

**Student Name : Santosh Acharya**

**Student Id : C0930325**

**Program Code : CSD 2206**

**Individual Project**

**Table Content**

<b>PART I.....</b>	<b>2-12</b>
<b>PART II.....</b>	<b>13-48</b>

## PART I

- ❖ DDL stands for Data Definition Language.
- ❖ DDL is the part of the SQL that helps to create, alerts and delete database.
- ❖ DML stands for Data Manipulation Language.
- ❖ DML is the SQL that helps to insert, update and delete database.
- ❖ CREATE is a syntax in the sql used to create the new table with in the database.
- ❖ Alter is use to change the table values after the table is created. It is table level query.
- ❖ Column level Constraint are used PRIMARY KEY and NOT NULL
- ❖ UNIQUE is used as the table level constraints
- ❖ SQL Query:

```
CREATE TABLE Student_0325(
    student_id int PRIMARY KEY,
    first_name varchar(50),
    last_name varchar(50) NOT NULL,
    contact_number varchar(10),
    address varchar(25),
    program_code varchar(15)
);
ALTER TABLE Student_0325
    ADD CONSTRAINTS student_uk
    UNIQUE(contact_number);
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows an Oracle connection named 'practicedatabase' and a selected connection named 'santoshDb' which contains tables, views, indexes, packages, procedures, functions, operators, queues, queue tables, triggers, types, sequences, materialized views, materialized view logs, synonyms, and public synonyms. The central area is the Worksheet tab, labeled 'SQL QUERY', containing the SQL code for creating and altering the 'Student\_0325' table. The right side shows the 'OUTPUT' tab, which displays the results of the execution: 'Table STUDENT\_0325 created.' and 'Table STUDENT\_0325 altered.' A red box highlights the output message 'Table STUDENT\_0325 created.'

- ❖ Insert is the DDL query used to enter the data in the database.
- ❖ We can insert the multiple value in the table using the insert.
- ❖ SQL Query

```
insert into Student_0325
```

```
Values(102,'Santosh','Acharya','4379551559','Mississauga','FSDM');
```

```
insert into Student_0325
```

```
Values(110,'Bhim Kumari','Lamsal','5876599999','Toronto','CPCM');
```

```
insert into Student_0325
```

```
Values(112,'Soniya','Bhattarai','6548974545','Alberta','QA');
```

```
insert into Student_0325
```

```
Values(105,'Roshan','Acharya','5489688888','Mississauga','FSDM');
```

```
insert into Student_0325
```

```
Values(122,'Ram','KC','4566561559','Mississauga','FSDM');
```

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Connections' sidebar lists 'practicedatabase' and 'santoshDb'. The 'santoshDb' connection is expanded to show 'Tables (Filtered)', 'Views', 'Indexes', 'Packages', 'Procedures', 'Functions', 'Operators', 'Queues', 'Queues Tables', 'Triggers', 'Types', 'Sequences', 'Materialized Views', 'Materialized View Logs', 'Synonyms', and 'Public Synonyms'. Below this is the 'Reports' sidebar with sections for 'All Reports', 'Analytic View Reports', 'Data Dictionary Reports', 'Data Modeler Reports', 'OLAP Reports', 'TimesTen Reports', and 'User Defined Reports'. The main workspace has a blue header bar labeled 'SQL QUERY'. The 'Worksheet' tab is active, containing the five 'insert into' statements. The 'Script Output' tab at the bottom shows the results: '1 row inserted.' repeated five times. A yellow box labeled 'OUTPUT' is overlaid on the bottom right of the script output area.

```

insert into Student_0325
values(102,'Santosh','Acharya','4379551559','Mississauga','FSDM');

insert into Student_0325
values(110,'Bhim Kumari','Lamsal','5876599999','Toronto','CPCM');

insert into Student_0325
values(112,'Soniya','Bhattarai','6548974545','Alberta','QA');

insert into Student_0325
values(105,'Roshan','Acharya','5489688888','Mississauga','FSDM');

