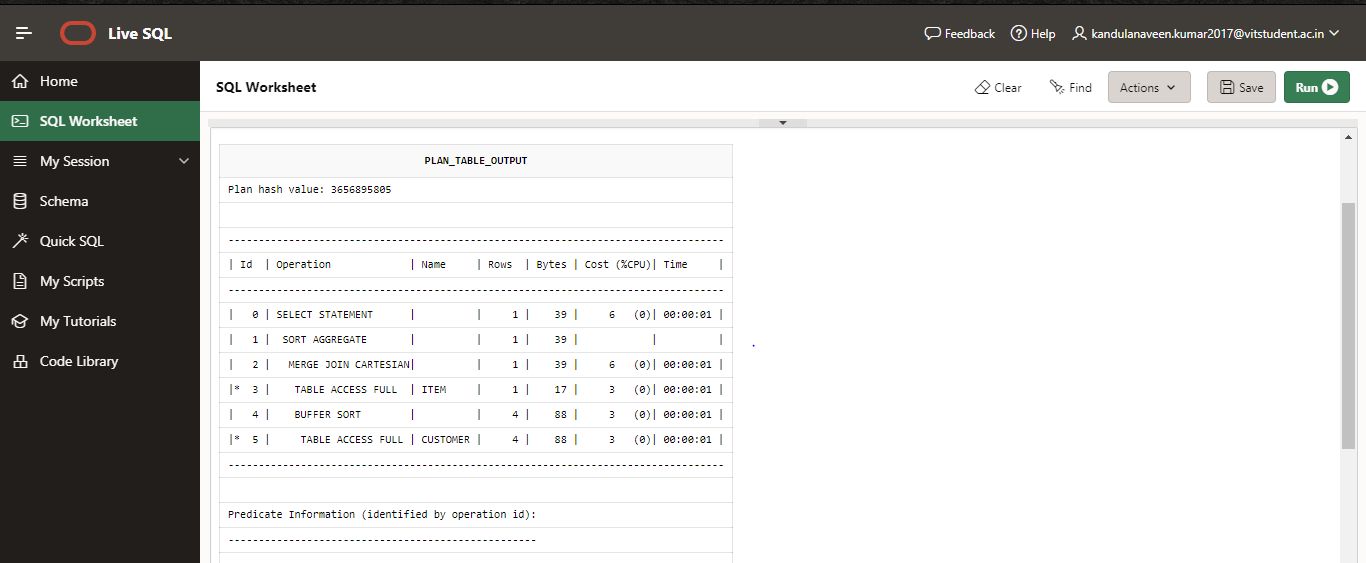


* + 1. Explain plan for Query3 (Mention what scan is chosen by optimizer)

ANS:

select first\_name||''|| last\_name as fullname,email from customer where email like 'p%'or email like 'P%';

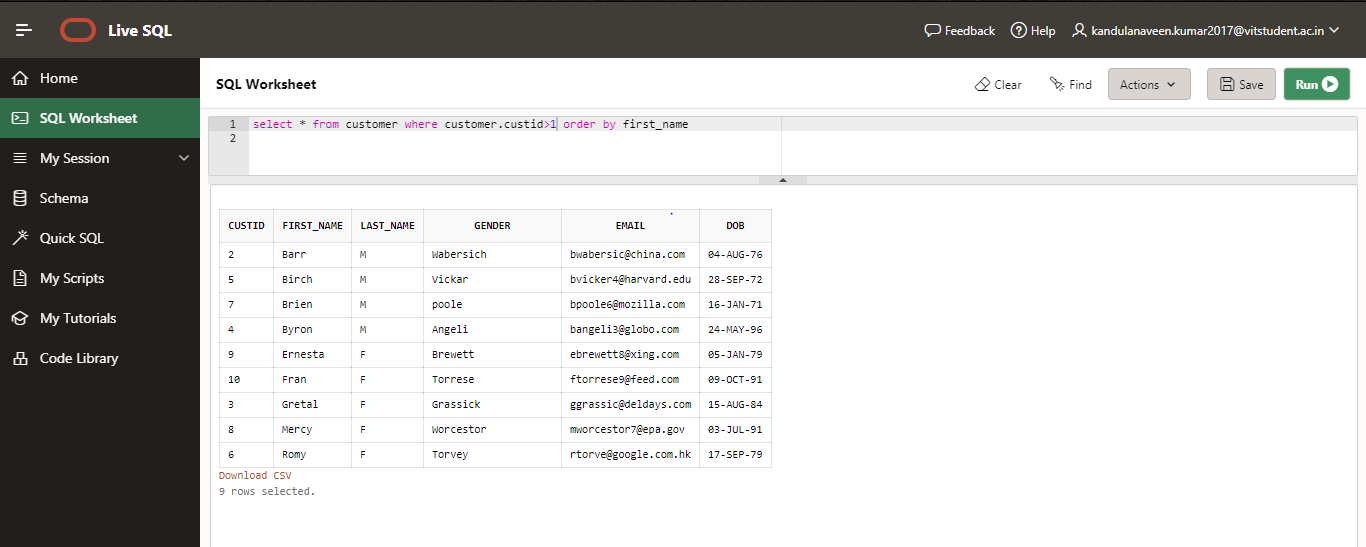
SELECT plan\_table\_output FROM TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



* + 1. Query 4: Display the customer’s details whose customer id greater than 2001 and sort by their name in descending order.

