

Creating and Managing Tables

EX_NO:1

DATE:

1. Create the DEPT table based on the DEPARTMENT following the table instance chart below. Confirm that the table is created.

Column name	ID	NAME
Key Type		
Nulls/Unique		
FK table		
FK column		
Data Type	Number	Varchar2
Length	7	25

QUERY:

```
Create table depcse(id int,name varchar(20));
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'aarthis' (aarthi001), and a schema dropdown set to 'WKSP_AARTHIO01'. The main area is titled 'SQL Commands' with a language dropdown set to 'SQL'. Below it, there are buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. The command history shows two lines of code: 'create table depcse(id int,name varchar(20))' and '2'. At the bottom, tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History' are visible, along with a message 'Table created.' and a performance metric '0.04 seconds'.

2. Create the EMP table based on the following instance chart. Confirm that the table is created.

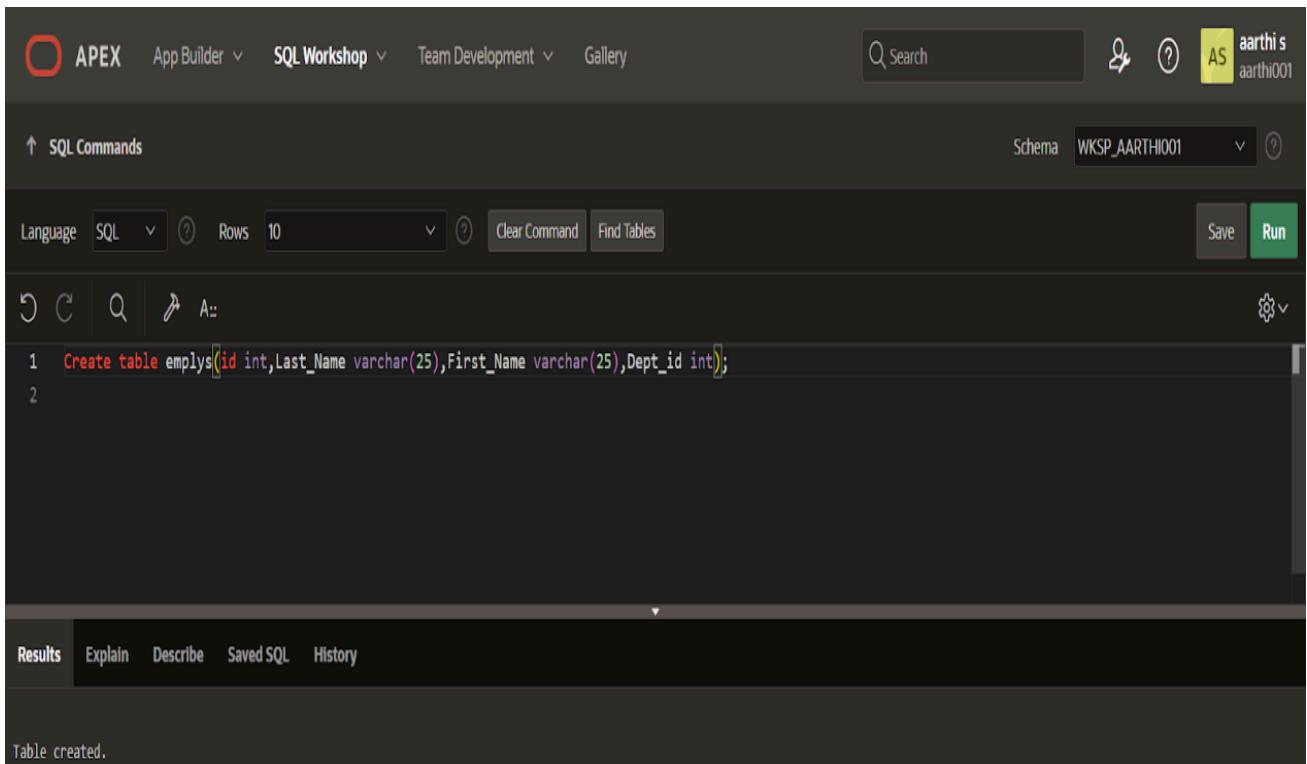
Column name	ID	LAST_NAME	FIRST_NAME	DEPT_ID
Key Type				
Nulls/Unique				
FK table				
FK column				

Data Type	Number	Varchar2	Varchar2	Number
Length	7	25	25	7

QUERY:

```
Create table empoly(id int,Last_Name varchar(25),First_Name varchar(25),Dept_id int);
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right side, there's a user profile for 'aarthis' (ID: aarthi001). The main workspace is titled 'SQL Commands'. It features a toolbar with icons for Undo, Redo, Search, and Find Tables. Below the toolbar, there are dropdown menus for Language (set to SQL) and Rows (set to 10), along with buttons for Clear Command and Run. The SQL command area contains the following code:

```
1 Create table empoly(id int,Last_Name varchar(25),First_Name varchar(25),Dept_id int);
2
```

At the bottom of the workspace, there are tabs for Results, Explain, Describe, Saved SQL, and History. The 'Results' tab is active, displaying the message "Table created.".

3. Modify the EMP table to allow for longer employee last names. Confirm the modification.(Hint: Increase the size to 50)

QUERY:

```
Alter table emp modify(Last_Name varchar(25))
```

OUTPUT:

The screenshot shows a SQL command window with the following details:

- Header: SQL Commands, Schema: WKSP_AARTHI001.
- Toolbar: Language (SQL), Rows (10), Clear Command, Find Tables, Save, Run.
- Text Area:

```
1 Alter table emp modify(Last_Name varchar(25))
2
```
- Results Tab: Shows the output of the command: "Table altered." and "0.06 seconds".

4. Create the EMPLOYEES2 table based on the structure of EMPLOYEES table. Include Only the Employee_id, First_name, Last_name, Salary and Dept_id coloumns. Name the columns Id, First_name, Last_name, salary and Dept_id respectively.

QUERY:

```
Create table employees3 (id int,first_name varchar(25),Last_name varchar2(25),Salary int,Dept_id int);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there are navigation links: APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, a user icon labeled 'aarthi s aarthi001', and a schema dropdown set to 'WKSP_AARTHI001'. Below the header, the title 'SQL Commands' is displayed. The main area contains a code editor with the following SQL command:

```
1 Create table employees5 (id int,first_name varchar(25),Last_name varchar2(25),Salary int,Dept_id int);  
2  
3
```

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is selected. The output section shows the message 'Table created.' and a execution time of '0.04 seconds'.

5. Drop the EMP table.

QUERY:

```
Drop table employees3;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The title 'SQL Commands' is at the top. The main area contains the following SQL command:

```
1 Drop table employees3  
2
```

The Results tab is selected. The output section shows the message 'Table dropped.'

6. Rename the EMPLOYEES2 table as EMP.

QUERY:

```
Alter table employ2 rename to em2;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links for App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, user information for 'aarthi s' (schema 'WKSP_AARTHIO01'), and a 'Run' button. The main area is titled 'SQL Commands'. It shows the following command:

```
1 Alter table emp2 rename to employees;
```

Below the command, the results show:

Table altered.
0.15 seconds

At the bottom, it displays the user's email (220701001@rajalakshmi.edu.in), session ID (aarthi001), and language (en). The copyright notice is 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version is 'Oracle APEX 23.2.4'.

7. Add a comment on DEPT and EMP tables. Confirm the modification by describing the table.

QUERY:

Comment on table dept is 'Department info';
Comment on table emp is Employee info';

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links for App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, user information for 'aarthi s' (schema 'WKSP_AARTHIO01'), and a 'Run' button. The main area is titled 'SQL Commands'. It shows the following commands:

```
1 Comment on table dept is 'Department info';
2
3
```

Below the commands, the results show:

statement processed.
0.04 seconds

8. Drop the First_name column from the EMP table and confirm it.

QUERY:

Alter table my_emp drop column first_name

Output:

The screenshot shows the Oracle SQL Workshop interface. At the top, there are navigation links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right side, there is a search bar, a user icon labeled 'aarthi s aarthi001', and a connection status indicator. Below the header, the title bar says 'SQL Commands'. The main area contains a code editor with the following SQL command:

```
1 Alter table my_emp drop column first_name
```

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is selected. The output section displays the message "Table altered." and "0.08 seconds".

Evaluation Procedure	Marks awarded
Query (5)	
Execution (5)	
Viva (5)	
Total (5)	
Faculty Signature	

MANIPULATING DATA

EX_NO:2

DATE:

1. Create MY_EMPLOYEE table with the following structure.

NAME	null?	Type
ID	Not null	number(4)
Last_name		varchar(25)
First_name		varchar(25)
Userid		varchar(25)
Salary		number(9,2)

QUERY:

```
Create table my_employee2(id int,last_name varchar(25),first_name varchar(25),userid varchar(20),salary number(9,2));
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands tab, the following SQL code is entered:

```
1 create table my_employee2(id int,last_name varchar(25),first_name varchar(25),userid varchar(20),salary number(9,2));
2 
```

In the Results tab, the output shows:

```
Table created.
0.05 seconds
```

2. Add the first and second rows data to MY_EMPLOYEE table from the following sample data.

ID	Last_name	First_name	Userid	salary
1	Patel	Ralph	Rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	Cnewman	750
5	Ropebur	Audrey	aropebur	1550

QUERY:

```
Insert into my_employee2 values(1,'patel','ralph','rptel',895),
(2,'dancs','betty','bdancs',860);
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands pane, the following SQL code is entered:

```
1 Insert into my_employee2 (ID, LAST_NAME, FIRST_NAME,
2 | USERID,Salary)values(1,'patel1','ralph1','rptell1',895);
3
```

In the Results pane, the output is:

```
1 row(s) inserted.  
0.03 seconds
```

At the bottom, it shows the user information: 220701001@rajalakshmi.edu.in, aarthi001, en, and the copyright notice: Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4.

3. Populate the next two rows of data from the sample data. Concatenate the first letter of the first_name with the first seven characters of the last_name to produce Userid.

QUERY:

```
INSERT INTO MY_EMPLOYEE2  
(ID,LAST_NAME,USERID,SALARY,COMMISSION,FIRST_NAME) VALUES (002,'raman',  
SUBSTR('raman', 1, 1) || SUBSTR('laxmanan', 1, 7),800,12, 'laxmanan');
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands pane, the following SQL code is entered:

```
1 select * from MY_EMPLOYEE2  
2 where userid is not null;  
3  
4  
5
```

In the Results pane, the output is a table:

ID	LAST_NAME	SALARY	COMMISSION	FIRST_NAME	USERID
1	John	800	12	doe	JDoe
2	raman	800	12	laxmanan	rlaxmana

At the bottom, it shows the message: 2 rows returned in 0.01 seconds and a Download button.

4. Display the table with values.

QUERY:

```
select *from my_employee2;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The schema is set to WKSP_AARTHI001. A query is run: `select * from my_employee2`. The results are displayed in a table:

ID	LAST_NAME	SALARY	COMMISSION	FIRST_NAME	USERID
3	dancs	860	17	betty	-
1	John	800	12	doe	JDoe
2	patel	1000	12	ralph	-
4	Biri	1100	15	Ben	-

5. Make the data additions permanent.

QUERY:

```
commit;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. A query is run: `commit;`. The message "Commit statement not applicable. All statements are automatically committed." is displayed.

6. Change the last name of employee 3 to Drexler.

QUERY:

```
Update my_employee2 set last_name ='drexler' where id=3;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. A query is run: `Update my_employee2 set last_name ='drexler' where id=3;`. The message "0 row(s) updated." is displayed.

7. Change the salary to 1000 for all the employees with a salary less than 900.

QUERY:

Update my_emp set salary=1000 where salary<900;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top right, it says "Schema WKSP_AARTHIO01". The main area is titled "SQL Commands" and contains the following SQL code:

```
1 Update my_emp set salary=1000 where salary<900;
2
```

Below the code, the "Results" tab is selected, showing the output: "0 row(s) updated."

8. Delete Betty dancs from MY_EMPLOYEE table.

QUERY:

Delete from my emp where first name='chad';

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top right, it says "Schema WKSP_AARTHIO01". The main area is titled "SQL Commands" and contains the following SQL code:

```
1 DELETE FROM MY_EMP
2 WHERE last_name = 'Dancs'
```

Below the code, the "Results" tab is selected, showing the output: "0 row(s) deleted."

Evaluation Procedure	Marks awarded
Query (5)	

INCLUDING CONSTRAINTS

EX_NO:3

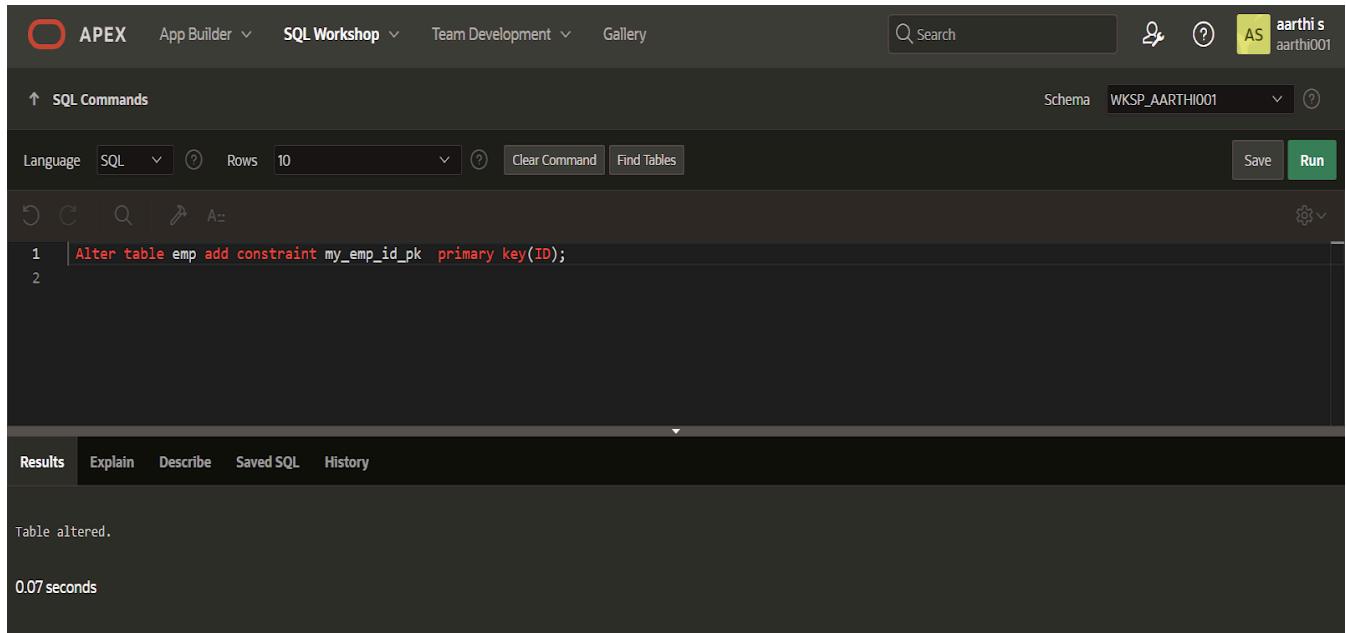
DATE:

1. Add a table-level PRIMARY KEY constraint to the EMP table on the ID column. The constraint should be named at creation. Name the constraint my_emp_id_pk.

QUERY:

```
Alter table emp add constraint my_emp_id_pk primary key(ID);
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and user profile 'aarthi s aarthi001' are also present. The main workspace is titled 'SQL Commands' and shows the following command being run:

```
1 | Alter table emp add constraint my_emp_id_pk primary key(ID);
2 |
```

The results pane at the bottom displays the output: 'Table altered.' and '0.07 seconds'.

2. Create a PRIMARY KEY constraint to the DEPT table using the ID column. The constraint should be named at creation. Name the constraint my_dept_id_pk.

QUERY:

```
Alter table dept add constraint my_dept_id_pk primary key(ID);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. On the right side, a user profile for 'aarthis aarthi001' is shown. The main area is titled 'SQL Commands'. The 'Language' dropdown is set to 'SQL'. The 'Rows' dropdown is set to '10'. There are buttons for 'Clear Command' and 'Find Tables'. On the far right, there are 'Save' and 'Run' buttons. The code editor contains the following SQL command:

```
1 Alter table dept add constraint my_dept_id_pk primary key(id);  
2 |  
3
```

Below the code editor, the results tab is selected. The output shows:

```
Table altered.  
0.08 seconds
```

3. Add a column DEPT_ID to the EMP table. Add a foreign key reference on the EMP table that ensures that the employee is not assigned to non-existent department. Name the constraint my_emp_dept_id_fk.

QUERY:

```
ALTER TABLE my_emp ADD CONSTRAINT my_emp_dept_id_fk FOREIGN KEY (id) REFERENCES dept (id);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar and user profile are identical to the previous screenshot. The 'SQL Commands' section contains the following SQL command:

```
1 ALTER TABLE my_emp ADD CONSTRAINT my_emp_dept_id_fk FOREIGN KEY (id) REFERENCES dept (id);
```

The results tab shows the output:

```
Table altered.  
0.06 seconds
```

4. Modify the EMP table. Add a COMMISSION column of NUMBER data type, precision 2, scale 2. Add a constraint to the commission column that ensures that a commission value is greater than zero.