insert into Student_0325
values(122,'Ram','KC','4566561559','Mississauga','FSDM');

```

- ❖ Select is use to display the value from the table.
- ❖ It helps to get the value from the table.
- ❖ SQL Query

**select \* from Student\_0325;**

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, showing a connection to 'santoshDb'. The central area has a 'Worksheet' tab open with the query `select * from Student_0325;`. A red box highlights this query. To the right, a blue box labeled 'SQL QUERY' is positioned above the results. Below the query is a table titled 'All Rows Fetched: 5 in 0.132 seconds'. The table has columns: STUDENT\_ID, FIRST\_NAME, LAST\_NAME, CONTACT\_NUMBER, ADDRESS, and PROGRAM\_CODE. The data is as follows:

STUDENT_ID	FIRST_NAME	LAST_NAME	CONTACT_NUMBER	ADDRESS	PROGRAM_CODE
1	102 Santosh	Acharya	4379551559	Mississauga	FSDM
2	110 Bhim Kumari	Lamsal	5876599999	Toronto	CPCM
3	112 Soniya	Bhattarai	6548974545	Alberta	QA
4	105 Roshan	Acharya	5489688888	Mississauga	FSDM
5	122 Ram	KC	4566561559	Mississauga	FSDM

Below the table, a yellow box labeled 'OUTPUT' contains the message 'All Rows Fetched: 5 in 0.132 seconds'.

- ❖ Update is the query used to update the value in the table.
- ❖ It need to specified the value which it want to update with the value that will be updated.
- ❖ SQL Query

**update Student\_0325 set address='Brampton', Program\_code ='SEO' where student\_id = 122;**

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, showing a connection to 'santoshDb'. The central area has a 'Worksheet' tab open with the query `update Student_0325 set address='Brampton', Program_code ='SEO' where student_id = 122;`. A blue box highlights this query. To the right, a blue box labeled 'SQL QUERY' is positioned above the results. Below the query is a message in a yellow box: 'Task completed in 0.037 seconds'. Another yellow box labeled 'OUTPUT' contains the message '1 row updated.'

View of the table after Update

❖ SQL Query

**select \* from Student\_0325;**

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, with 'santoshDb' selected. The central area has a red box around the 'Worksheet' tab, which contains the SQL statement: 'select \* from Student\_0325;'. Below it is the 'Query Result' tab, also with a red box, showing a table with 5 rows of student data:

STUDENT_ID	FIRST_NAME	LAST_NAME	CONTACT_NUMBER	ADDRESS	PROGRAM_CODE
1	102 Santosh	Acharya	4379551559	Mississauga	FSDM
2	110 Bhim Kumari	Lamsai	5876599999	Toronto	CPCM
3	112 Soniya	Bhattarai	6548974545	Alberta	QA
4	105 Roshan	Acharya	5489688888	Mississauga	FSDM
5	122 Ram	KC	4566561559	Brampton	SEO

A yellow box labeled 'OUTPUT' is at the bottom right.

❖ SQL Query to update next data

**Update Student\_0325 set first\_name ='Subash' where student\_id = 105;**

The screenshot shows the Oracle SQL Developer interface. The Connections tree on the left shows 'santoshDb' selected. The central area has a red box around the 'Worksheet' tab, which contains the SQL statement: 'update Student\_0325 set first\_name='Subash' where student\_id = 105;'. Below it is the 'Query Result' tab, also with a red box, showing the message: '1 row updated.' A yellow box labeled 'OUTPUT' is at the bottom right.

- ❖ SQL Query to view the data

**Select \* from Student\_0325;**

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' sidebar with 'practicedatabase' and 'santoshDb' selected. The main area has a 'Worksheet' tab open with the query `select * from Student_0325;`. Below it is a 'Query Result' tab showing the following data:

	STUDENT_ID	FIRST_NAME	LAST_NAME	CONTACT_NUMBER	ADDRESS	PROGRAM_CODE
1	102	Santosh	Acharya	4379551559	Mississauga	FSDM
2	110	Bhim Kumari	Lamsal	5876599999	Toronto	CPCM
3	112	Soniya	Bhattarai	6548974545	Alberta	QA
4	105	Subash	Acharya	5489688888	Mississauga	FSDM
5	122	Ram	KC	4566561559	Brampton	SEO

A blue box labeled 'SQL QUERY' highlights the worksheet area, and a yellow box labeled 'OUTPUT' highlights the query result area.

- ❖ Delete is used to remove the value from the table in the SQL query.
- ❖ Delete can be used to remove the single or multiple data in the table.
- ❖ SQL Query

**Delete from student\_0325 where student\_id = 122;**

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' sidebar with 'practicedatabase' and 'santoshDb' selected. The main area has a 'Worksheet' tab open with the query `delete from student_0325 where student_id = 122;`. Below it is a 'Query Result' tab showing the message `Task completed in 0.035 seconds` and `1 row deleted.`. A red box highlights the 'Query Result' tab, and a yellow box labeled 'OUTPUT' highlights the message.

## View of the table

- ❖ SQL Query

Select \* from Student\_0325;

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, showing 'practicedatabase' and 'santoshDb'. The 'Tables (Filtered)' node under 'santoshDb' is expanded, showing various table types like Tables, Views, Indexes, etc. The central area has a red box around the 'Worksheet' tab which contains the SQL query: 'select \* from Student\_0325;'. To the right of the worksheet is a blue box labeled 'SQL QUERY'. Below the worksheet is the 'Script Output' and 'Query Result' tabs, with the 'Query Result' tab active. It displays a table with four rows of student data:

STUDENT_ID	FIRST_NAME	LAST_NAME	CONTACT_NUMBER	ADDRESS	PROGRAM_CODE
1	102 Santosh	Acharya	4379551559	Mississauga	FSDM
2	110 Bhim Kumari	Lamsal	5876599999	Toronto	CPCM
3	112 Soniya	Bhattarai	6548974545	Alberta	QA
4	105 Subash	Acharya	5489688888	Mississauga	FSDM

To the right of the result table is a yellow box labeled 'OUTPUT'.

- ❖ Commit is used to save the checkpoint in the table
- ❖ It is a point where the table will rollback in the sql.
- ❖ After deleting one data commit was used to save the table.
- ❖ SQL Query

Commit;

The screenshot shows the Oracle SQL Developer interface. The 'Worksheet' tab contains the SQL command 'commit;'. A red box highlights this command. To the right of the worksheet is a blue box labeled 'SQL QUERY'. Below the worksheet is the 'Script Output' and 'Query Result' tabs, with the 'Query Result' tab active. It shows the message 'Commit complete.' A yellow box to the right is labeled 'OUTPUT'.

❖ SQL Query

Delete from student\_0325 where contact\_number = '4379551559';

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, showing a connection to 'santoshDb'. The central area has a 'Worksheet' tab open with the query: 'delete from student\_0325 where contact\_number = '4379551559';'. Below the worksheet is a 'Script Output' tab showing the message '1 row deleted.' A yellow box labeled 'OUTPUT' is overlaid on the 'Script Output' tab. The top right corner of the interface has a blue bar labeled 'SQL QUERY'.

View table after delete

❖ SQL Query

Select \* from Student\_0325

The screenshot shows the Oracle SQL Developer interface. The 'Worksheet' tab contains the query 'select \* from Student\_0325;'. Below it is a 'Query Result' tab displaying a table with three rows of data. A yellow box labeled 'OUTPUT' is overlaid on the 'Query Result' tab. The top right corner of the interface has a blue bar labeled 'SQL QUERY'.

STUDENT_ID	FIRST_NAME	LAST_NAME	CONTACT_NUMBER	ADDRESS	PROGRAM_CODE
1	110 Bhim Kumari	Lamsal	5876599999	Toronto	CPCM
2	112 Soniya	Bhattarai	6548974545	Alberta	QA
3	105 Subash	Acharya	5489688888	Mississauga	FSDM

- ❖ Rollback is used to move the table to the previous commit point.
- ❖ It helps to recover the data in case there is any error.
- ❖ SQL Query

### Rollback;

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows a connection to 'santoshDb'. The central area has a red box around the 'SQL QUERY' tab, which contains the command 'rollback;'. Below it, the 'OUTPUT' tab shows the message 'Rollback complete.'.

### View after the rollback

- ❖ SQL Query

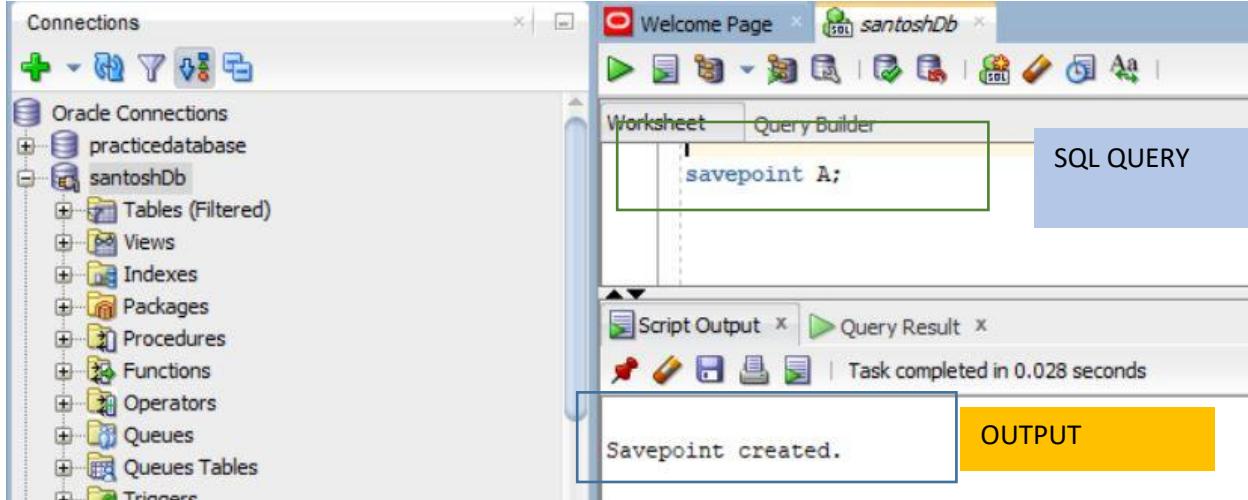
### Select \* from Student\_0325;

The screenshot shows the Oracle SQL Developer interface. The 'SQL QUERY' tab contains the command 'select \* from Student\_0325;'. The 'OUTPUT' tab displays a table with four rows of student data:

STUDENT_ID	FIRST_NAME	LAST_NAME	CONTACT_NUMBER	ADDRESS	PROGRAM_CODE
1	102 Santosh	Acharya	4379551559	Mississauga	FSDM
2	110 Bhim Kumari	Lamsal	5876599999	Toronto	CPCM
3	112 Soniya	Bhattarai	6548974545	Alberta	QA
4	105 Subash	Acharya	5489688888	Mississauga	FSDM

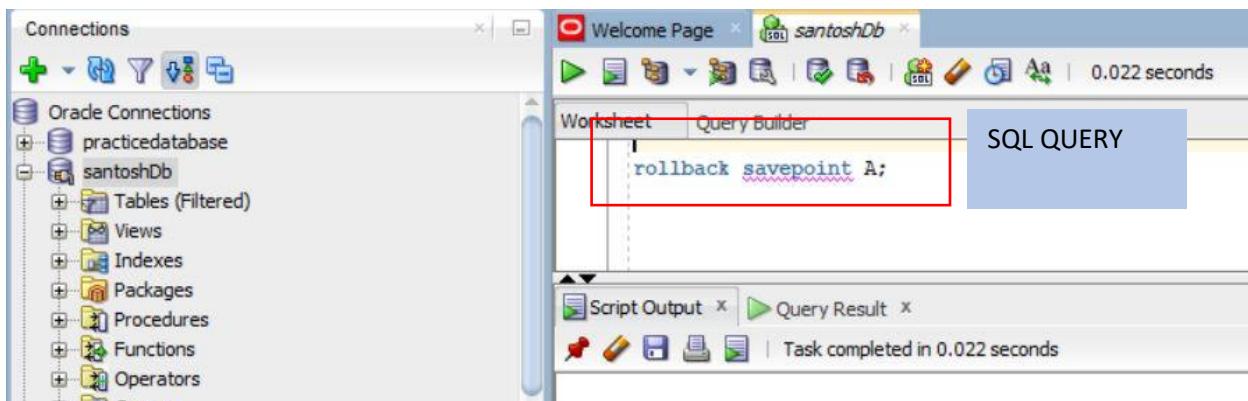
- ❖ Savepoint is similar to commit that is the check point that is used to save the different point in the table.
- ❖ SQL Query:

**Savepoint A;**



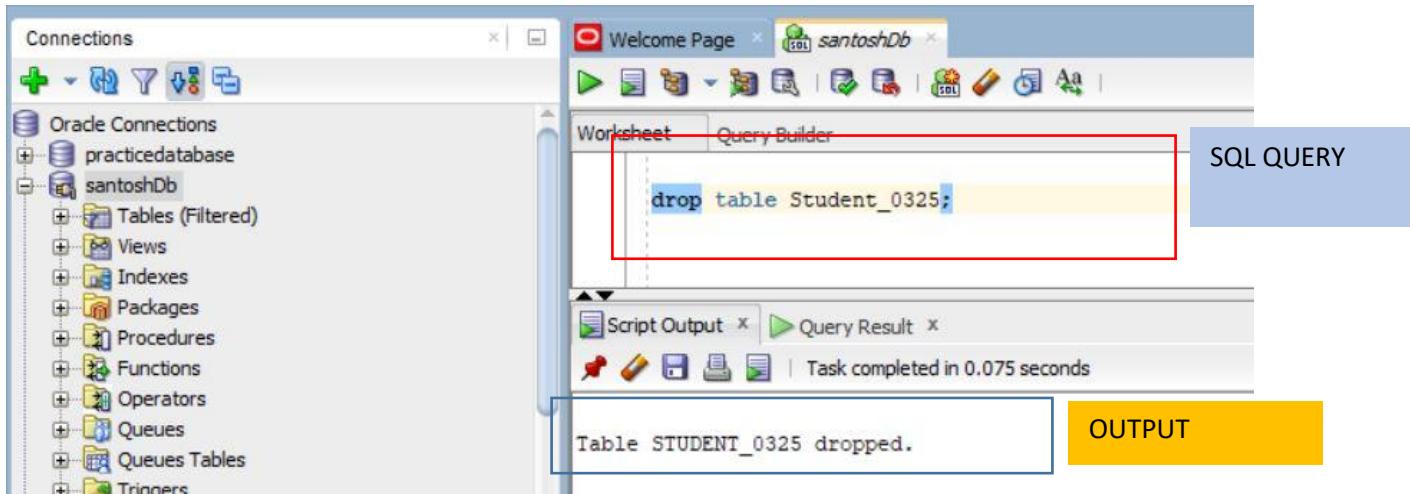
- ❖ SQL Query to rollback

**Rollback savepoint A;**



- ❖ Drop is used to delete the whole table from the database.
- ❖ It will not only remove the data but also the all structure of the table in the database.
- ❖ SQL Query

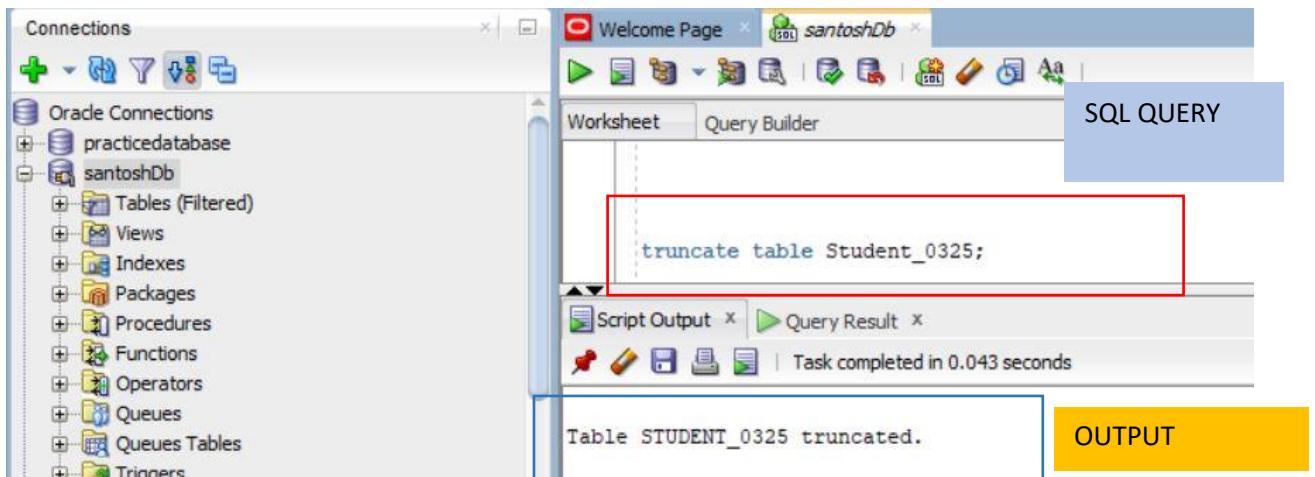
#### **Drop table Student\_0325;**



The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows an Oracle connection named 'practicedatabase' and a local connection named 'santoshDb'. The 'Tables (Filtered)' node under 'santoshDb' is selected. In the center, the Worksheet tab contains the SQL query: `drop table Student_0325;`. This query is highlighted with a red rectangle. To the right, the SQL QUERY panel shows the query being processed. Below it, the Script Output and Query Result tabs are visible, along with a message: "Task completed in 0.075 seconds". The OUTPUT panel at the bottom displays the result: "Table STUDENT\_0325 dropped." A yellow box highlights the output message.

- ❖ Truncate is the query which will remove the data from the table but not the structure of the table.
- ❖ It will remove all the data.

#### **Truncate table Student\_0325**



The screenshot shows the Oracle SQL Developer interface. The Connections tree is identical to the previous one. In the Worksheet tab, the SQL query is `truncate table Student_0325;`, which is highlighted with a red rectangle. The SQL QUERY panel shows the query being processed. The Script Output and Query Result tabs are visible, with the message "Task completed in 0.043 seconds". The OUTPUT panel at the bottom displays the result: "Table STUDENT\_0325 truncated." A yellow box highlights the output message.

## View After the Truncate Query

Select \* from Student\_0325;

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows a selected connection to 'santoshDb'. The central area has a 'Worksheet' tab open with the SQL query: 'select \* from Student\_0325;'. To the right of the worksheet is a 'SQL QUERY' panel. Below the worksheet is a 'Script Output' tab and a 'Query Result' tab. The 'Query Result' tab is active, displaying the results of the query. A red box highlights the header row of the result set, which includes columns: STUDENT..., FIRST\_NA..., LAST\_NAME, CONTACT..., ADDRESS, and PROGRA... . The message 'All Rows Fetched: 0 in 0.005 seconds' is visible above the result grid.

OUTPUT

## PART II

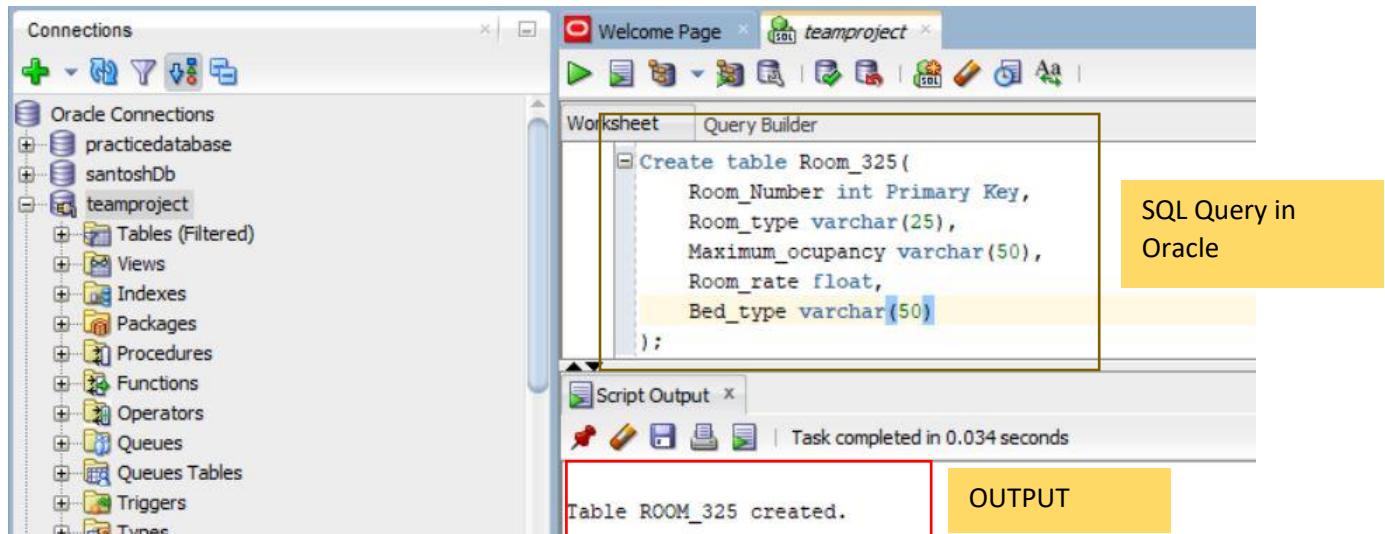
### ALL table that has been used in the Query

- ❖ The Create is use to create the table.
- ❖ Create is the sql query used to add the new table in the database.
- ❖ It is DDL based query in the oracle.
  
- ❖ SQL Query