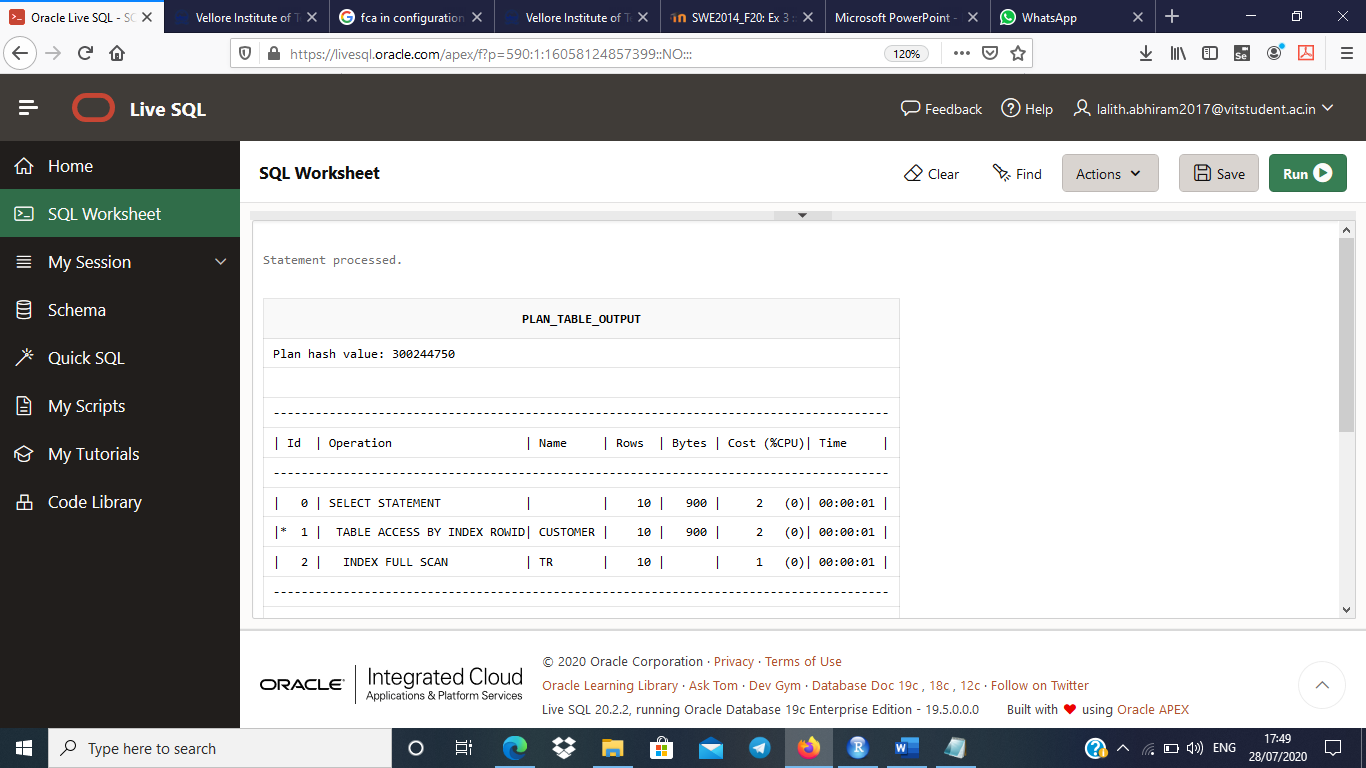
ANS:

select \* from customer where customer.custid>1 order by first\_name



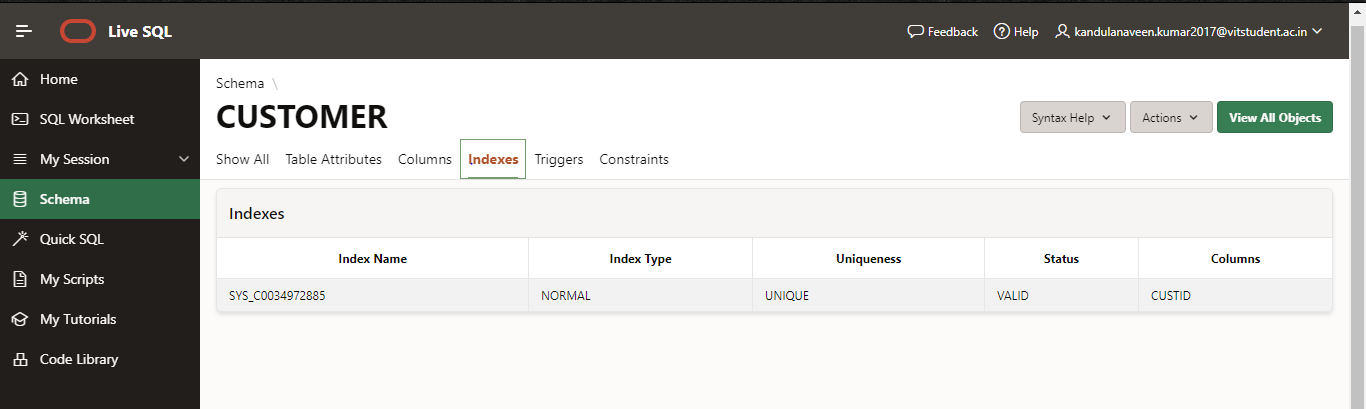
* + 1. Explain plan for Query4 (Mention what scan is chosen by optimizer)