Query:

```
Alter table emp3 add commission ck check(commision>0);
```

OUTPUT:

The screenshot shows a MySQL Workbench interface with the following details:

- Toolbar:** Includes icons for Undo, Redo, Search, and Save.
- Search Bar:** Displays "in:sent".
- SQL Editor:** Contains the SQL command: `alter table emp3 add[constraints ck check(comission>0)]`.
- Status Bar:** Shows "Active" and other system icons.
- Results Tab:** Active tab, showing the output of the executed query.
- Output:**
 - Text: "Table altered."
 - Text: "0.07 seconds"

Evaluation Procedure	Marks awarded
Query (5)	
Execution (5)	
Viva (5)	
Total (5)	
Faculty Signature	

INCLUDING CONSTRAINTS

EX_NO:

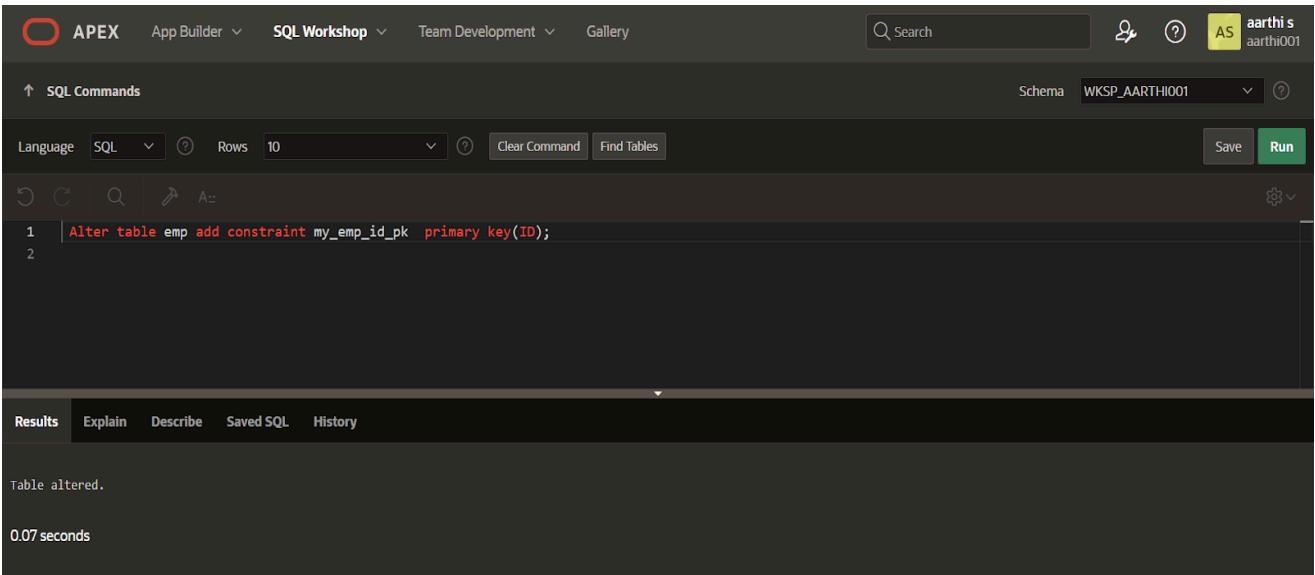
DATE:

1. Add a table-level PRIMARY KEY constraint to the EMP table on the ID column. The constraint should be named at creation. Name the constraint my_emp_id_pk.

QUERY:

```
Alter table emp add constraint my_emp_id_pk primary key(ID);
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands tab, the following SQL command is entered and executed:

```
1 | Alter table emp add constraint my_emp_id_pk primary key(ID);
2 |
```

The results pane shows the output:

```
Table altered.

0.07 seconds
```

2. Create a PRIMAY KEY constraint to the DEPT table using the ID colum. The constraint should be named at creation. Name the constraint my_dept_id_pk.

QUERY:

```
Alter table dept add constraint my_dept_id_pk primary key(ID);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: 'Alter table dept add constraint my_dept_id_pk primary key(id);'. Below the code, the results show 'Table altered.' and a execution time of '0.08 seconds'. The schema is set to 'WKSP_AARTHI001'.

```
1 Alter table dept add constraint my_dept_id_pk primary key(id);
```

Results
Table altered.
0.08 seconds

3. Add a column DEPT_ID to the EMP table. Add a foreign key reference on the EMP table that ensures that the employee is not assigned to non existent department. Name the constraint my_emp_dept_id_fk.

QUERY:

```
ALTER TABLE my_emp ADD CONSTRAINT my_emp_dept_id_fk FOREIGN KEY (id) REFERENCES dept (id);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: 'ALTER TABLE my_emp ADD CONSTRAINT my_emp_dept_id_fk FOREIGN KEY (id) REFERENCES dept (id);'. Below the code, the results show 'Table altered.' and a execution time of '0.06 seconds'. The schema is set to 'WKSP_AARTHI001'.

```
1 ALTER TABLE my_emp ADD CONSTRAINT my_emp_dept_id_fk FOREIGN KEY (id) REFERENCES dept (id);
```

Results
Table altered.
0.06 seconds

4. Modify the EMP table. Add a COMMISSION column of NUMBER data type, precision 2, scale 2. Add a constraint to the commission column that ensures that a commission value is greater than zero.

QUERY:

```
Alter table emp3 add commission ck check(commision>0);
```

OUTPUT:

```
1 alter table emp3 add[constraints ck check(comission>0)]
```

Results Explain Describe Saved SQL History

Table altered.

0.07 seconds

Restricting and Sorting data

EX_NO: 5

DATE:

1. Create a query to display the last name and salary of employees earning more than 12000.

QUERY:

```
SELECT last_name, salary FROM MY_EMPLOYEES4 WHERE salary > 12000;
```

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

↑ SQL Commands Schema WKSP_AARTHI001

Language SQL Rows 10 Save Run

```
1 SELECT last_name, salary
2 FROM MY_EMPLOYEES4
3 WHERE salary > 12000;
4
```

Results Explain Describe Saved SQL History

LAST_NAME	SALARY
Harthik	20000
Rohit	15500
aarthi	30000

2. Create a query to display the employee last name and department number for employee number 176

QUERY:

```
SELECT last_name, DEPT_ID FROM MY_EMPLOYEES4 WHERE emp_id = 176;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there are navigation links for App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, user profile information for 'aarthi s aarthi001', and a schema dropdown set to 'WKSP_AARTHI001'. Below the header, the main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab is active, displaying the following query:

```

1 SELECT last_name,DEPT_ID
2 FROM MY_EMPLOYEES
3 WHERE emp_id = 176;
4

```

The Results tab shows the output of the query:

LAST_NAME	DEPT_ID
ram	2005

Below the table, it says '1 rows returned in 0.01 seconds' and has a 'Download' link.

3. Create a query to display the last name and salary of employees whose salary is not in the range of 5000 and 12000. (hints: not between)

QUERY:

```
SELECT last_name, salary FROM my_employees4 WHERE salary NOT BETWEEN 5000 AND 12000;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different query. The SQL Commands tab is active, displaying:

```

1 SELECT last_name, salary
2 FROM my_employees4
3 WHERE salary NOT BETWEEN 5000 AND 12000;

```

The Results tab shows the output:

LAST_NAME	SALARY
Harthik	20000
ram	15500
Rohit	15500
aarthi	30000

4. Display the employee last name, job ID, and start date of employees hired between February 20,1998 and May 1,1998.order the query in ascending order by start date.(hints: between)

QUERY:

```
SELECT last_name, job_id, hire_date1 FROM my_employees4 WHERE hire_date1 BETWEEN DATE '1998-02-20' AND DATE '1998-05-01' ORDER BY hire_date1 ASC;
```

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

↑ SQL Commands Schema WKSP_AARTHI001 AS aarthi001

Language SQL Rows 10 Clear Command Find Tables Save Run