```
Create table Room_325(  
    Room_Number int Primary Key,  
    Room_type varchar(25),  
    Maximum_occupancy varchar(50),  
    Room_rate float,  
    Bed_type varchar(50)  
)
```

- ❖ SQL Query Explanation

Create is used to make the table name Room\_325 with Room\_number as the primary key.



- ❖ Inserted into Room
- ❖ Insert is used enter the data into the database.
- ❖ It is DDL based query
- ❖ SQL Query

```
insert into Room_325
values(101,'Single Room','5',50,'Single Bed');
insert into Room_325
values(102,'Double Room','2',50,'Double Bed');
```

**SQL Query in Oracle**

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' sidebar with 'teamproject' selected. The main area has a 'Worksheet' tab open with the following SQL query:

```
insert into Room_325
values(101,'Single Room','5',50,'Single Bed');
```

This query is highlighted with a red box. Below the worksheet is a 'Script Output' window showing the result:

```
1 row inserted.
```

A yellow box labeled 'OUTPUT' is placed over the 'Script Output' window.

**SQL Query in Oracle**

The screenshot shows the Oracle SQL Developer interface with the same setup as the previous one. The 'teamproject' connection is selected in the 'Connections' sidebar. The 'Worksheet' tab contains the following SQL query:

```
insert into Room_325
values(102,'Double Room','2',50,'Double Bed');
```

This query is highlighted with a red box. The 'Script Output' window below shows the result:

```
1 row inserted.
```

A yellow box labeled 'OUTPUT' is placed over the 'Script Output' window.

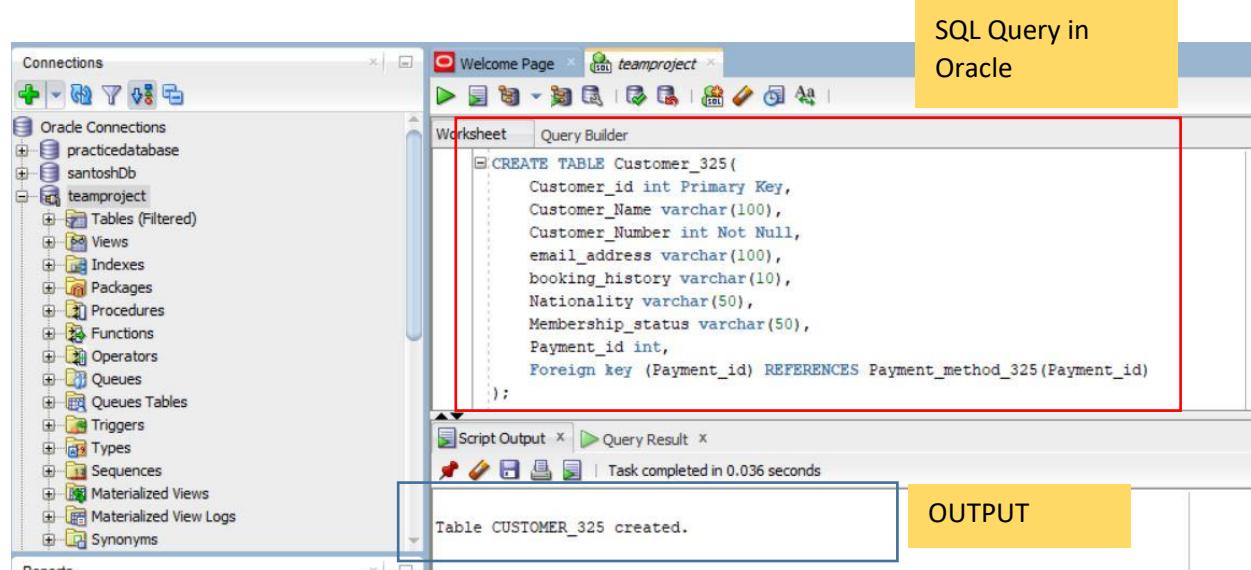
### **Customer\_325 table**

- ❖ Foreign key is used to get the column that is been assigned to the other table in the same database.
- ❖ References is used to refer the table from where the foreign key is selected.
  
- ❖ SQL Query Explanation :

Customer\_325 with the primary key customer\_id , customer\_number is also not null and Payment\_id is the foreign key used from the payment\_method\_325.

- ❖ SQL Query

```
CREATE TABLE Customer_325(
    Customer_id int Primary Key,
    Customer_Name varchar(100),
    Customer_Number int Not Null,
    email_address varchar(100),
    booking_history varchar(10),
    Nationality varchar(50),
    Membership_status varchar(10),
    Payment_id int,
    Foreign key (Payment_id) REFERENCES Payment_method_325(Payment_id)
);
```

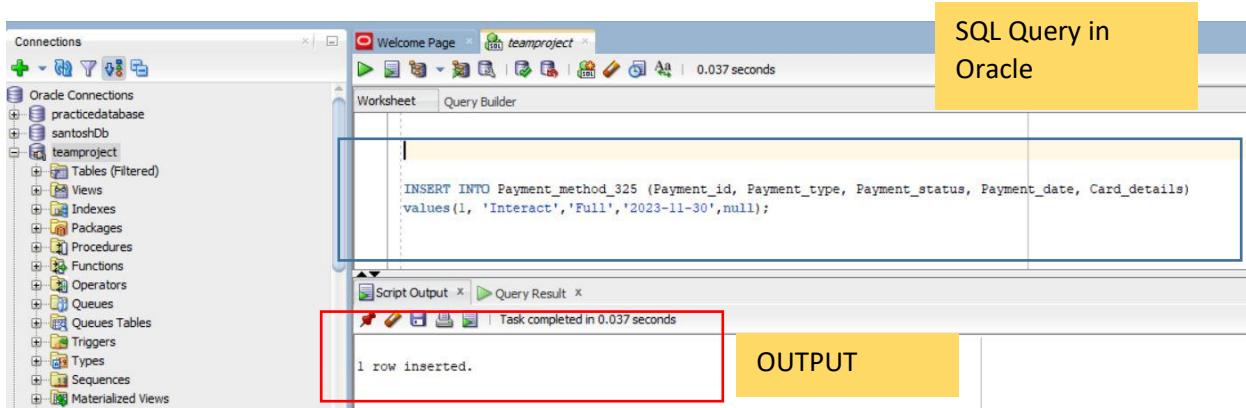


The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' sidebar with 'practicedatabase' and 'teamproject' selected. The 'teamproject' connection is expanded to show 'Tables (Filtered)', 'Views', 'Indexes', 'Packages', 'Procedures', 'Functions', 'Operators', 'Queues', 'Queues Tables', 'Triggers', 'Types', 'Sequences', 'Materialized Views', 'Materialized View Logs', and 'Synonyms'. The main workspace is titled 'Worksheet' and contains the SQL code for creating the 'Customer\_325' table. A yellow box labeled 'SQL Query in Oracle' highlights the query. The code is enclosed in a red box. Below the worksheet is the 'Script Output' tab, which displays the message 'Table CUSTOMER\_325 created.' A yellow box labeled 'OUTPUT' highlights this message. The top status bar shows 'Welcome Page' and 'teamproject'.

### **Inserted into Customer\_325**

- ❖ Values are inserted according to the data types that has been created using the table.
- ❖ SQL Query

```
insert into Customer_325  
values(201,'Santosh Acharya',4379551559,'acharyaanish16@gmail.com',null,'Nepali','Full  
Member',1);
```



### **Payment\_method\_325**

```
CREATE TABLE Payment_method_325(  
    Payment_id int Primary Key,  
    Payment_type varchar(50),  
    Payment_status varchar(50) Not Null,  
    Payment_date date,  
    Card_details varchar(50)  
)
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree displays three databases: practicedatabase, santoshDb, and teamproject. The teamproject database is expanded, showing Tables (Filtered), Views, Indexes, Packages, Procedures, Functions, Operators, Queues, Queues Tables, Triggers, Types, Sequences, Materialized Views, and Materialized View Logs. In the center, the Worksheet tab is active, containing the following SQL code:

```
CREATE TABLE Payment_method_325
(
    Payment_id int Primary Key,
    Payment_type varchar(50),
    Payment_status varchar(50) Not Null,
    Payment_date varchar(50),
    Card_details varchar(50)
);
```

Below the code, the Script Output tab shows the message "Table PAYMENT\_METHOD\_325 created." highlighted with a red box.