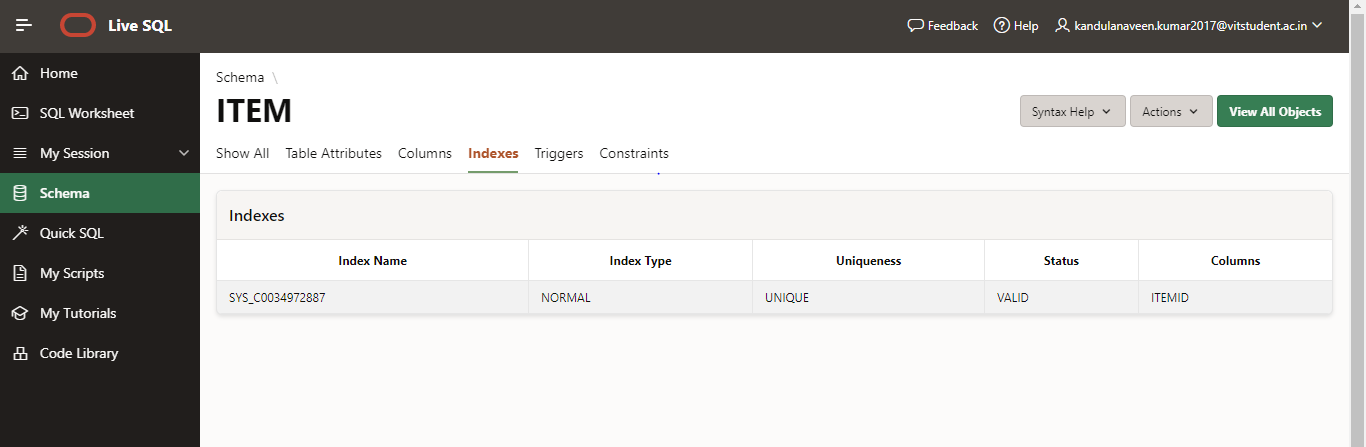
ANS:

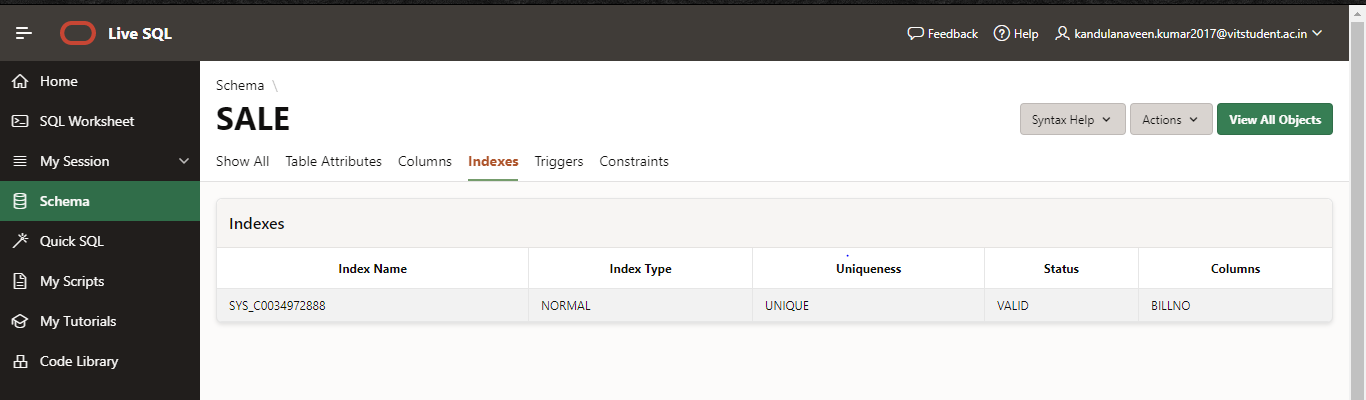


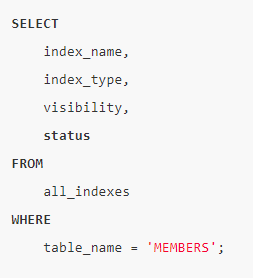
* 1. View all indexes of a table, you query from the all\_indexes view:

ANS:



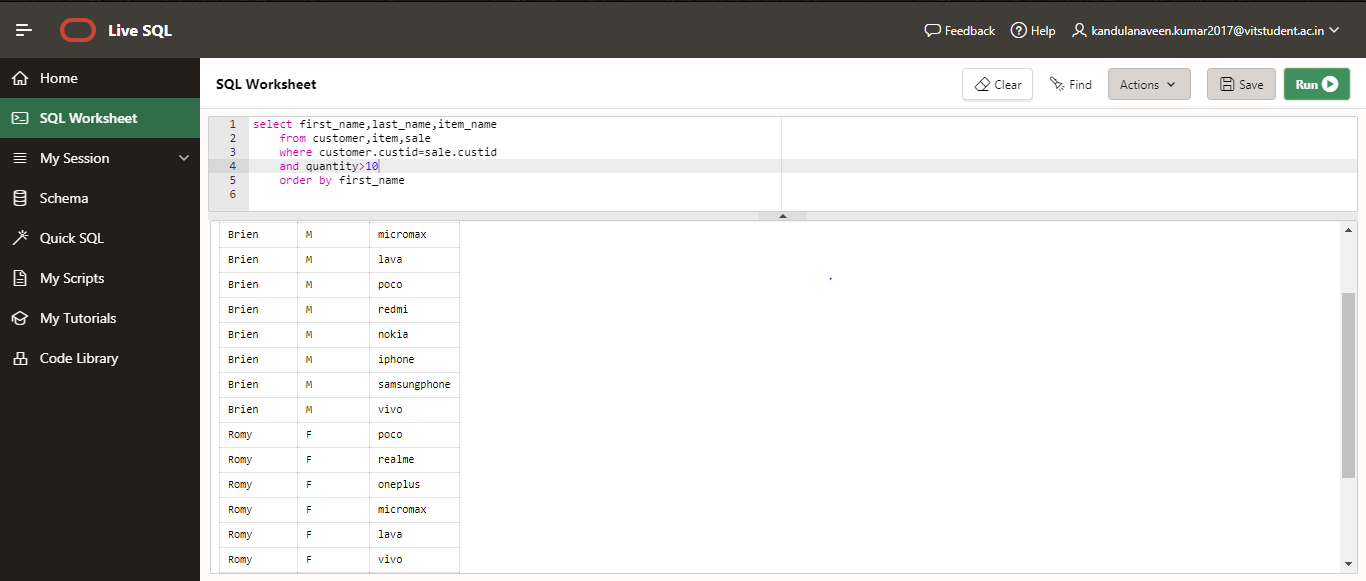






* 1. Join queries:
     1. Query 5 : Display the customer first and last name , item name whose quantity buy more than 10 items. Sort by customers first name.

ANS: select first\_name,last\_name,item\_name from customer,item,sale where customer.custid=sale.custid and quantity>10 order by first\_name



* + 1. Explain plan for Query4 (Mention what JOIN is chosen by optimizer)

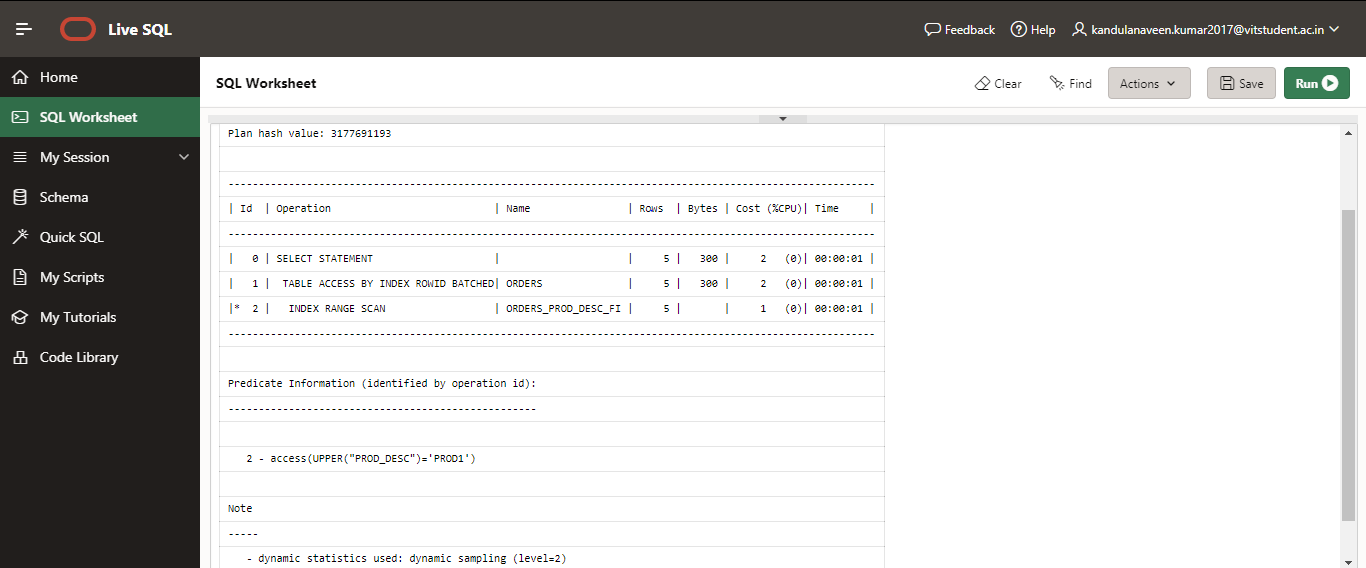
ANS:

select first\_name,last\_name,item\_name from customer,item,sale where customer.custid=sale.custid

and quantity>15 order by first\_name;