```
1 SELECT last_name, job_id, hire_date1
2 FROM my_employees4
3 WHERE hire_date1 BETWEEN DATE '1998-02-28' AND DATE '1998-05-01'
4 ORDER BY hire_date1 ASC;
```

Results Explain Describe Saved SQL History

LAST_NAME	JOB_ID	HIRE_DATE1
Revathi	1014	02/20/1998
Mark	1011	02/28/1998
Kinshuk	1015	04/29/1998

3 rows returned in 0.01 seconds Download

5. Display the last name and department number of all employees in departments 20 and 50 in alphabetical order by name.(hints: in, orderby)

QUERY:

```
SELECT last_name, dept_id FROM my_employees4 WHERE dept_id IN (20, 50) ORDER BY last_name ASC;
```

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

↑ SQL Commands Schema WKSP_AARTHI001 AS aarthi001

Language SQL Rows 10 Clear Command Find Tables Save Run

```
1 SELECT last_name, dept_id
2 FROM my_employees4
3 WHERE dept_id IN (20, 50)
4 ORDER BY last_name ASC;
5
```

Results Explain Describe Saved SQL History

LAST_NAME	DEPT_ID
Vinith	20
Vinith	20
priya	50

3 rows returned in 0.01 seconds Download

6. Display the last name and salary of all employees who earn between 5000 and 12000 and are in departments 20 and 50 in alphabetical order by name. Label the columns EMPLOYEE, MONTHLY SALARY respectively.(hints: between, in)

QUERY:

```
SELECT last_name AS EMPLOYEE, salary AS "MONTHLY SALARY" FROM my_employees4
```

```
WHERE salary BETWEEN 5000 AND 12000 AND dept_id IN (20, 50)
ORDER BY last_name ASC;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', and a search bar. The user is identified as 'aarthis aarthi001'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_AARTHI001'. The query entered is:

```
1 SELECT last_name AS EMPLOYEE, salary AS "MONTHLY SALARY"
2 FROM my_employees4
3 WHERE salary BETWEEN 5000 AND 12000
4 AND dept_id IN (20, 50)
5 ORDER BY last_name ASC;
```

The results section shows a table with two rows:

EMPLOYEE	MONTHLY SALARY
raj	7000
sam	8000

Below the table, it says '2 rows returned in 0.01 seconds' and has a 'Download' link.

- Display the last name and hire date of every employee who was hired in 1994.(hints: like)

QUERY:

```
SELECT last_name, hire_date1 FROM my_employees4 WHERE EXTRACT(YEAR FROM
hire_date1) like '%1994%';
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', and a search bar. The user is identified as 'aarthis aarthi001'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_AARTHI001'. The query entered is:

```
1 SELECT last_name, hire_date1
2 FROM my_employees4
3 WHERE EXTRACT(YEAR FROM hire_date1) like '%1994%';
```

The results section shows a table with three rows:

LAST_NAME	HIRE_DATE1
Arul	04/29/1994
Tanushree	04/29/1994
Koushik	04/29/1994

Below the table, it says '3 rows returned in 0.01 seconds' and has a 'Download' link.

- Display the last name and job title of all employees who do not have a manager.(hints: is null)

QUERY:

```
SELECT last_name, job_title FROM my_employees4 WHERE manager_id IS NULL;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, user profile 'aarthis aarthi001', and a schema dropdown set to 'WKSP_AARTHI001'. Below the navigation is a toolbar with language selection (SQL), row limit (Rows 10), and command-related buttons (Clear Command, Find Tables). The main area contains a code editor with the following SQL query:

```
1 SELECT last_name, job_title
2 FROM my_employees4
3 WHERE manager_id IS NULL;
4
```

Below the code editor, tabs for Results, Explain, Describe, Saved SQL, and History are visible. The Results tab is selected, displaying a table with two columns: LAST_NAME and JOB_TITLE. The data is as follows:

LAST_NAME	JOB_TITLE
Vinoth	Snr analyst
sam	Snr analyst
Harthik	analyst
ram	analyst

9. Display the last name, salary, and commission for all employees who earn commissions. Sort data in descending order of salary and commissions.(hints: is not null,orderby)

QUERY:

```
SELECT last_name, salary, commission FROM my_employees4 WHERE commission IS NOT NULL ORDER BY salary DESC, commission DESC;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar, schema, and toolbar are the same. The code editor contains the following SQL query:

```
1 SELECT last_name, salary, commission
2 FROM my_employees4
3 WHERE commission IS NOT NULL
4 ORDER BY salary DESC, commission DESC;
5
```

The Results tab is selected, displaying a table with three columns: LAST_NAME, SALARY, and COMMISSION. The data is as follows:

LAST_NAME	SALARY	COMMISSION
sai	40000	20
Vinoth	40000	15

At the bottom left, it says '2 rows returned in 0.01 seconds' and has a 'Download' link.

10. Display the last name of all employees where the third letter of the name is a.(hints:like)

QUERY:

```
SELECT last_name FROM my_employees4 WHERE last_name LIKE '__a%';
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'aarthis aarthi001'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_AARTHI001'. Below the title are buttons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The SQL command entered is:

```
1 SELECT last_name
2 FROM my_employees4
3 WHERE last_name LIKE '__a%';
4
```

The results section shows a single row with the value 'clark' under the 'LAST_NAME' column. A message at the bottom indicates '1 rows returned in 0.00 seconds'.

11. Display the last name and job and salary for all employees whose job is sales representative or stock clerk and whose salary is not equal to 2500 ,3500 or 7000.(hints:in,not in)

QUERY:

```
SELECT last_name, job_id, salary FROM my_employees4 WHERE job_title IN ('Sales representative', 'Stock clerk') AND salary NOT IN (2500, 3500, 7000);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'aarthis aarthi001'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_AARTHI001'. Below the title are buttons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The SQL command entered is:

```
1 SELECT last_name, job_id, salary
2 FROM my_employees4
3 WHERE job_title IN ('Sales representative', 'Stock clerk')
4 AND salary NOT IN (2500, 3500, 7000);
5
```

The results section shows a single row with values 'Vaithianathan', '1016', and '5000' under the 'LAST_NAME', 'JOB_ID', and 'SALARY' columns respectively. A message at the bottom indicates '1 rows returned in 0.01 seconds'.

12. Display the last name and job and salary for all employees whose job is sales representative or stock clerk and whose salary is not equal to 2500 ,3500 or 7000.(hints:in,not in)

QUERY:

```
SELECT last_name, job_id, salary FROM my_employees4 WHERE (job_title = 'Sales representative' or job_title= 'Stock clerk') AND salary NOT IN (2500, 3500, 7000);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query entered is:

```
1 SELECT last_name, job_id, salary
2 FROM my_employees4
3 WHERE (job_title = 'Sales representative' or job_title= 'Stock clerk')
4 AND salary NOT IN (2500, 3500, 7000);
```

The results table displays two rows:

LAST_NAME	JOB_ID	SALARY
Vaithianathan	1016	5000
Robert	1018	5000

2 rows returned in 0.00 seconds

13. Display the last name, salary, and commission for all employees whose commission amountis 20%. (hints:use predicate logic)

QUERY:

```
SELECT last_name, salary, commission FROM my_employees4 WHERE commission = 20;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query entered is:

```
1 SELECT last_name, salary, commission
2 FROM my_employees4
3 WHERE commission = 20;
```

The results table displays one row:

LAST_NAME	SALARY	COMMISSION
sai	40000	20

1 rows returned in 0.01 seconds

Evaluation Procedure	Marks awarded
Query (5)	
Execution (5)	
Viva (5)	
Total (5)	
Faculty Signature	

SINGLE ROW FUNCTIONS

EX_NO:6

DATE:

1. Write a query to display the current date. Label the column Date.

QUERY:

```
select sysdate from dual;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'aarthi s' and the schema 'WKSP_AARTHI001'. The main area is titled 'SQL Commands' with a 'Run' button. The command entered is 'SELECT CURRENT_DATE AS "date" FROM dual;'. The results section shows a single row with the date '05/12/2024' under the 'date' column. The bottom status bar indicates '1 rows returned in 0.02 seconds'.

date
05/12/2024

2. The HR department needs a report to display the employee number, last name, salary, and increased by

15.5% (expressed as a whole number) for each employee. Label the column New Salary.