Inserted into payment\_method\_325

```
insert into Payment_method_325
values(1, 'Interact','Full Payment','2023-11-31',null);
```

The screenshot shows the Oracle SQL Developer interface. The Connections tree on the left shows the teamproject database selected. The Worksheet tab contains the following SQL code:

```
INSERT INTO Payment_method_325 (Payment_id, Payment_type, Payment_status, Payment_date, Card_details)
values(1, 'Interact','Full','2023-11-30',null);
```

The Script Output tab below shows the message "1 row inserted." highlighted with a green box.

```
insert into Payment_method_325
values(2, 'Debit Card','No Payment','2023-12-31','RBC');
insert into Payment_method_325
values(3, 'Credit Card','Partial Payment','2024-01-23','CBIC');
insert into Payment_method_325
values(4, 'Google pay','No Payment','2024-03-15',null);
insert into Payment_method_325
values(5, 'Bank Transfer','Partial Payment','2024-01-23','TD');
```

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Connections' sidebar lists 'teamproject' as the current connection, which contains tables, views, indexes, packages, procedures, functions, operators, queues, triggers, types, sequences, materialized views, materialized view logs, and synonyms. Below it, the 'Reports' sidebar lists various report categories. The main workspace consists of two tabs: 'Worksheet' and 'Query Builder'. The 'Worksheet' tab contains the following SQL code:

```

INSERT INTO Payment_method_325 (Payment_id, Payment_type, Payment_status, Payment_date, Card_details)
values(1, 'Interact','Full','2023-11-30',null);

insert into Payment_method_325
values(2, 'Debit-Card','No Payment','2023-12-31','RBC');

insert into Payment_method_325
values(3, 'Credit Card','Partial Payment','2024-01-23','CBIC');

insert into Payment_method_325
values(4, 'Google pay','No Payment','2024-03-15',null);

```

The 'Script Output' tab at the bottom shows the results of the execution:

```

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

```

A green box highlights the 'Worksheet' tab, and a red box highlights the 'Script Output' tab.

## Inventory\_325

- ❖ It is the table used to store the inventory related information in the hotel reservation system.
- ❖ SQL Query :

```

create table inventory_325(
    inventory_id int Primary key,
    inventory_name varchar(100),
    inventory_rate float,
    inventory_quantity int,
    inventory_description varchar(100),
    Purchase_date varchar(100)

);

```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree displays several databases, including 'teamproject'. The central area is a Worksheet window titled 'Worksheet' with the tab 'Query Builder' selected. A code editor contains the SQL command to create the 'inventory\_325' table:

```
create table inventory_325(
    inventory_id int Primary key,
    inventory_name varchar(100),
    inventory_rate float,
    inventory_quantity int,
    inventory_description varchar(100),
    Purchase_date varchar(20)
);
```

Below the code editor, the status bar indicates "Task completed in 0.043 seconds". In the bottom right corner of the worksheet area, a message box displays the text "Table INVENTORY\_325 created.".

### Supplier\_325

```
create table Supplier_325(
    Supplier_id int Primary Key,
    Product_type varchar(100),
    Supplier_address varchar(100),
    Supplier_contact varchar(10)
);
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree displays several databases, including 'teamproject'. The central area is a Worksheet window titled 'Worksheet' with the tab 'Query Builder' selected. A code editor contains the SQL command to create the 'Supplier\_325' table:

```
create table Supplier_325(
    Supplier_id int Primary Key,
    Product_type varchar(100),
    Supplier_address varchar(100),
    Supplier_contact varchar(10)
);
```

Below the code editor, the status bar indicates "Task completed in 0.099 seconds". In the bottom right corner of the worksheet area, a message box displays the text "Table SUPPLIER\_325 created.".

## Category A

1

- ❖ Select is the query used to view the data in the table. It is used to select and view the particular table.
- ❖ SQL Query

```
select Room_Number as Santosh_Room_Number_0930325, Room_type, Maximum_occupancy,  
Room_rate, Bed_type from Room_325;
```

- ❖ SQL Query Explanation

In the sql query room\_number, room type, maximum\_occupancy, room\_rate, bed\_type is selected from the room\_325 table.

SQL Query in Oracle

SANTOSH_ROOM_NUMBER_0930325	ROOM_TYPE	MAXIMUM_OCCUPANCY	ROOM_RATE	BED_TYPE
1	101 Single Room	5	50	Single Bed
2	102 Double Room	2	50	Double Bed
3	103 Suite	5	50	Queen
4	104 Deluxe Room	6	100	King
5	105 Quad room	4	150	Large King
6	106 Triple room	3	65	Queen
7	107 Penthouse suites	5	55	King
8	108 Junior Suites	4	35	Double Bed
9	109 Connecting rooms	3	45	King

OUTPUT

2

- ❖ NOT is the operator used to denote the select query that will select all other value except the assigned value.
- ❖ SQL Query :

```
Select * from Customer_325 where NOT Nationality = "American"
```

- ❖ SQL Query Explanation :

In the query the table will show the value except the value that contain Nationality = "American"

CUSTOMER_ID	CUSTOMER_NAME	CUSTOMER_NUMBER	EMAIL_ADDRESS	BOOKING_HISTORY	NATIONALITY	MEMBERSHIP_STATUS	PAYMENT_ID
1	201 Santosh Acharya	4379551559	acharyanish16@gmail.com	(null)	Nepali	Full Member	1
2	202 Anish Thapa	5365962458	anishthapa@gmail.com	2	Canadian	Expired	2
3	203 Bhim Kumar Lamsal	5584166465	bhimkumarilamsal5@gmail.com	(null)	Nepali	(null)	3
4	205 David Becham	7589351234	davidbecham5@gmail.com	(null)	British	Full Member	5

- ❖ Multiplication is a mathematical operation used to multiply numeric values, typically within SELECT statements or as part of UPDATE or INSERT statements.
- ❖ SQL Query:

```
SELECT Room_Number as Room_Number_325, Room_rate, Room_rate * 0.5 AS discount FROM Room_325;
```

- ❖ SQL Query Explanation:

In the query the multiplication is used to get the discount rate of 50% to the room rate from room\_325

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections sidebar with Oracle Connections expanded, showing databases like practicedatabase, santoshDb, and teamproject. The teamproject connection is selected. The main area has a Worksheet tab open with the query:

```
SELECT Room_Number as Room_Number_325, Room_rate, Room_rate * 0.5 AS discount FROM Room_325;
```

Below the worksheet is a Script Output tab showing the results of the query. A red box highlights the results table:

ROOM_NUMBER_325	ROOM_RATE	DISCOUNT
1	101	50
2	102	50
3	103	50
4	104	100
5	105	150
6	106	65
7	107	55
8	108	35
9	109	45

## 4

- ❖ Concatenation is used to combine the string together. It is useful to join the string
- ❖ SQL Query :

```
SELECT Room_Number as Room_Number_325, Room_type || '-' || Bed_type AS  
Room_Details_325 FROM Room_325;
```

- ❖ SQL query explanation:

In the query, room type and bed type is connected with each other with the ‘-’ sign.

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows several databases, with 'teamproject' selected. The Worksheet tab contains the query: `SELECT Room_Number as Room_Number_325, Room_type || '-' || Bed_type AS Room_Details_325 FROM Room_325;`. The Script Output tab shows the results in a table:

	ROOM_NUMBER_325	ROOM_DETAILS_325
1		101 Single Room - Single Bed
2		102 Double Room - Double Bed
3		103 Suite - Queen
4		104 Deluxe Room - King
5		105 Quad room - Large King
6		106 Triple room - Queen
7		107 Penthouse suites - King
8		108 Junior Suites - Double Bed
9		109 Connecting rooms - King

## 5

- ❖ DISTINCT is use to return the different value.
- ❖ SQL Query

```
SELECT DISTINCT membership_status as Membership_Status_325 FROM Customer_325;
```

- ❖ SQL Explanation:

In the query, the membership status with different value is selected.

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows several databases, with 'teamproject' selected. The Worksheet tab contains the query: `SELECT DISTINCT membership_status as Membership_Status_325 FROM Customer_325;`. The Script Output tab shows the results in a table:

	MEMBERSHIP_STATUS_325
1	Full Member
2	Expired
3	(null)

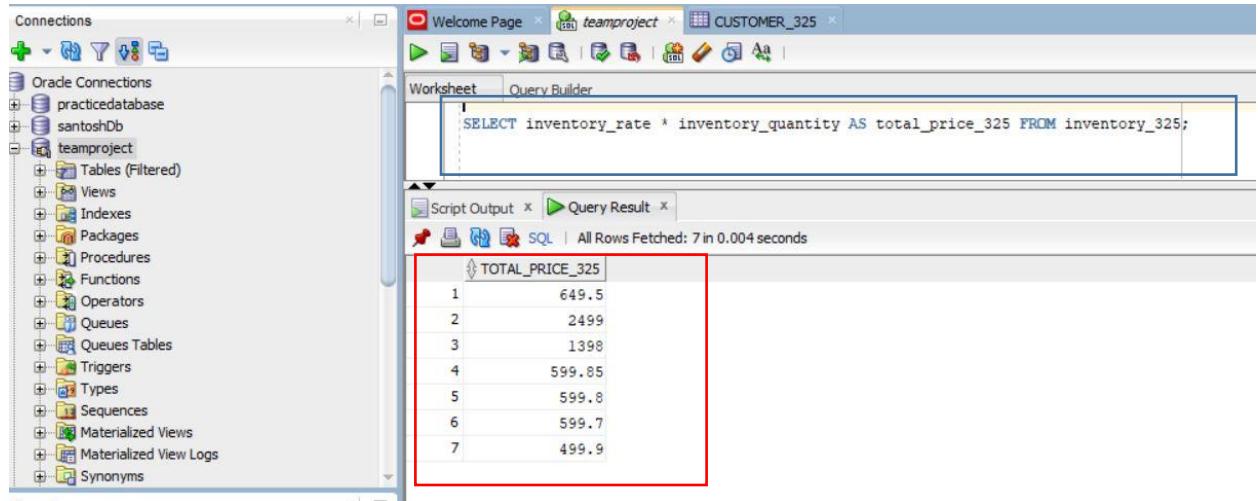
## 6

- ❖ Addition, Subtraction , Multiplication is the mathematical operand used with the select in the sql query.
- ❖ SQL Query

**Select inventory\_rate \* inventory\_quantity as total\_price\_325 from inventory\_325;**

- ❖ SQL Query Explanation:

In the query the inventory rate is multiplied with the inventory quantity to get the total\_price from the inventory\_325 table.



The screenshot shows the Oracle SQL Developer interface. On the left is the Object Navigator pane, which lists various database objects under the 'teamproject' connection. In the center is the Worksheet pane where the SQL query is typed. Below it is the Script Output pane, which displays the results of the query. A red box highlights the results table, which contains the following data:

	TOTAL_PRICE_325
1	649.5
2	2499
3	1398
4	599.85
5	599.8
6	599.7
7	499.9

## Category B

1

- ❖ Comparison operator helps to know the certain condition in the database and select the data according to the condition.
- ❖ Greater than sign will compare the value and determine if the value is greater than the given condition
- ❖ SQL query explanation :

In the query the room rate greater than 40 will be selected from the room\_325 table.

- ❖ SQL Query