SELECT plan\_table\_output

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

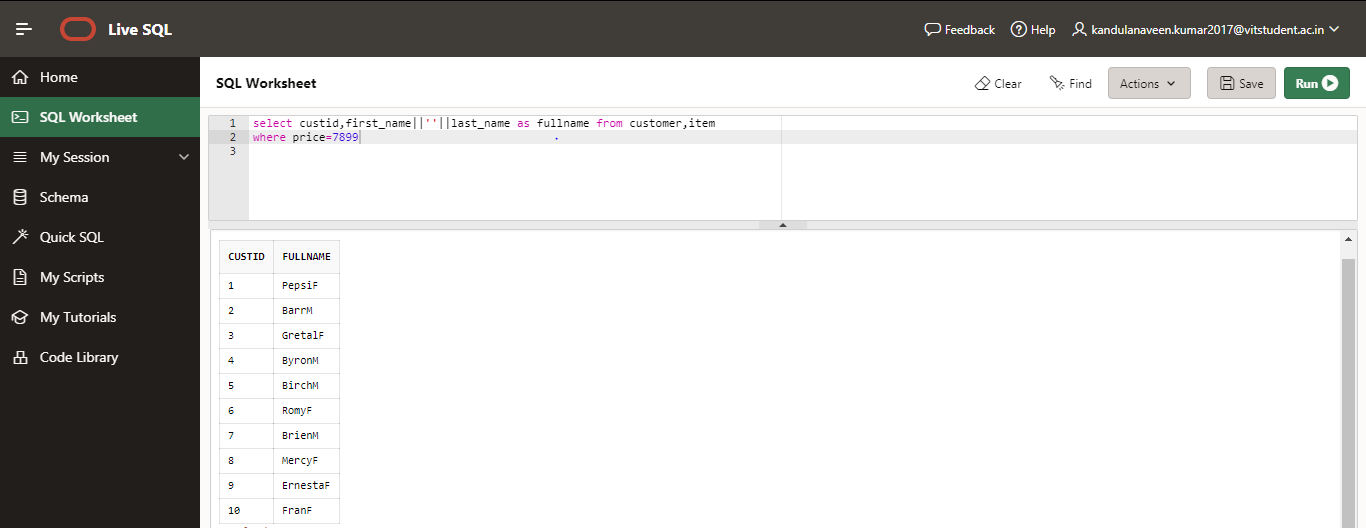


* 1. Range Aggregate queries: (Use order by appropriately)
     1. Query 6 : Select the customers who buy the item whose price is the highest.

ANS:

select custid,first\_name||''||last\_name as fullname from customer,item

where price=7899



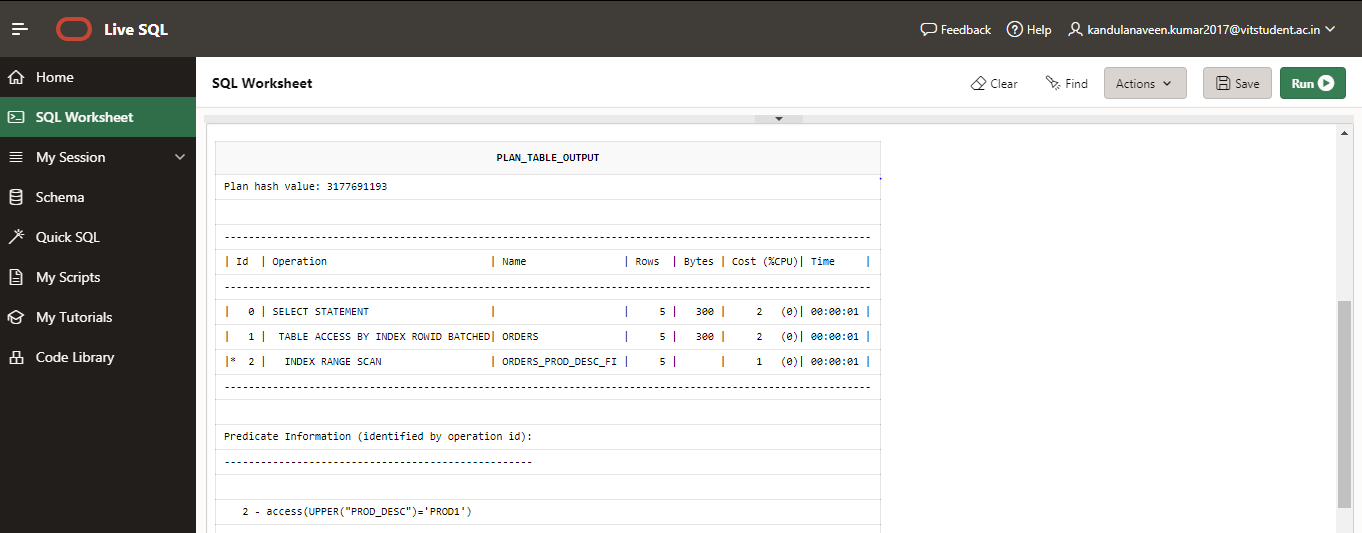
* + 1. Explain plan for Query 6 (Mention what JOIN is chosen by optimizer)

ANS:

select custid,first\_name||''||last\_name as fullname from customer,item where price=7899;

SELECT plan\_table\_output

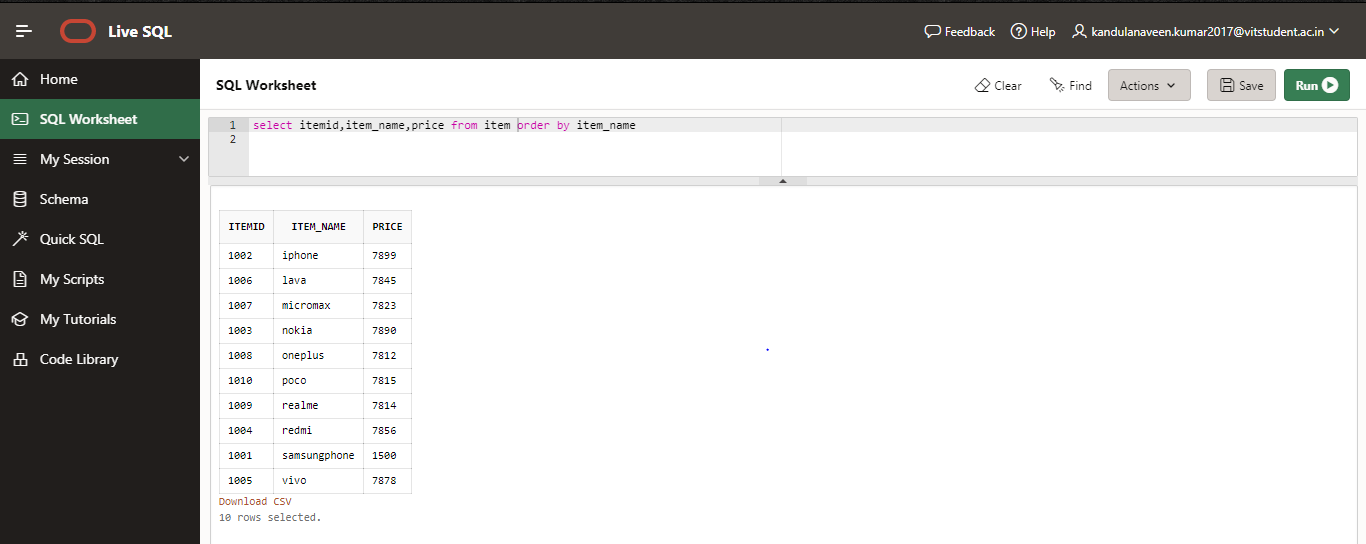
FROM TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