QUERY:

```
select employee_id, last_name, salary, salary+(15.5/100*salary) "new_salary" from my_employees4;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there's a toolbar with various icons and a user profile for 'aarthi001'. Below the toolbar is a search bar and a schema dropdown set to 'WKSP_AARTHI001'. The main area has tabs for 'SQL Commands' and 'Run'. A SQL command is entered in the text area:

```
1 select emp_id, last_name, salary, salary+(15.5/100*salary) "new_salary" from my_employees4;
```

Below the command, there's a results grid with the following data:

	EMP_ID	LAST_NAME	SALARY	new_salary
1		Vinoth	40000	46200
23		sam	8000	9240
1		sal	40000	46200
2		Harthik	20000	23100

3. Modify your query lab_03_02.sql to add a column that subtracts the old salary from the new salary. Label the column Increase.

QUERY:

```
select employee_id, last_name, salary, salary+(15.5/100*salary)  
"new_salary", salary+(15.5/100*salary)- salary as "Increase" from my_employees4;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL command entered is:

```
1 select emp_id, last_name, salary, salary+(15.5/100*salary) "new_salary", [salary+(15.5/100*salary)]-salary as  
2 | "increase" from my_employees4;
```

The results section displays the following data:

EMP_ID	LAST_NAME	SALARY	new_salary	Increase
1	Vinoth	40000	46200	6200
23	sam	8000	9240	1240
1	sal	40000	46200	6200
2	Harthik	20000	23100	3100

4. Write a query that displays the last name (with the first letter uppercase and all other letters lowercase) and the length of the last name for all employees_table whose name starts with the letters J, A, or M. Give each column an appropriate label. Sort the results by the employees_table' last names.

QUERY:

```
Select initcap(last_name),length(last_name) as "Length_of_last_name" from my_employees4 where  
last_name like 'J%' or last_name like 'A%' or last_name like 'M%' order by last_name asc;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL command entered is:

```
1 initcap(last_name),length(last_name) as "Length_of_last_name" from my_employees4 where last_name like 'J%' or last_name like 'A%' or last_name like 'M%' order by last_n  
2  
3
```

The results section displays the following data:

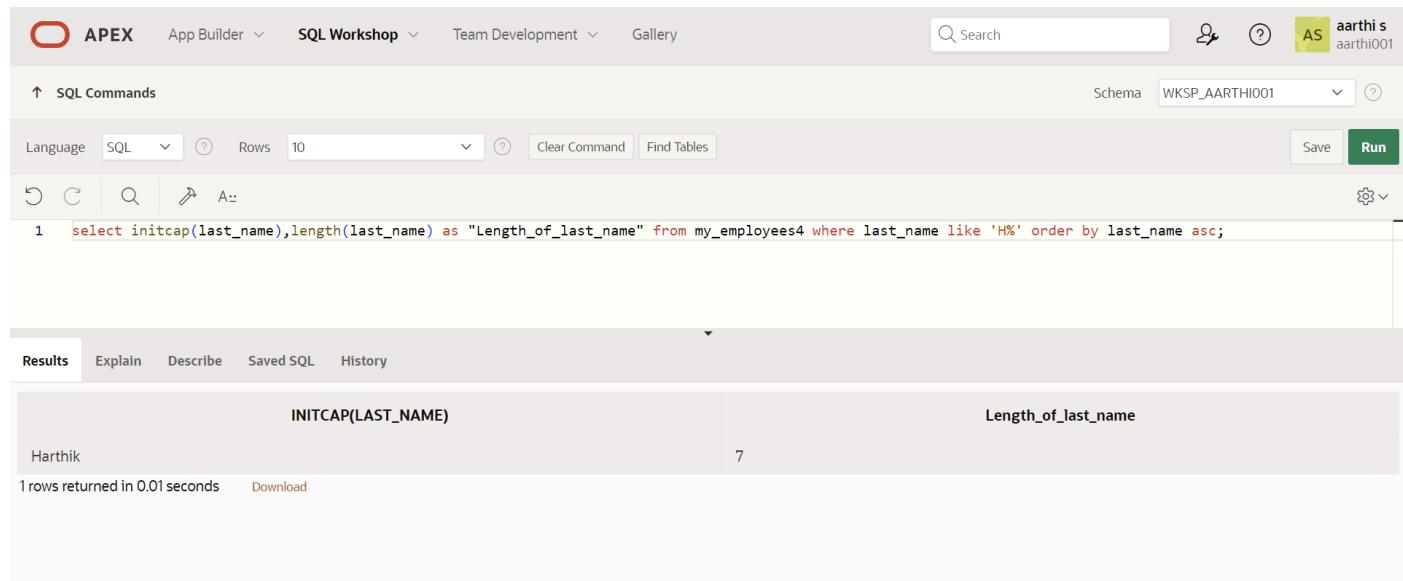
INITCAP(LAST_NAME)	Length_of_last_name
Arul	4
Joe	3
Mark	4
Mark	4

5. Rewrite the query so that the user is prompted to enter a letter that starts the last name. For example, if the user enters H when prompted for a letter, then the output should show all employees_table whose last name starts with the letter H.

QUERY:

```
select initcap(last_name),length(last_name) as "Length_of_last_name" from my_employees4 where last_name like 'H%' order by last_name asc;
```

OUTPUT:



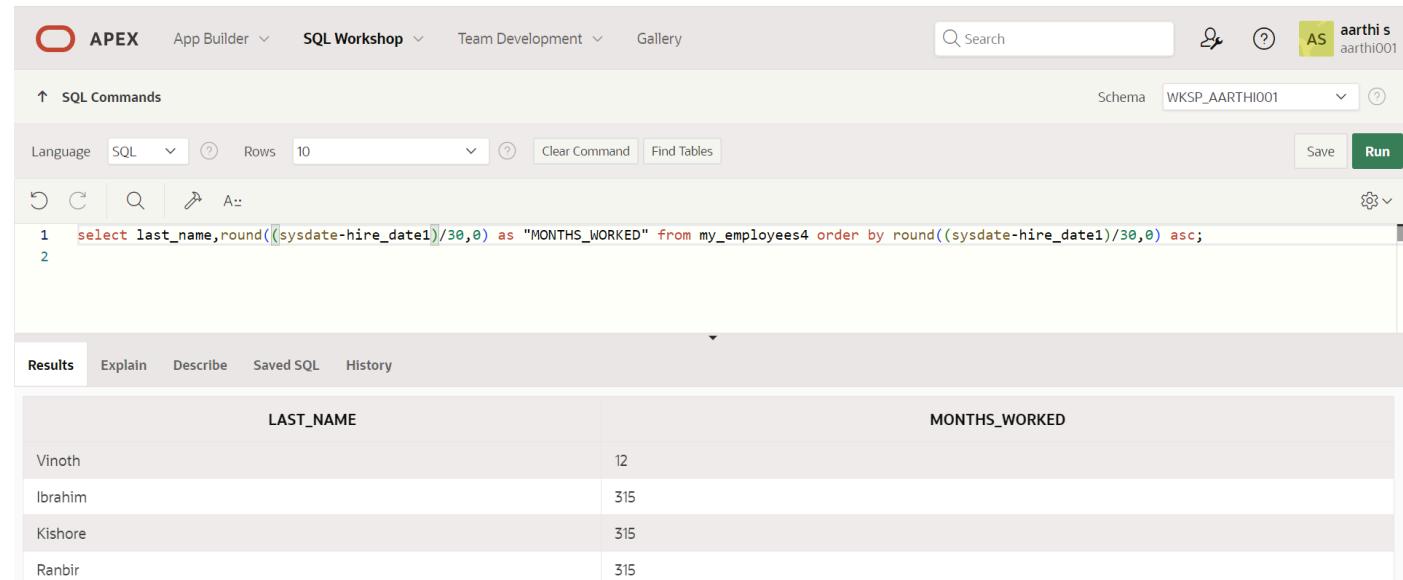
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'aarthi s' (aarthi001). The main area is titled 'SQL Commands' with a search bar and schema dropdown set to 'WKSP_AARTHI001'. Below is a toolbar with Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run buttons. The SQL editor contains the query: `select initcap(last_name),length(last_name) as "Length_of_last_name" from my_employees4 where last_name like 'H%' order by last_name asc;`. The results section shows a single row: INITCAP(LAST_NAME) is 'Harthik' and Length_of_last_name is 7. A note says '1 rows returned in 0.01 seconds'.

6. The HR department wants to find the length of employment for each employee. For each employee, display the last name and calculate the number of months between today and the date on which the employee was hired. Label the column MONTHS_WORKED. Order your results by the number of months employed. Round the number of months up to the closest whole number.

QUERY:

```
select last_name,round((sysdate-hire_date)/30,0) as "MONTHS_WORKED" from my_employees4 order by round((sysdate-hire_date)/30,0) asc;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'aarthi s' (aarthi001). The main area is titled 'SQL Commands' with a search bar and schema dropdown set to 'WKSP_AARTHI001'. Below is a toolbar with Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run buttons. The SQL editor contains the query: `select last_name,round((sysdate-hire_date)/30,0) as "MONTHS_WORKED" from my_employees4 order by round((sysdate-hire_date)/30,0) asc;`. The results section shows four rows: Vinoth (12 months), Ibrahim (315 months), Kishore (315 months), and Ranbir (315 months).