```
select Room_Number as Santosh_Room_Number_0930325, Maximum_occupancy, Room_rate from  
Room_325 where room_rate > 40;
```

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' sidebar with 'practicedatabase' and 'santoshDb' selected. The main area has a 'Worksheet' tab open with the query: 'select Room\_Number as Santosh\_Room\_Number\_0930325, Maximum\_occupancy, Room\_rate from Room\_325 where room\_rate > 40;'. Below the worksheet is a 'Query Result' tab showing the output:

	SANTOSH_ROOM_NUMBER_0930325	MAXIMUM_OCCUPANCY	ROOM_RATE
1	101 5	50	
2	102 2	50	
3	103 5	50	
4	104 6	100	
5	105 4	150	
6	106 3	65	
7	107 5	55	
8	109 3	45	

2

- ❖ Between Operands will help to get the value within a given range.
- ❖ SQL Query

```
SELECT Room_Number as Room_Number_0930325, Room_rate, Room_type, maximum_occupancy,  
bed_type FROM Room_325 WHERE Room_rate BETWEEN 50 AND 100;
```

- ❖ SQL Query explanation:

In the query the Room Number, Room rate, Room type , maximum occupancy and bed type is selected from the room\_325 table where room rate is between 50 and 100.

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' sidebar with 'teamproject' selected. The main area has a 'Worksheet' tab open with the following SQL query:

```
SELECT Room_Number as Room_Number_0930325, Room_rate, Room_type, maximum_occupancy, bed_type
FROM Room_325
WHERE Room_rate BETWEEN 50 AND 100;
```

Below the worksheet is a 'Script Output' tab showing the results:

ROOM_NUMBER_0930325	ROOM_RATE	ROOM_TYPE	MAXIMUM_OCCUPANCY	BED_TYPE
1	101	50 Single Room	5	Single Bed
2	102	50 Double Room	2	Double Bed
3	103	50 Suite	5	Queen
4	104	100 Deluxe Room	6	King
5	106	65 Triple room	3	Queen
6	107	55 Penthouse suites	5	King

### 3

- ❖ IN operands helps to specify multiple values in a Where clause. It is similar of using multiple OR condition
- ❖ SQL Query

```
SELECT Customer_id AS Customer_id_0930325, Customer_Name, Customer_Number,
email_address, booking_history, Nationality, Membership_status, Payment_id
FROM Customer_325
WHERE Nationality IN ('American', 'British', 'Canadian');
```

- ❖ SQL Explanation :

In the query the customer id, customer name, customer number, email address, booking history , membership status and payment id is selected when nationality is either American, british or Canadian.

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' sidebar with 'teamproject' selected. The main area has a 'Worksheet' tab open with the following SQL query:

```
SELECT Customer_id AS Customer_id_0930325, Customer_Name, Customer_Number, email_address, booking_history, Nationality, Membership_status, Payment_id
FROM Customer_325
WHERE Nationality IN ('American', 'British', 'Canadian');
```

Below the worksheet is a 'Script Output' tab showing the results:

CUSTOMER_ID_0930325	CUSTOMER_NAME	CUSTOMER_NUMBER	EMAIL_ADDRESS	BOOKING_HISTORY	NATIONALITY	MEMBERSHIP_STATUS	PAYMENT_ID
1	202 Anish Thapa	5365962458	anishthapa@gmail.com	2	Canadian	Expired	2
2	204 Roshan Acharya	4372415609	roshanacharya050@gmail.com	5	American	Full Member	4
3	205 David Becham	7589351234	davidbecham5@gmail.com	(null)	British	Full Member	5

- ❖ Like operand is used to search the similar pattern in the table. Percentage % and Underscore \_ are often used with like operand.
- ❖ SQL Query

```
SELECT Customer_id AS Customer_id_0930325, Customer_Name, Customer_Number, Nationality,
Membership_status, Payment_id
FROM Customer_325
WHERE Customer_Name LIKE '%Sa%';
```

- ❖ SQL Explanation:

In the query, customer id, customer name, customer number, nationality, membership status and payment id is selected from the customer\_325 where customer name is like '%Sa%'.

CUSTOMER_ID_0930325	CUSTOMER_NAME	CUSTOMER_NUMBER	NATIONALITY	MEMBERSHIP_STATUS	PAYMENT_ID
1	201 Santosh Acharya	4379551559	Nepali	Full Member	1

- ❖ = sign is used to compare the exact value within the table.
- ❖ It will provide the value that match the condition.
- ❖ SQL Query

```
SELECT Room_Number as Room_Number_0930325, Room_type, Room_rate,
maximum_occupancy, bed_type
FROM Room_325
WHERE Room_type = 'Single Room';
```

ROOM_NUMBER_0930325	ROOM_TYPE	ROOM_RATE	MAXIMUM_OCCUPANCY	BED_TYPE
1	101 Single Room	50.5	5	Single Bed

- ❖ != also Known as Not Equal to is used select the value that is not equal to the given value.
- ❖ It is the comparison operand used in the sql query.
- ❖ SQL Query

```
SELECT Room_Number as Room_Number_0930325, Room_type, Room_rate,
maximum_occupancy,bed_type
FROM Room_325
WHERE Room_rate != 50;
```

- ❖ SQL Query Explanation:

IN the query the given value is used to select the room number, room type, room rate, maximum occupancy and bed type from room\_325 table that is not equal to 50.

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' sidebar with 'teamproject' selected. The main area has a 'Worksheet' tab open with the following SQL code:

```
SELECT Room_Number as Room_Number_0930325, Room_type, Room_rate, maximum_occupancy,bed_type
FROM Room_325
WHERE Room_rate != 50;
```

Below the worksheet is a 'Query Result' tab showing the output of the query:

ROOM_NUMBER_0930325	ROOM_TYPE	ROOM_RATE	MAXIMUM_OCCUPANCY	BED_TYPE
1	104 Deluxe Room	100 6	King	
2	105 Quad room	150 4	Large King	
3	106 Triple room	65 3	Queen	
4	107 Penthouse suites	55 5	King	
5	108 Junior Suites	35 4	Double Bed	
6	109 Connecting rooms	45 3	King	

The status bar at the bottom indicates "All Rows Fetched: 6 in 0.012 seconds".

## Category C

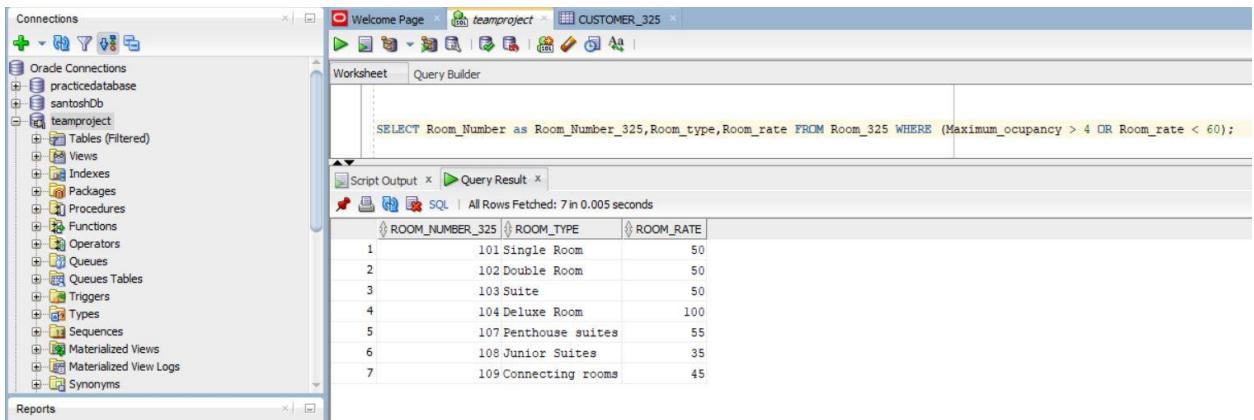
1

- ❖ Logical Operator are the operator test for the truth of some condition.
- ❖ SQL Query

```
SELECT Room_Number as Room_Number_325,Room_type,Room_rate FROM Room_325 WHERE  
(Maximum_occupancy > 4 OR Room_rate < 60);
```

- ❖ SQL query Explanation:

In the query, the room number, room type, room rate from room\_325 where maximum occupancy is greater than 4 or room rate is less than 60.



The screenshot shows the Oracle SQL Developer interface. On the left is the Object Navigator pane, which lists various database objects under the 'teamproject' schema, including Tables (Filtered), Views, Indexes, Procedures, Functions, Operators, Queues, Sequences, and Materialized Views. The central workspace contains a 'Worksheet' tab with the SQL query: 'SELECT Room\_Number as Room\_Number\_325,Room\_type,Room\_rate FROM Room\_325 WHERE (Maximum\_occupancy > 4 OR Room\_rate < 60);'. Below the worksheet is a 'Script Output' tab showing the execution results. A table titled 'ROOM\_NUMBER\_325' is displayed with the following data:

ROOM_NUMBER_325	ROOM_TYPE	ROOM_RATE
1	101 Single Room	50
2	102 Double Room	50
3	103 Suite	50
4	104 Deluxe Room	100
5	107 Penthouse suites	55
6	108 Junior Suites	35
7	109 Connecting rooms	45

2

- ❖ Order By is use to sort the result in ascending or descending order.
- ❖ It will display the result in the either from low value to high or vice-versa.
- ❖ SQL Query

```
SELECT Room_Number as Room_Number_325, Room_type, Maximum_occupancy, Room_rate,  
Bed_type FROM Room_325 ORDER BY Room_rate DESC;
```

- ❖ SQL Query Explanation:

In the query, the selected table will be display the value with the descending order by the room rate.

The screenshot shows the Oracle SQL Developer interface. The left sidebar displays connections to 'practicedatabase' and 'teamproject'. The main area has tabs for 'Worksheet' and 'Query Builder'. The Worksheet tab contains a query:

```
SELECT Room_Number as Room_Number_325, Room_type, Maximum_occupancy, Room_rate, Bed_type FROM Room_325 ORDER BY Room_rate DESC;
```

The results are displayed in a table titled 'Script Output' under the 'Query Result' tab:

ROOM_NUMBER_325	ROOM_TYPE	MAXIMUM_OCCUPANCY	ROOM_RATE	BED_TYPE
1	105 Quad room	4	150	Large King
2	104 Deluxe Room	6	100	King
3	106 Triple room	3	65	Queen
4	107 Penthouse suites	5	55	King
5	102 Double Room	2	50	Double Bed
6	103 Suite	5	50	Queen
7	101 Single Room	5	50	Single Bed
8	109 Connecting rooms	3	45	King
9	108 Junior Suites	4	35	Double Bed

3

- ❖ Average also known as AVG is function that will return the average value of the column.
  - ❖ SQL Query