* 1. queries: (Use order by appropriately)
     1. Query 7: Display the itemsid, item name , price order by item name.

ANS:

select itemid,item\_name,price from item order by item\_name



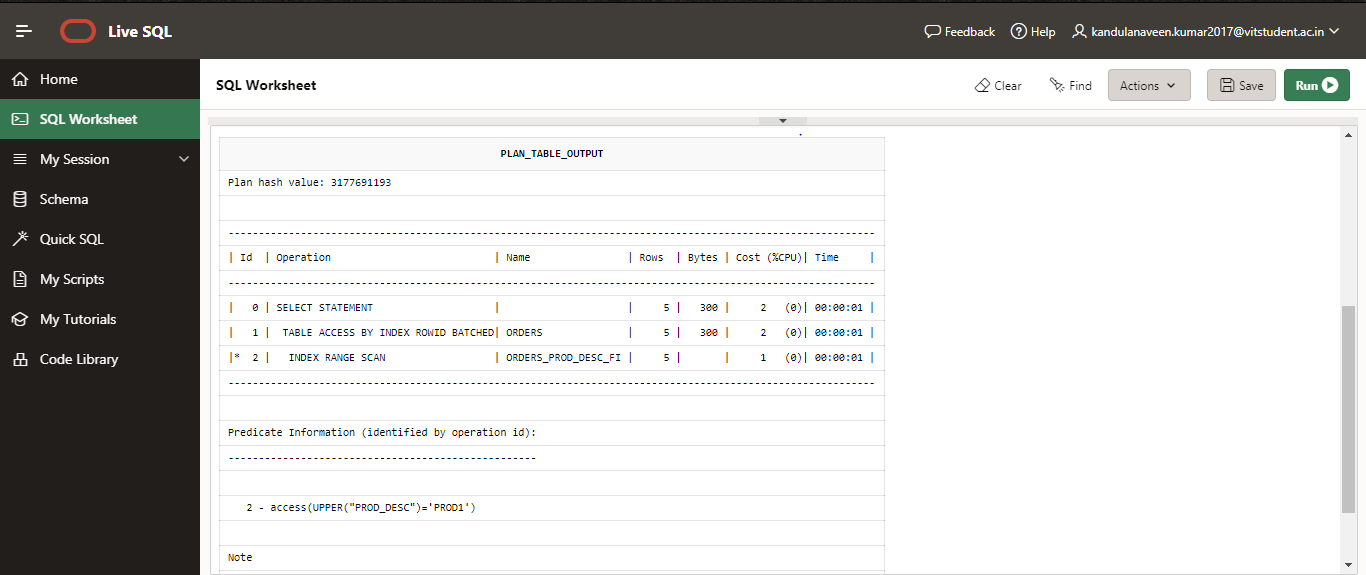
* + 1. Explain plan for Query 7 (Mention what JOIN is chosen by optimizer)

ANS:

select itemid,item\_name,price from item order by item\_name;

SELECT plan\_table\_output

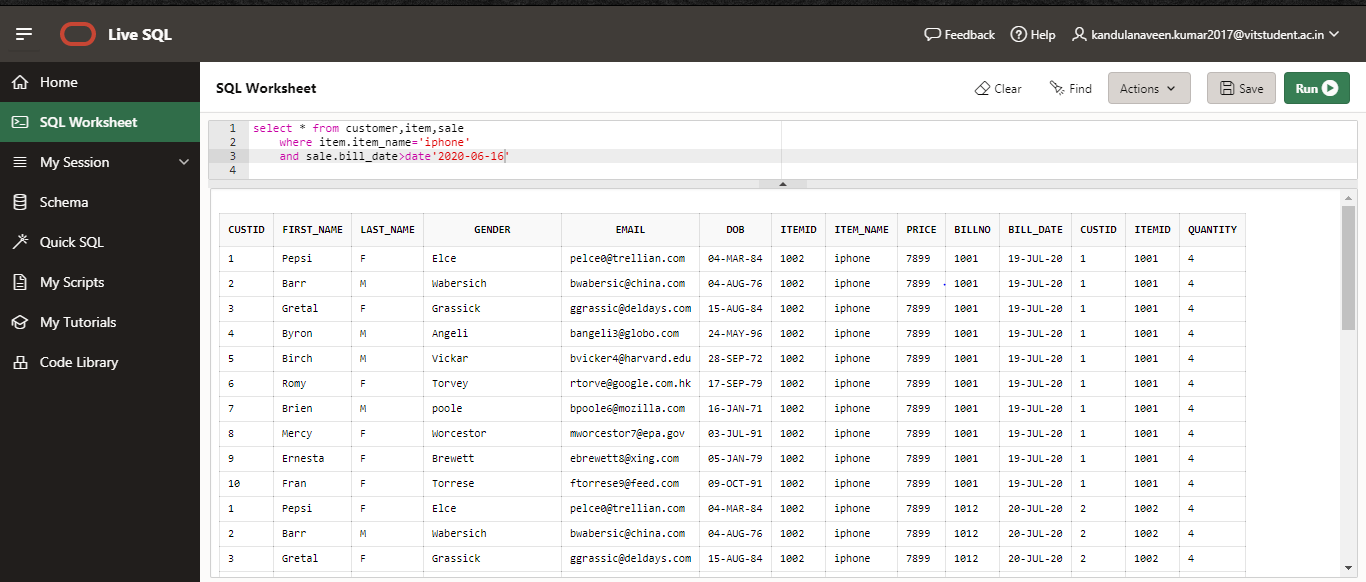
FROM TABLE(DBMS\_XPLAN.DISPLAY('plan\_table',null,'typical'));