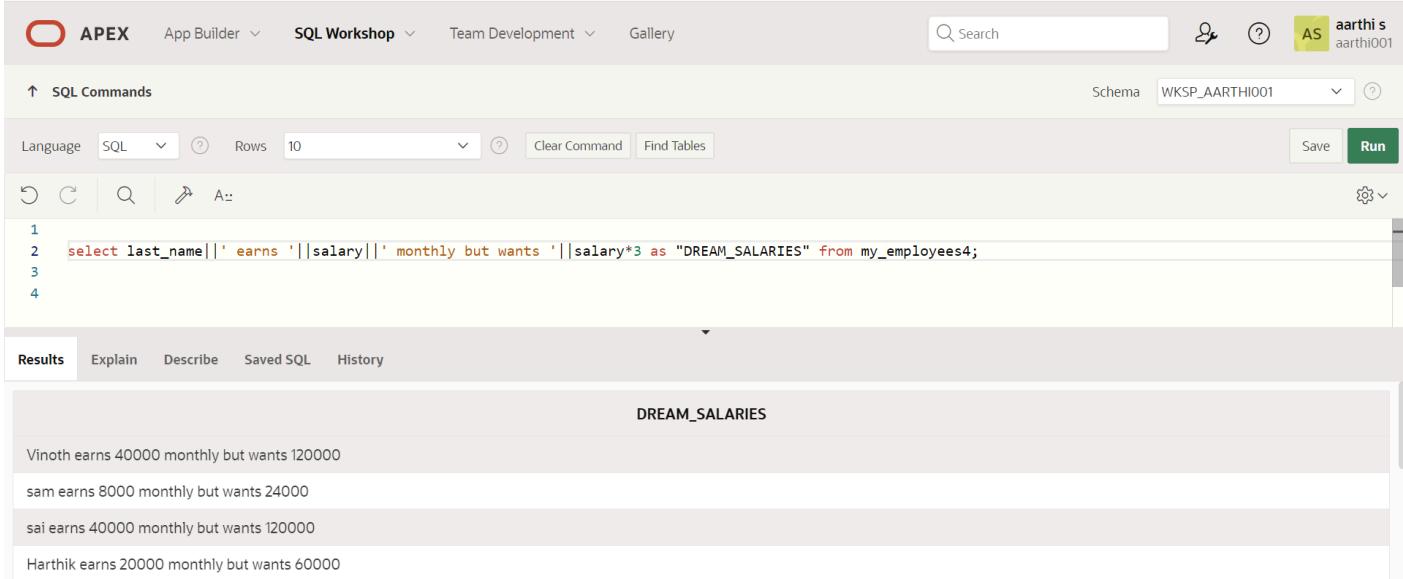
LAST_NAME	MONTHS_WORKED
Vinoth	12
Ibrahim	315
Kishore	315
Ranbir	315

7. Create a report that produces the following for each employee:
<employee last name> earns<salary>monthly but wants <3 times salary>.Label the column Dream Salaries.

QUERY:

```
select last_name||' earns'||salary||' monthly but wants'||salary*3 as "DREAM_SALARIES" from my_employees4
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The query window contains the following code:

```
1 select last_name||' earns'||salary||' monthly but wants'||salary*3 as "DREAM_SALARIES" from my_employees4;
```

The results pane displays the output:

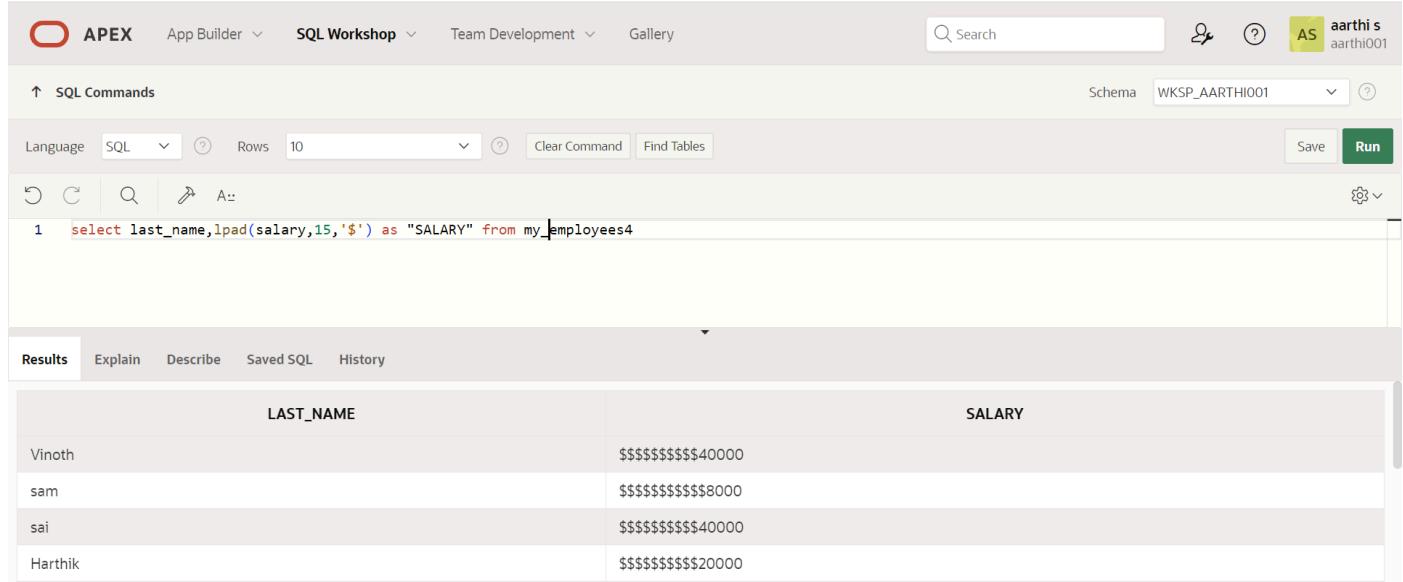
DREAM_SALARIES
Vinodh earns 40000 monthly but wants 120000
sam earns 8000 monthly but wants 24000
sai earns 40000 monthly but wants 120000
Harthik earns 20000 monthly but wants 60000

8. Create a query to display the last name and salary for all employees_table. Format the salary to be 15 characters long, left-padded with the \$ symbol. Label the column SALARY.

QUERY:

```
select last_name, lpad(salary, 15, '$') as "SALARY" from my_employees4;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The query window contains the following code:

```
1 select last_name, lpad(salary, 15, '$') as "SALARY" from my_employees4
```

The results pane displays the output:

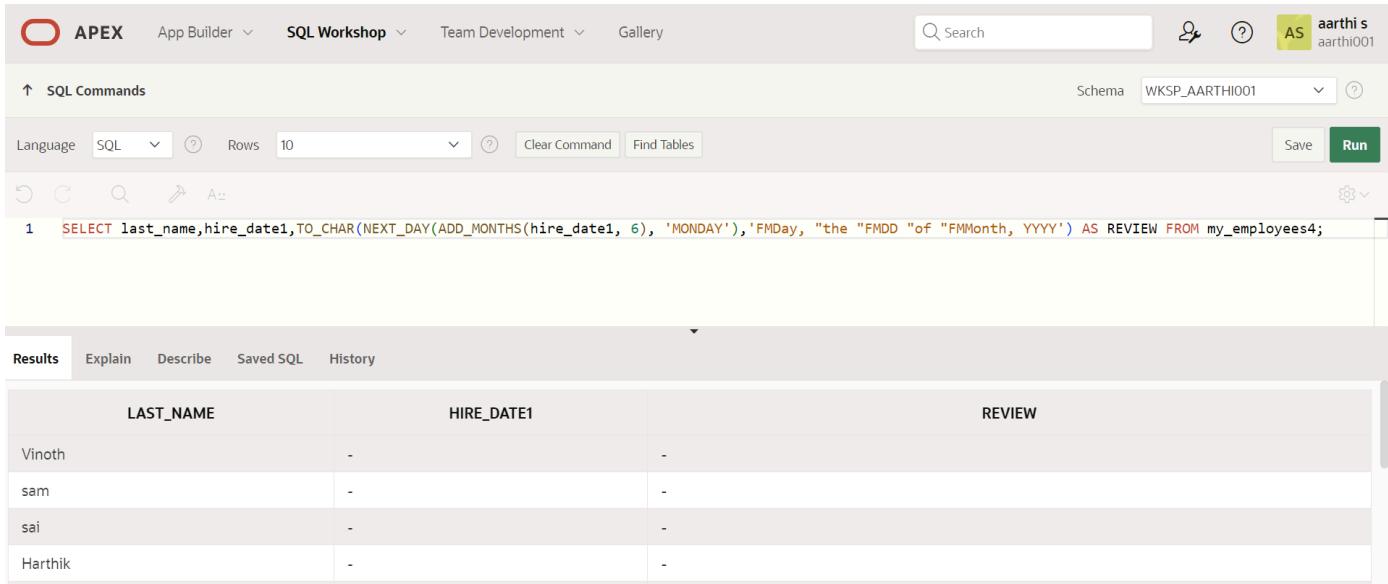
LAST_NAME	SALARY
Vinodh	\$\$\$\$\$\$\$\$\$\$40000
sam	\$\$\$\$\$\$\$\$\$\$8000
sai	\$\$\$\$\$\$\$\$\$\$40000
Harthik	\$\$\$\$\$\$\$\$\$\$20000

9. Display each employee's last name, hire date, and salary review date, which is the first Monday after six months of service. Label the column REVIEW. Format the dates to appear in the format similar to "Monday, the Thirty-First of July, 2000."