```
SELECT AVG(Room_rate) AS Avg_Room_Rate_325 FROM Room_325;
```

- ## ❖ SQL Query Explanation:

In the query the average of the room rate is displayed from the room\_325

- ❖ Min is the function that will return the minimum value of the column.
- ❖ SQL Query

```
SELECT MIN(Room_rate) AS Min_Room_Rate_325 FROM Room_325;
```

- ❖ SQL Query Explanation:

In the query the the room rate minimum value is return from the room\_325

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows several databases, including 'teamproject'. In the center, the Worksheet tab contains the SQL query: 'SELECT MIN(Room\_rate) AS Min\_Room\_Rate\_325 FROM Room\_325;'. Below the worksheet, the Query Result tab displays the output: a single row with 'MIN\_ROOM\_RATE\_325' having a value of 35. The status bar at the bottom indicates 'All Rows Fetched: 1 in 0.004 seconds'.

- ❖ Count is the function is the used to return the count of the value of the column.
- ❖ SQL Query

```
SELECT COUNT(DISTINCT Nationality) AS Nationality_325 FROM Customer_325;
```

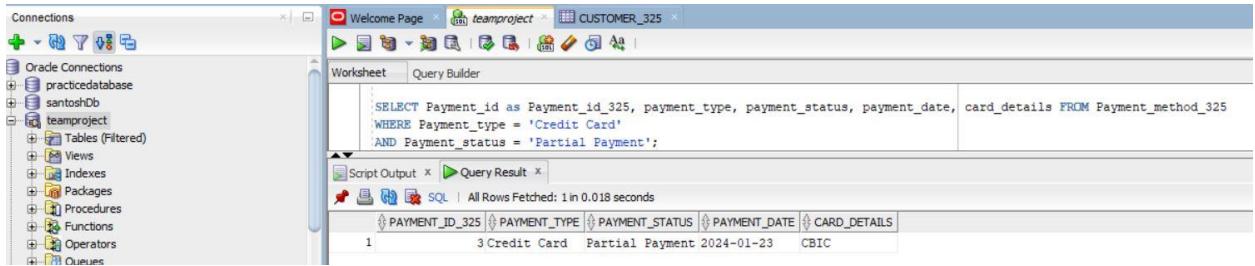
- ❖ SQL Query Explanation:

In the query the distinct nationality that count from the customer\_325

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows the 'teamproject' database. In the center, the Worksheet tab contains the SQL query: 'SELECT COUNT(DISTINCT Nationality) AS Nationality\_325 FROM Customer\_325;'. Below the worksheet, the Query Result tab displays the output: a single row with 'NATIONALITY\_325' having a value of 4. The status bar at the bottom indicates 'All Rows Fetched: 1 in 0.004 seconds'.

- ❖ And operator is used to determine the value that will match all the case before returning the value.
- ❖ SQL Query

```
SELECT Payment_id as Payment_id_325, payment_type, payment_status, payment_date,  
card_details FROM Payment_method_325 WHERE Payment_type = 'Credit Card'  
AND Payment_status = 'Partial Payment';
```



The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree displays several databases, with 'teamproject' selected. The central workspace shows a query in the Worksheet tab:

```
SELECT Payment_id as Payment_id_325, payment_type, payment_status, payment_date,  
card_details FROM Payment_method_325 WHERE Payment_type = 'Credit Card'  
AND Payment_status = 'Partial Payment';
```

Below the query, the Script Output tab shows the results:

PAYMENT_ID_325	PAYMENT_TYPE	PAYMENT_STATUS	PAYMENT_DATE	CARD_DETAILS
1	Credit Card	Partial Payment	2024-01-23	CBIC

The message "All Rows Fetched: 1 in 0.018 seconds" is displayed below the table.

## Category D

1

- ❖ INSTR is the function that return the first occurrence position of a string in another string.
- ❖ SUBSTR is the function that extract substring from a string.
- ❖ SQL Query

```
select Customer_id as Customer_id_325, SUBSTR(Customer_Name, 1, INSTR(Customer_Name, ' ') - 1) AS First_Name, SUBSTR(Customer_Name, INSTR(Customer_Name, ' ') + 1) AS Last_Name, customer_number from Customer_325;
```

The screenshot shows the Oracle SQL Developer interface. On the left is the Object Navigator pane, which lists various database objects under the 'teamproject' schema, including tables, views, indexes, packages, procedures, functions, operators, queues, queue tables, triggers, types, sequences, materialized views, and materialized view logs. The central workspace contains a query window with the following SQL code:

```
select Customer_id as Customer_id_325, SUBSTR(Customer_Name, 1, INSTR(Customer_Name, ' ') - 1) AS First_Name, SUBSTR(Customer_Name, INSTR(Customer_Name, ' ') + 1) AS Last_Name, customer_number from Customer_325;
```

Below the query window is the 'Script Output' tab, which displays the execution results:

CUSTOMER_ID_325	FIRST_NAME	LAST_NAME	CUSTOMER_NUMBER
1	201 Santosh	Acharya	4379551559
2	202 Anish	Thapa	5365962458
3	203 Bhim	Kumari Lamsal	5584166465
4	204 Roshan	Acharya	4372415609
5	205 David	Becham	7589351234

2

- ❖ UPPER is the function that will return the all the string in the column with the upper case.
- ❖ SQL Query

```
SELECT UPPER(Customer_Name) AS Cusotmer_Name_325, Nationality, customer_number FROM Customer_325;
```

The screenshot shows the Oracle SQL Developer interface. The Object Navigator pane on the left is identical to the previous screenshot, listing objects under the 'teamproject' schema. The central workspace contains a query window with the following SQL code:

```
SELECT UPPER(Customer_Name) AS Cusotmer_Name_325, Nationality, customer_number FROM Customer_325;
```

Below the query window is the 'Script Output' tab, which displays the execution results:

CUSOTMER_NAME_325	NATIONALITY	CUSTOMER_NUMBER
1 SANTOSH ACHARYA	Nepali	4379551559
2 ANISH THAPA	Canadian	5365962458
3 BHIM KUMARI LAMSAL	Nepali	5584166465
4 ROSHAN ACHARYA	American	4372415609
5 DAVID BECHAM	British	7589351234

3

- ❖ TO\_DATE is the function that will convert the string to the date format in the SQL.
- ❖ SQL Query

```
SELECT Payment_id, Payment_status, TO_DATE(Payment_date, 'YYYY-MM-DD') AS payment_date_325 FROM Payment_method_325;
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows several databases, with 'teamproject' selected. The central area has a 'Worksheet' tab open with the following SQL code:

```
SELECT Payment_id, Payment_status, TO_DATE(Payment_date, 'YYYY-MM-DD') AS payment_date_325 FROM Payment_method_325;
```

Below the worksheet is a 'Query Result' tab showing the output of the query:

PAYMENT_ID	PAYMENT_STATUS	PAYMENT_DATE_325
1	1 Full	30-NOV-23
2	2 No Payment	31-DEC-23
3	3 Partial Payment	23-JAN-24
4	4 No Payment	15-MAR-24
5	5 Partial Payment	23-JAN-24

4

- ❖ CEIL is the function that will return the nearest small value that is greater than or less than the number.
- ❖ SQL Query

```
SELECT CEIL(Room_rate) AS Room_Rate_325, Room_type, bed_type FROM Room_325;
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows several databases, with 'teamproject' selected. The central area has a 'Worksheet' tab open with the following SQL code:

```
SELECT CEIL(Room_rate) AS Room_Rate_325, Room_type, bed_type FROM Room_325;
```

Below the worksheet is a 'Query Result' tab showing the output of the query:

ROOM_RATE_325	ROOM_TYPE	BED_TYPE
1	50 Single Room	Single Bed
2	50 Double Room	Double Bed
3	50 Suite	Queen
4	100 Deluxe Room	King
5	150 Quad room	Large King
6	65 Triple room	Queen
7	55 Penthouse suites	King
8	35 Junior Suites	Double Bed
9	45 Connecting rooms	King

- ❖ COUNT is the function that is used to count the value in the table.
- ❖ HAVING is used in the case when the WHERE clause can't be used in the query.
- ❖ SQL Query

```
SELECT Nationality as Nationality_325, COUNT(*) AS Total_Customers_325 FROM Customer_325
GROUP BY Nationality HAVING COUNT(*) > 0;
```

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, showing Oracle Connections, practicedatabase, santoshDb, and teamproject. The teamproject connection is expanded, showing Tables (Filtered), Views, Indexes, Packages, Procedures, Functions, Operators, Queues, Queues Tables, Triggers, and Types. The main workspace shows a Worksheet tab with the query:

```
SELECT Nationality as Nationality_325, COUNT(*) AS Total_Customers_325 FROM Customer_325
GROUP BY Nationality HAVING COUNT(*) > 0;
```

Below the query is the Script Output tab, which shows the results:

NATIONALITY_325	TOTAL_CUSTOMERS_325
1 Nepali	2
2 Canadian	1
3 American	1
4 British	1

- ❖ Round is the function that is used to round off the value to a specified number of decimal places.
- ❖ SQL Query

```
SELECT ROUND(inventory_rate, 2)as Inventory_rate_0930325 FROM inventory_325;
```

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, showing Oracle Connections, practicedatabase, santoshDb, and teamproject. The teamproject connection is expanded, showing Tables (Filtered), Views, Indexes, Packages, Procedures, Functions, Operators, Queues, Queues Tables, Triggers, Types, Sequences, Materialized Views, and Materialized View Logs. The main workspace shows a Worksheet tab with the query:

```
SELECT ROUND(inventory_rate, 2)as Inventory_rate_0930325 FROM inventory_325;
```

Below the query is the Script Output tab, which shows the results:

INVENTORY_RATE_0930325	
1	12.99
2	24.99
3	6.99
4	39.99
5	29.99
6	19.99
7	49.99

- ❖ COALESCE is used to return the first non-value in the list.
- ❖ It used two argument within it.
- ❖ SQL Query

```
SELECT inventory_id as inventory_id_0930325, inventory_rate, COALESCE(inventory_description, 'No description available') FROM inventory_325;
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows several databases, including 'teamproject'. The central area has a 'Worksheet' tab open with the following SQL query:

```
SELECT inventory_id as inventory_id_0930325, inventory_rate, COALESCE(inventory_description, 'No description available') FROM inventory_325;
```

Below the worksheet, the 'Script Output' tab shows the results of the query. The output table has three columns: INVENTORY\_ID\_0930325, INVENTORY\_RATE, and COALESCE(INVENTORY\_DESCRIPTION,'NODESCRIPTIONAVAILABLE'). The data is as follows:

INVENTORY_ID_0930325	INVENTORY_RATE	COALESCE(INVENTORY_DESCRIPTION,'NODESCRIPTIONAVAILABLE')
1	1	12.99 White cotton bath towels for guest rooms
2	2	24.99 Luxurious cotton bed linens for all bed sizes
3	3	6.99 Travel-sized shampoo, conditioner, and soap
4	4	39.99 Drip coffee maker with complimentary coffee pods
5	5	29.99 Steam iron and ironing board for guest use
6	6	19.99 Professional-grade hair dryer with multiple heat settings
7	7	49.99 Electronic safe for securing valuables

The 'Script Output' tab also displays a message: 'All Rows Fetched: 7 in 0.007 seconds'.