* 1. queries: (Use order by appropriately) Insert few values according to query.
     1. Query 8: Select the customers detail who buy more than two ‘IPhone’ after 22-May-2020

ANS:

select \* from customer,item,sale where item.item\_name='iphone' and sale.bill\_date>date'2020-06-16'



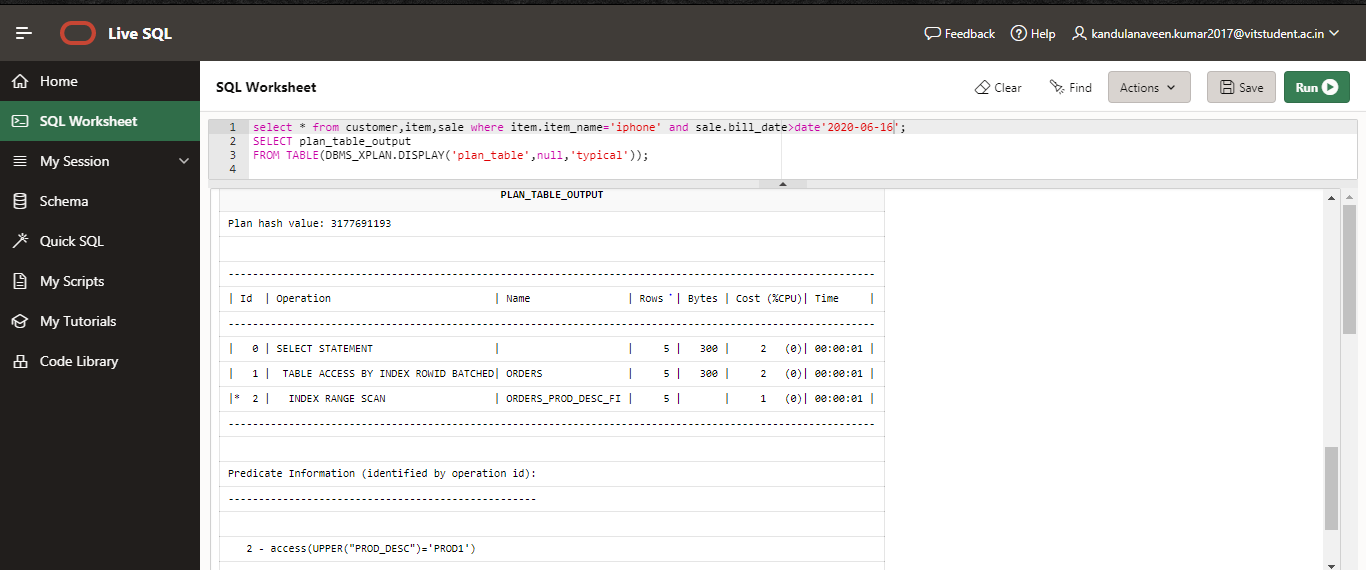
* + 1. Explain plan for Query 8 (Mention what JOIN is chosen by optimizer)

ANS:

select \* from customer,item,sale where item.item\_name='iphone' and sale.bill\_date>date'2020-06-16';

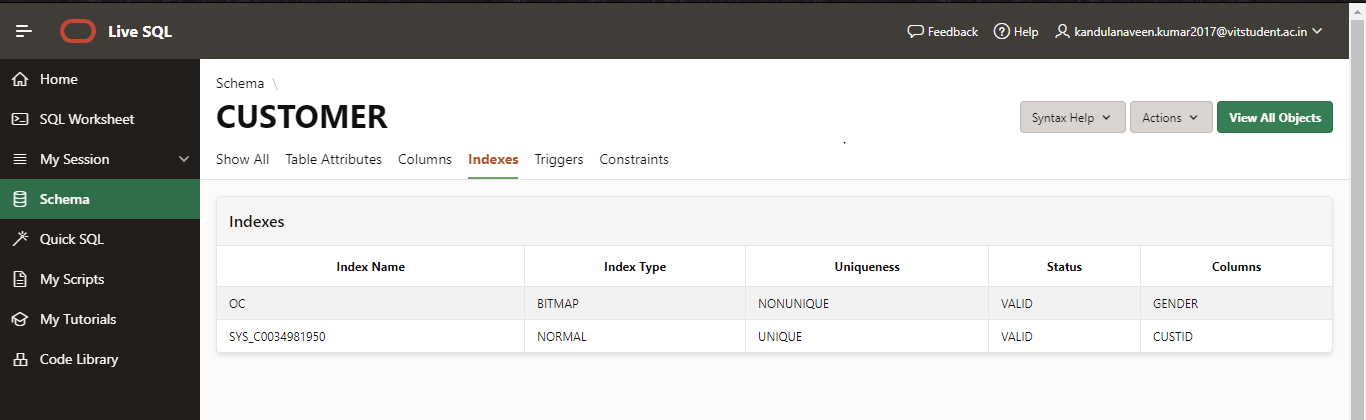
SELECT plan\_table\_output

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



* 1. Create a bitmap index on Gender.