QUERY:

```
SELECT last_name,hire_date,TO_CHAR(NEXT_DAY(ADD_MONTHS(hire_date, 6), 'MONDAY'),'FMDay, "the  
"FMDD "of "FMMonth, YYYY') AS REVIEW FROM my_employees4;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user information (aarthi s, aarthi001) are also present. The SQL Commands tab is selected, showing the query: `SELECT last_name,hire_date1,TO_CHAR(NEXT_DAY(ADD_MONTHS(hire_date1, 6), 'MONDAY'),'FMDay, "the "FMDD "of "FMMonth, YYYY') AS REVIEW FROM my_employees4;`. The results tab displays the output:

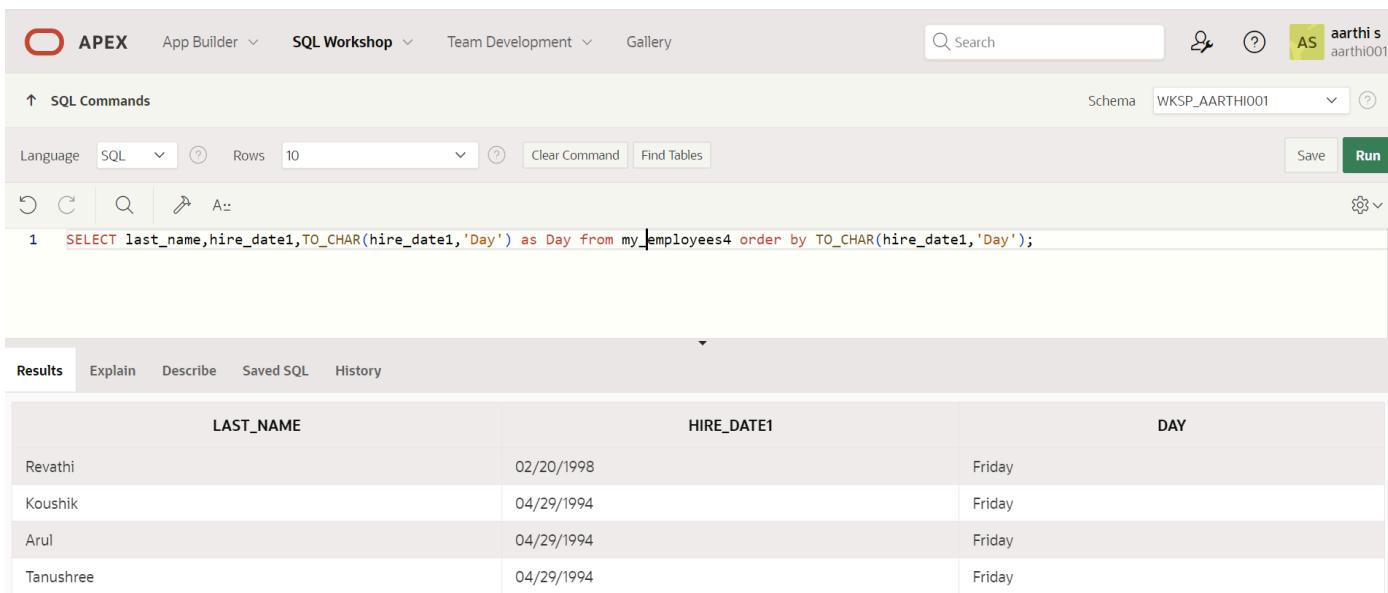
LAST_NAME	HIRE_DATE1	REVIEW
Vinoth	-	-
sam	-	-
sai	-	-
Harthik	-	-

10. Display the last name, hire date, and day of the week on which the employee started. Label the column DAY. Order the results by the day of the week, starting with Monday.

QUERY:

```
SELECT last_name,hire_date,TO_CHAR(hire_date,'Day') as Day from my_ employees4 order by  
TO_CHAR(hire_date,'Day');
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user information (aarthi s, aarthi001) are also present. The SQL Commands tab is selected, showing the query: `SELECT last_name,hire_date1,TO_CHAR(hire_date1,'Day') as Day from my_ employees4 order by TO_CHAR(hire_date1,'Day');`. The results tab displays the output:

LAST_NAME	HIRE_DATE1	DAY
Revathi	02/20/1998	Friday
Koushik	04/29/1994	Friday
Arul	04/29/1994	Friday
Tanushree	04/29/1994	Friday

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

RESULT:

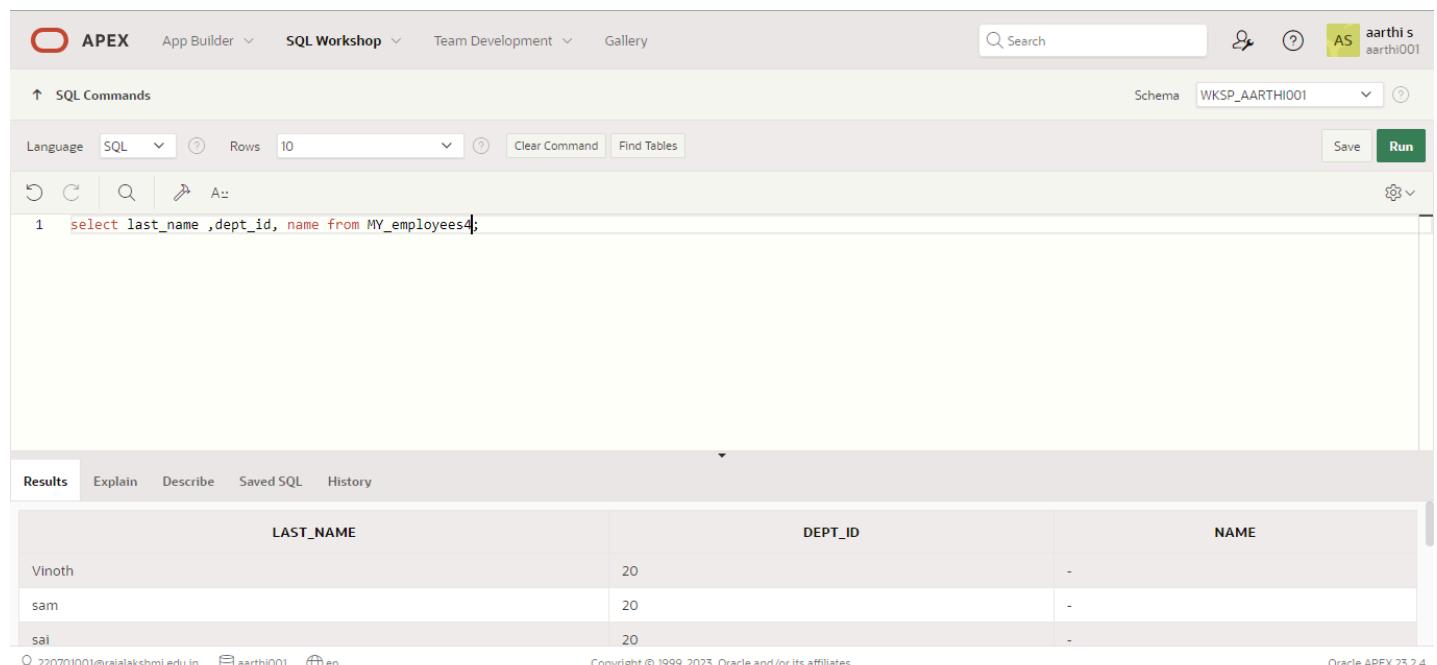
Evaluation Procedure	Marks awarded
Query (5)	
Execution (5)	
Viva (5)	
Total (5)	
Faculty Signature	

EXNO:7 DISPLAYING DATA FROM MULTIPLE TABLES DATE:

1. Write a query to display the last name, department number, and department name for all employees.

QUERY: select lname ,dept_id, dept_name from MY_emp;

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The query entered is:

```
1 select last_name ,dept_id, name from MY_employees;
```

The results section displays the following data:

LAST_NAME	DEPT_ID	NAME
Vinoth	20	-
sam	20	-
sai	20	-

At the bottom, it shows the user information: 220701001@rajalakshmi.edu.in, aarthi001, en, Copyright © 1999, 2023, Oracle and/or its affiliates, and Oracle APEX 23.2.4.