## Category E

1

- ❖ Inner join is the function that is used to join the table when both the table has some matching values in a field that is common to both the table.
- ❖ SQL Query

```
SELECT c.Customer_id as Customer_id_325 , c.Customer_Name, p.Payment_type  
FROM Customer_325 c  
INNER JOIN Payment_method_325 p ON c.Payment_id = p.Payment_id;
```

- ❖ SQL Explanation:

In the query, the Customer\_325 is join with the Payment\_method\_325 where customer.paymentid is equal to payment.paymentid. c denote as the alias for the table customer and p for the payment.

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows an Oracle connection named 'practicedatabase' and a schema named 'teamproject'. The 'Tables (Filtered)' node under 'teamproject' is expanded, showing various table types like Tables, Views, Indexes, Packages, Procedures, Functions, Operators, Queues, Queues Tables, Triggers, Types, Sequences, Materialized Views, and Materialized View Logs. In the center, the Worksheet tab contains the SQL query:

```
SELECT c.Customer_id as Customer_id_325 , c.Customer_Name, p.Payment_type  
FROM Customer_325 c  
INNER JOIN Payment_method_325 p ON c.Payment_id = p.Payment_id;
```

Below the worksheet, the Script Output tab shows the results of the query execution:

All Rows Fetched: 5 in 0.004 seconds

CUSTOMER_ID_325	CUSTOMER_NAME	PAYMENT_TYPE
1	201 Santosh Acharya	Interact
2	202 Anish Thapa	Debit-Card
3	203 Bhim Kumari Lamsal	Credit Card
4	204 Roshan Acharya	Google pay
5	205 David Becham	Bank Transfer

## 2

- ❖ Left outer join will return all the record from the left table and the matching record from the right table in the sql.
- ❖ SQL Query

```
SELECT c.Customer_id as Customer_id_325, c.Customer_Name,c.Customer_Number,
p.Payment_type FROM Customer_325 c
LEFT OUTER JOIN Payment_method_325 p ON c.Payment_id = p.Payment_id;
```

- ❖ SQL Query Explanation:

In the query , the table will return all the record from the customer and matching record from the payment method.

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' sidebar with 'teamproject' selected. The main area has a 'Worksheet' tab open with the following SQL code:

```
SELECT c.Customer_id as Customer_id_325, c.Customer_Name,c.Customer_Number,
p.Payment_type FROM Customer_325 c
LEFT OUTER JOIN Payment_method_325 p ON c.Payment_id = p.Payment_id;
```

Below the worksheet is a 'Query Result' tab showing the output:

	CUSTOMER_ID_325	CUSTOMER_NAME	CUSTOMER_NUMBER	PAYMENT_TYPE
1	201	Santosh Acharya	4379551559	Interact
2	202	Anish Thapa	5365962458	Debit-Card
3	203	Bhim Kumari Lamsal	5584166465	Credit Card
4	204	Roshan Acharya	4372415609	Google pay
5	205	David Becham	7589351234	Bank Transfer

## 3

- ❖ Right outer join is similar to the left outer join but in Right outer join all the record from the right table is recorded and all the matching record from the left table is extracted.
- ❖ SQL Query

```
SELECT Supplier_id as Supplier_id_0930325, product_type, supplier_address,
supplier_contact,Inventory_id , Inventory_rate,Inventory_quantity, purchase_date
FROM Supplier_325
RIGHT OUTER JOIN inventory_325 ON Supplier_325.Supplier_id = inventory_325.inventory_id;
```

```

SELECT Supplier_id as Supplier_id_0930325, product_type, supplier_address, supplier_contact,Inventory_id , Inventory_rate,Inventory_quantity, purchase_date
FROM Supplier_325
RIGHT OUTER JOIN inventory_325 ON Supplier_325.Supplier_id = inventory_325.Inventory_id;

```

Script Output | Query Result | All Rows Fetched: 7 in 0.006 seconds

SUPPLIER_ID_0930325	PRODUCT_TYPE	SUPPLIER_ADDRESS	SUPPLIER_CONTACT	INVENTORY_ID	INVENTORY_RATE	INVENTORY_QUANTITY	PURCHASE_DATE
1	1 Electronics	123 Main St, City A	1234567	1	12.99	50	2024-04-07
2	2 Clothing	456 Elm St, City B	9876543	2	24.99	100	2024-04-07
3	3 Books	789 Oak St, City C	5554321	3	6.99	200	2024-04-07
4	4 Furniture	321 Pine St, City D	1112222	4	39.99	15	2024-04-07
5	5 Groceries	555 Maple St, City E	3334444	5	29.99	20	2024-04-07
6	6 Electronics	777 Cedar St, City F	8889999	6	19.99	30	2024-04-07
7	7 Clothing	999 Walnut St, City G	7771111	7	49.99	10	2024-04-07

4

- ❖ Cross join is used to combine the each row of one table to each row of another table.
- ❖ It return the Cartesian product of the sets of rows.
- ❖ SQL Query

```

SELECT c.customer_id as Customer_id_0930325, c.customer_name, c.customer_number,
c.nationality, p.payment_id,p.payment_status FROM Customer_325 c
CROSS JOIN Payment_method_325 p;

```

```

SELECT c.customer_id as Customer_id_0930325, c.customer_name, c.customer_number, c.nationality, p.payment_id,p.payment_status
FROM Customer_325 c
CROSS JOIN Payment_method_325 p;

```

Script Output | Query Result | All Rows Fetched: 25 in 0.007 seconds

CUSTOMER_ID_0930325	CUSTOMER_NAME	CUSTOMER_NUMBER	NATIONALITY	PAYMENT_ID	PAYMENT_STATUS
1	201 Santosh Acharya	4379551559	Nepali	1	Full
2	201 Santosh Acharya	4379551559	Nepali	2	No Payment
3	201 Santosh Acharya	4379551559	Nepali	3	Partial Payment
4	201 Santosh Acharya	4379551559	Nepali	4	No Payment
5	201 Santosh Acharya	4379551559	Nepali	5	Partial Payment
6	202 Anish Thapa	5365962450	Canadian	1	Full
7	202 Anish Thapa	5365962450	Canadian	2	No Payment
8	202 Anish Thapa	5365962450	Canadian	3	Partial Payment
9	202 Anish Thapa	5365962450	Canadian	4	No Payment
10	202 Anish Thapa	5365962450	Canadian	5	Partial Payment
11	203 Bhim Kumari Lameal	5584166465	Nepali	1	Full
12	203 Bhim Kumari Lameal	5584166465	Nepali	2	No Payment
13	203 Bhim Kumari Lameal	5584166465	Nepali	3	Partial Payment
14	203 Bhim Kumari Lameal	5584166465	Nepali	4	No Payment
15	203 Bhim Kumari Lameal	5584166465	Nepali	5	Partial Payment
16	204 Roshan Acharya	4372415609	American	1	Full
17	204 Roshan Acharya	4372415609	American	2	No Payment
18	204 Roshan Acharya	4372415609	American	3	Partial Payment
19	204 Roshan Acharya	4372415609	American	4	No Payment
20	204 Roshan Acharya	4372415609	American	5	Partial Payment
21	205 David Becham	7589351234	British	1	Full
22	205 David Becham	7589351234	British	2	No Payment
23	205 David Becham	7589351234	British	3	Partial Payment
24	205 David Becham	7589351234	British	4	No Payment
25	205 David Becham	7589351234	British	5	Partial Payment

## 5

- ❖ Self join is the query in which table is combined with itself in the sql.
- ❖ Here all the value is combined with itself in the query.
- ❖ SQL Query

```
SELECT t1.inventory_id as inventory_id_0930325,
t1.inventory_name,t1.inventory_rate,t1.inventory_quantity,t2.inventory_id,
t2.inventory_rate,t2.inventory_quantity FROM inventory_325 t1, inventory_325 t2
WHERE t1.inventory_id = t2.inventory_id;
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections sidebar lists 'practicedatabase' and 'teamproject'. The 'teamproject' connection is selected, showing tables like 'Inventory\_325', 'Views', 'Indexes', etc. The central area has a 'Worksheet' tab with the SQL query:

```
SELECT t1.inventory_id as inventory_id_0930325,
t1.inventory_name,t1.inventory_rate,t1.inventory_quantity,t2.inventory_id,
t2.inventory_rate,t2.inventory_quantity FROM inventory_325 t1, inventory_325 t2
WHERE t1.inventory_id = t2.inventory_id;
```

Below the worksheet is a 'Script Output' tab showing the execution details: 'All Rows Fetched: 7 in 0.005 seconds'. The 'Query Result' tab displays the query results in a grid:

	INVENTORY_ID_0930325	INVENTORY_NAME	INVENTORY_RATE	INVENTORY_QUANTITY	INVENTORY_ID	INVENTORY_RATE_1	INVENTORY_QUANTITY_1
1	1	Bath Towels	12.99	50	1	12.99	50
2	2	Bed Linens	24.99	100	2	24.99	100
3	3	Toiletries Kit	6.99	200	3	6.99	200
4	4	Coffee Maker	39.99	15	4	39.99	15
5	5	Iron and Ironing Board	29.99	20	5	29.99	20
6	6	Hair Dryer	19.99	30	6	19.99	30
7	7	Room Safe	49.99	10	7	49.99	10

## Category F

1

- ❖ Nested query is the query that has query inside the other query.
- ❖ Exists is the query in the sql which will return the value in the given condition exists in the query
- ❖ SQL Query

```
SELECT *
FROM Customer_325 c
WHERE EXISTS (SELECT * FROM Payment_method_325 p WHERE p.Payment_id = c.Payment_id);
```

- ❖ SQL Explanation:

In the query the customer table will return the value if the nested query full fill the requirement where paymentid of payment method table is equal to the paymentid of customer table.

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Connections' sidebar lists 'practicedatabase', 'santoshDb', and 'teamproject'. The 'teamproject' connection is selected, showing tables like 'Tables (Filtered)', 'Views', 'Indexes', 'Procedures', 'Functions', 'Operators', 'Queues', 'Triggers', 'Types', 'Sequences', 'Materialized Views', and 'Materialized View Logs'. The central workspace has a 'Worksheet' tab with the following SQL code:

```
SELECT *
FROM Customer_325 c
WHERE EXISTS (SELECT * FROM Payment_method_325 p WHERE p.Payment_id = c.Payment_id);
```

Below the worksheet is the 'Query Result' tab, which displays the results of the executed query. The results are as follows:

CUSTOMER_ID	CUSTOMER_NAME	CUSTOMER_NUMBER	EMAIL_ADDRESS	BOOKING_HISTORY	NATIONALITY	MEMBERSHIP_STATUS	PAYMENT_ID
1	201 Santosh Acharya	4379551559	acharyaanish1@gmail.com	(null)	Nepali	Full Member	1
2	202 Anish Thapa	5365962458	aanishthapa@gmail.com	2	Canadian	Expired	2
3	203 Bhim Kumari Lamsal	5584166465	bhimkumari.lamsal5@gmail.com	(null)	Nepali	(null)	3
4	204 Roshan Acharya	4372415609	roshanacharya050@gmail.com	5	American	Full Member	4
5	205 David Becham	7589351234	davidbecham5@gmail.com	(null)	British	Full Member	5

2

- ❖ UNION is use to combine the result set of two or more select statement.
- ❖ SQL Query

```
SELECT Customer_id as Customer_id_325, Customer_Name FROM Customer_325
UNION
SELECT Payment_id, Payment_type FROM Payment_method_325;
```

- ❖ SQL Explanation:

In the query the customer\_325 table is combine with the payment\_method\_325 table.