ANS: create bitmap index oc on customer(gender);



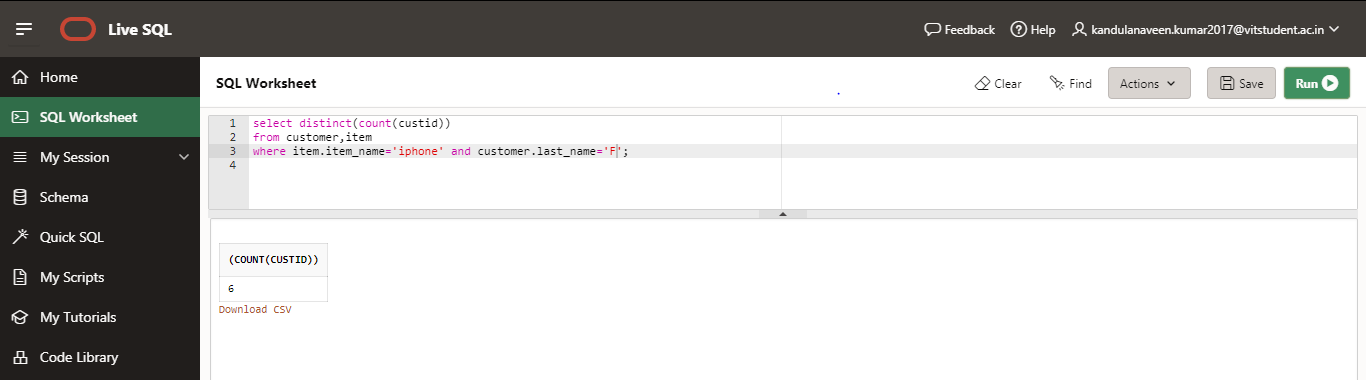
* + 1. Query 9: Count the number of customers purchased Iphone whose gender is Male

ANS:

select distinct(count(custid))

from customer,item

where item.item\_name='iphone' and customer.last\_name='F';



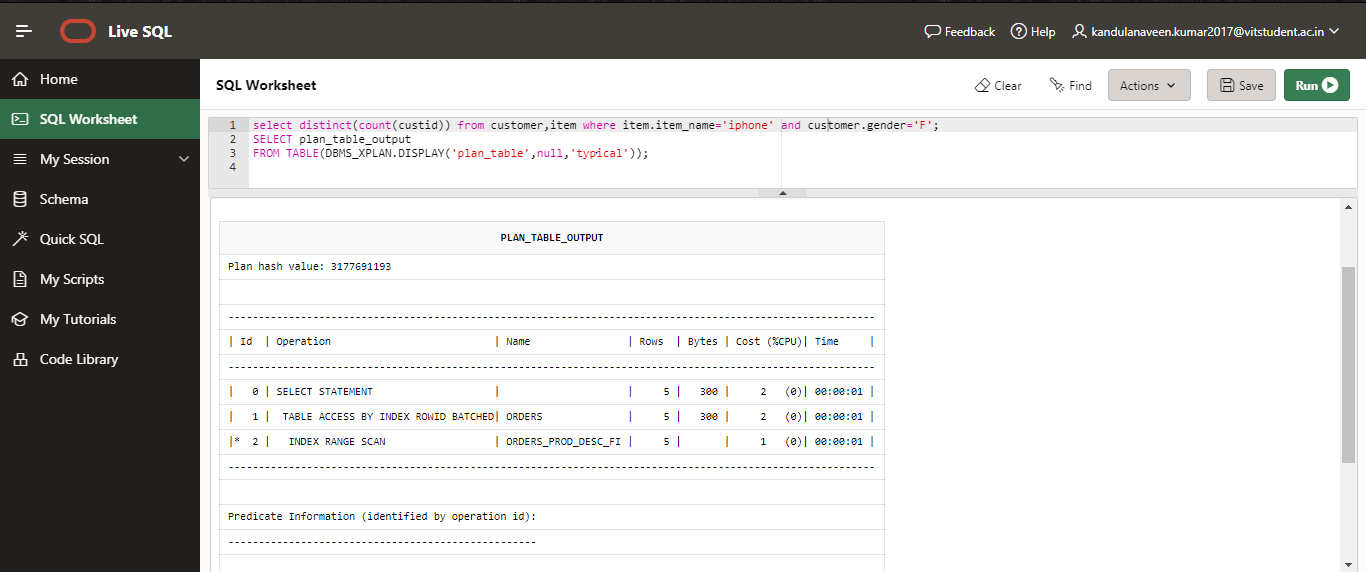
* + 1. Explain plan for Query9 (Mention what scan is chosen by optimizer)

ANS:

select distinct(count(custid)) from customer,item where item.item\_name='iphone' and customer.gender='F';

SELECT plan\_table\_output

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



* + 1. Drop the index.

ANS:

drop index oc