```

SELECT Customer_id as Customer_id_325, Customer_Name FROM Customer_325
UNION
SELECT Payment_id, Payment_type FROM Payment_method_325;

```

CUSTOMER_ID_325	CUSTOMER_NAME
1	201 Santosh Acharya
2	202 Anish Thapa
3	203 Bhim Kumari Lamsal
4	204 Roshan Acharya
5	205 David Becham
6	1 Interact
7	2 Debit-Card
8	3 Credit Card
9	4 Google pay
10	5 Bank Transfer

3

- ❖ MINUS is the function used to return all the rows in the first select statement that are not returned by the second.
- ❖ It will return the unique rows.
- ❖ SQL Query

```

SELECT Customer_id as Customer_id_325, Customer_Name FROM Customer_325
MINUS
SELECT Payment_id, Payment_type FROM Payment_method_325;

```

```

SELECT Customer_id as Customer_id_325, Customer_Name FROM Customer_325
MINUS
SELECT Payment_id, Payment_type FROM Payment_method_325;

```

CUSTOMER_ID_325	CUSTOMER_NAME
1	201 Santosh Acharya
2	202 Anish Thapa
3	203 Bhim Kumari Lamsal
4	204 Roshan Acharya
5	205 David Becham

- ❖ NOT EXISTS is the query that test the non existing value in the table.
- ❖ It is used when to ignore the logic of exists.
- ❖ SQL Query

```
SELECT inventory_id as inventory_id_0930325, inventory_name,
inventory_rate,inventory_quantity,Purchase_date FROM inventory_325 i
WHERE NOT EXISTS (
SELECT * FROM Supplier_325 s
WHERE s.Product_type = 'Bath Towels'
AND s.Supplier_id = i.inventory_id
);
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows an Oracle connection named 'practicedatabase' and a project named 'teamproject'. The 'Tables (Filtered)' node under 'teamproject' is expanded, showing various table types. The central workspace contains a 'Worksheet' tab with the following SQL code:

```
SELECT inventory_id as inventory_id_0930325, inventory_name, inventory_rate,inventory_quantity,Purchase_date
FROM inventory_325 i
WHERE NOT EXISTS (
SELECT *
FROM Supplier_325 s
WHERE s.Product_type = 'Books'
AND s.Supplier_id = i.inventory_id
);
```

Below the worksheet is a 'Script Output' tab showing the execution details: 'All Rows Fetched: 6 in 0.002 seconds'. The 'Query Result' tab displays the fetched data in a grid:

	INVENTORY_ID_0930325	INVENTORY_NAME	INVENTORY_RATE	INVENTORY_QUANTITY	PURCHASE_DATE
1	6	Hair Dryer	19.99	30	2024-04-07
2	1	Bath Towels	12.99	50	2024-04-07
3	7	Room Safe	49.99	10	2024-04-07
4	2	Bed Linens	24.99	100	2024-04-07
5	4	Coffee Maker	39.99	15	2024-04-07
6	5	Iron and Ironing Board	29.99	20	2024-04-07

## 5

- ❖ Not null is the function that will return the value that aren't null or have some value in the column or table.
- ❖ SQL Query

```
SELECT inventory_id as inventory_id_0930325, inventory_name, inventory_quantity,purchase_date
FROM inventory_325
WHERE inventory_quantity > (
    SELECT avg(inventory_rate) FROM inventory_325 WHERE inventory_name is not null
);
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree displays 'teamproject' as the current connection. The central area is a 'Worksheet' tab where the previously provided SQL query is pasted. Below the worksheet, the 'Script Output' tab shows the results of the query execution. A table titled 'INVENTORY\_ID\_0930325' is displayed with the following data:

	INVENTORY_NAME	INVENTORY_QUANTITY	PURCHASE_DATE
1	1 Bath Towels	50	2024-04-07
2	2 Bed Linens	100	2024-04-07
3	3 Toiletries Kit	200	2024-04-07
4	6 Hair Dryer	30	2024-04-07

## Advance SQL

1

- ❖ In will allow to accept multiple value . It operate as the OR operator in the sql
- ❖ SQL Query

```
SELECT inventory_id as inventory_id_0930325, inventory_name, inventory_rate
FROM inventory_325 WHERE inventory_id IN (
    SELECT inventory_id FROM Supplier_325 WHERE Product_type = 'Furniture'
);
```

- ❖ SQL Query Explanation:

In the query first the inventory id is selected from the suppliers table which match the furniture and then all the number that match in the inventory table is displayed.

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections sidebar lists 'practicedatabase' and 'santoshDb'. The 'teamproject' connection is selected, showing its schema objects like Tables, Views, Indexes, etc. Below it, the Reports sidebar shows 'All Reports', 'Analytic View Reports', 'Data Dictionary Reports', and 'Data Modeler Reports'. The central workspace has a 'Worksheet' tab open with the following SQL query:

```
SELECT inventory_id as inventory_id_0930325, inventory_name, inventory_rate
FROM inventory_325 WHERE inventory_id IN (
    SELECT inventory_id
    FROM Supplier_325
    WHERE Product_type = 'Furniture'
);
```

Below the worksheet, the 'Script Output' tab shows the results of the query:

INVENTORY_ID_0930325	INVENTORY_NAME	INVENTORY_RATE
1	1 Bath Towels	12.99
2	2 Bed Linens	24.99
3	3 Toiletries Kit	6.99
4	4 Coffee Maker	39.99
5	5 Iron and Ironing Board	29.99
6	6 Hair Dryer	19.99
7	7 Room Safe	49.99

## 2

- ❖ JOIN is used to combine the two or more table related to there column.
- ❖ SQL Query

```
SELECT c.Customer_Name as Customer_Name_0930325, p.Payment_type
FROM Customer_325 c
JOIN Payment_method_325 p ON c.Payment_id = p.Payment_id
WHERE c.Customer_id IN ( SELECT Customer_id FROM Room_325 WHERE Room_type = 'Suite'
);
```

- ❖ SQL Explanation:

In the Query the customerid is selected from the room table and according to the value the customer table and payment method table is combined.

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree shows a connection to 'teamproject'. The central area is the Worksheet window, which contains the SQL query:

```
SELECT c.Customer_Name as Customer_Name_0930325, p.Payment_type
FROM Customer_325 c
JOIN Payment_method_325 p ON c.Payment_id = p.Payment_id
WHERE c.Customer_id IN (
    SELECT Customer_id
    FROM Room_325
    WHERE Room_type = 'Suite'
);
```

Below the query, the Script Output tab shows the results:

CUSTOMER_NAME_0930325	PAYMENT_TYPE
1 Santosh Acharya	Interact
2 Anish Thapa	Debit-Card
3 Bhim Kumari Lamsal	Credit Card
4 Roshan Acharya	Google pay
5 David Becham	Bank Transfer

- ❖ Average function will return the average value of the column.
- ❖ SQL Query

```
SELECT inventory_name as inventory_name_0930325, inventory_quantity,
(SELECT AVG(inventory_quantity) FROM inventory_325 i2
WHERE i1.inventory_name = i2.inventory_name) AS avg_quantity
FROM inventory_325 i1;
```

❖ SQL Query Explanation:

In the query the average of inventory quantity is taken when the inventory\_name of the table is equal to the inventory name of the other table and the output is displayed.

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections sidebar lists 'practicedatabase' and 'teamproject'. The teamproject connection is expanded, showing 'Tables (Filtered)', 'Views', 'Indexes', 'Packages', 'Procedures', 'Functions', 'Operators', 'Queues', 'Queues Tables', 'Triggers', 'Types', 'Sequences', 'Materialized Views', 'Materialized View Logs', and 'Synonyms'. Below this is the Reports sidebar with 'All Reports' and 'Analytic View Reports'. The central workspace has tabs for 'Worksheet' and 'Query Builder'. The Worksheet tab contains the SQL query:

```
SELECT inventory_name as inventory_name_0930325,
       inventory_quantity,
       (SELECT AVG(inventory_quantity)
        FROM inventory_325 i2
       WHERE i1.inventory_name = i2.inventory_name) AS avg_quantity
    FROM inventory_325 i1;
```

Below the query is the 'Script Output' tab, which displays the results of the query. The results are presented in a table:

INVENTORY_NAME_0930325	INVENTORY_QUANTITY	AVG_QUANTITY
1 Bath Towels	50	50
2 Bed Linens	100	100
3 Toiletries Kit	200	200
4 Coffee Maker	15	15
5 Iron and Ironing Board	20	20
6 Hair Dryer	30	30
7 Room Safe	10	10

The status bar at the bottom of the interface indicates 'All Rows Fetched: 7 in 0.029 seconds'.

- ❖ MAX function will return the maximum value of the column.
- ❖ It is used to get the highest value with in the column.
- ❖ SQL Query

```
SELECT inventory_name as inventory_name_0930325, inventory_quantity,
(SELECT MAX(Room_rate) FROM Room_325) AS max_room_rate
FROM inventory_325;
```

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections tree displays several databases, with 'teamproject' selected. The central area is a Worksheet tab containing the SQL query:

```
SELECT inventory_name as inventory_name_0930325,
       inventory_quantity,
       (SELECT MAX(Room_rate)
        FROM Room_325) AS max_room_rate
  FROM inventory_325;
```

Below the worksheet, the Script Output tab shows the results of the query execution:

INVENTORY_NAME_0930325	INVENTORY_QUANTITY	MAX_ROOM_RATE
1 Bath Towels	50	150
2 Bed Linens	100	150
3 Toiletries Kit	200	150
4 Coffee Maker	15	150
5 Iron and Ironing Board	20	150
6 Hair Dryer	30	150
7 Room Safe	10	150

The message "All Rows Fetched: 7 in 0.006 seconds" is displayed at the bottom of the worksheet tab.

## 6

- ❖ Exists function will return the value when the condition is meet.
- ❖ It works only if there is one value that will be return.
- ❖ SQL Query

```
SELECT Customer_id as Customer_id_0930325, customer_name,  
customer_number,membership_status,payment_id  
FROM Customer_325 c WHERE EXISTS (  
SELECT * FROM Payment_method_325 p WHERE c.Payment_id = p.Payment_id  
AND p.Payment_status = 'Full'  
);
```