2. Create a unique listing of all jobs that are in department 80. Include the location of the department in the output.

QUERY: SELECT DISTINCT job_id, loc_id FROM My_emp, dept WHERE My_emp.dept_id = dept.dept_id and My_emp.d_id=80;

OUTPUT:

Language: SQL Rows: 10 Clear Command Find Tables

```
1 | SELECT DISTINCT job_id, location_id FROM My_employees4, depcse1 WHERE My_employees4.dept_id = depcse1.id and My_employees4.dept_id=80;
2 | 
```

Results Explain Describe Saved SQL History

no data found

3. Write a query to display the employee last name, department name, location ID, and city of all employees who earn a commission

QUERY: SELECT My_emp.lname, My_emp.job_title, dept.loc_id, My_emp.city FROM My_emp, dept WHERE My_emp.dept_id = dept.dept_id AND My_emp.commission IS NOT NULL;

OUTPUT:

Language: SQL Rows: 10 Clear Command Find Tables

```
1 | My_employees4.last_name,My_employees4.job_title, my_employees4.location_id, My_employees4.city FROM My_employees4, depcse1 WHERE My_employees.dept_id = depcse1.id AND My_
2 | 
```

Results Explain Describe Saved SQL History

ORA-00900: invalid SQL statement

0.01 seconds

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	

Faculty Signature

EXNO:8 AGGREGATING DATA USING GROUP FUNCTIONS DATE:

1. Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number

QUERY: select max(salary) as "MAXIMUM", min(salary) as "MINIMUM", sum(salary) as "SSUM", round(avg(salary),2) as "average" from My_emp;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query entered is:

```
1 | select max(salary) as "MAXIMUM", min(salary) as "MINIMUM", sum(salary) as "SSUM", round(avg(salary),2) as "average" from My_employees;
```

The results section displays the following data:

	MAXIMUM	MINIMUM	SSUM	average
	40400	1000	52600	13150

1 rows returned in 0.01 seconds Download

2. Modify the above query to display the minimum, maximum, sum, and average salary for each job type.

QUERY: select job_title, max(salary) as "MAXIMUM", min(salary) as "MINIMUM", sum(salary) as "SSUM", round(avg(salary),2) as "average" from My_emp group by job_title;

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands Schema WKSP_AARTHI001

Language SQL Rows 10 Clear Command Find Tables Save Run

```
1 | select job_title,max(salary) as "MAXIMUM", min(salary) as "MINIMUM",sum(salary) as "SSUM",round(avg(salary),2) as "average" from My_employees4 group by job_title;
```

Results Explain Describe Saved SQL History

JOB_TITLE	MAXIMUM	MINIMUM	SSUM	average
Stock Clerk	-	-	-	-
developer	1000	1000	1000	1000
Snr analyst	40400	1200	41600	20800

3. Determine the number of managers without listing them. Label the column Number of Managers. Hint: Use the MANAGER_ID column to determine the number of managers.

QUERY: select count(mg_id) as "MANAGER" from My_emp;

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands Schema WKSP_AARTHI001

Language SQL Rows 10 Clear Command Find Tables Save Run

```
1 | select count(manager_id) as "MANAGER" from My_employees4;
```

Results Explain Describe Saved SQL History

MANAGER
5

1 rows returned in 0.00 seconds Download

4. Find the difference between the highest and lowest salaries. Label the column DIFFERENCE

QUERY: select max(salary)-min(salary) as "DIFFERENCE"from My_emp;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema is set to WKSP_AARTH1001. The query editor contains the following SQL code:

```

1 select max(salary)-min(salary) as "DIFFERENCE"from My_employees4;
2

```

The results section displays the output of the query:

DIFFERENCE
39400

Below the results, it says "1 rows returned in 0.01 seconds" and provides a "Download" link. The bottom of the page shows copyright information and the version Oracle APEX 23.2.4.

EXNO:9 SUB QUERIES DATE:

1.The HR department needs a query that prompts the user for an employee last name. The query then displays the last name and hire date of any employee in the same department as the employee whose name they supply (excluding that employee). For example, if the user enters Zlotkey, find all employees who work with Zlotkey (excluding Zlotkey)

QUERY: select * from My_emp where lname='zlotkey';

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema is set to WKSP_AARTH1001. The query editor contains the following SQL code:

```

1 select * from My_employees4 where last_name='zlotkey';
2

```

The results section displays the output of the query:

LAST_NAME	JOB_ID	DEPT_ID	HIRE_DATE	EMP_ID	SALARY	START_DATE	JOB_TITLE	HIRE_DATE1	HIRE_DATE2	START_DATE2	MANAGER_ID	COMMISSION	DEPT_NUMBER
zlotkey	1234	4567	-	1003	-	-	IT	-	-	-	1111	10	4141

Below the results, it says "1 rows returned in 0.02 seconds" and provides a "Download" link.

2. Create a report that displays the employee number, last name, and salary of all employees who earn more than the average salary. Sort the results in order of ascending salary.

QUERY: select emp_id, lname, salary from My_emp where salary > (select avg(salary) from My_emp) order by salary asc;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The schema is set to 'WKSP_AARTHI001'. In the SQL Commands tab, the following query is entered:

```
1 select emp_id, last_name, salary from My_employees4 where salary > (select avg(salary) from My_emp) order by salary asc;
```

The Results tab shows the output: "no data found".

3. Write a query that displays the employee number and last name of all employees who work in a department with any employee whose last name contains a u.

QUERY: select emp_id,lname from My_emp where id_dept in(select id_dept from My_emp where lname like '%u%');

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The schema is set to 'WKSP_AARTHI001'. In the SQL Commands tab, the following query is entered:

```
1 select emp_id, last_name from My_employees4 where dept_id in(select dept_id from My_employees4 where last_name like '%u%');
```

The Results tab displays the following table:

EMP_ID	LAST_NAME
110	Vinod
23	sam
1	sai

At the bottom, the footer includes the URL 220701001@rajalakshmi.edu.in, the user aarthi001, and the page number en. Copyright information and the version Oracle APEX 23.2.4 are also present.

4. The HR department needs a report that displays the last name, department number, and job ID of all employees whose department location ID is 1700.

QUERY: select emp_id,lname from my_emp where loc_id in(select loc_id from My_emp where loc_id=1700);

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links for App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, user information (aarthi001), and a schema dropdown set to WKSP_AARTHI001. Below the header, the SQL Commands tab is selected. The SQL editor contains the following code:

```

1 select emp_id, last_name from my_employees4 where location_id in(select location_id from My_employees4 where location_id=1700);
2

```

Below the editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is active, showing the message "no data found".

5. Create a report for HR that displays the department number, last name, and job ID for every employee in the Executive department.

QUERY: select emp_id, lname, id_dept from My_emp where id_dept in (select id_dept from My_emp where job_title = 'ex');

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links for App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, user information (aarthi001), and a schema dropdown set to WKSP_AARTHI001. Below the header, the SQL Commands tab is selected. The SQL editor contains the following code:

```

1 select emp_id, last_name, dept_id from My_employees4 where dept_id in (select dept_id from My_employees4 where job_title = 'ex');
2
3

```

Below the editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is active, showing the message "no data found".

6. Modify the query 3 to display the employee number, last name, and salary of all employees who earn more than the average salary and who work in a department with any employee whose last name contains a u.

QUERY: select emp_id, lname, salary from My_emp where salary > (select avg(salary) from my_emp) and id_dept in (select id_dept from My_emp where lname like '%u%');

OUTPUT:

Language: SQL Rows: 10

```

1   me, salary from My_employees4 where salary > (select avg(salary) from my_employees4) and dept_id in (select dept_id from My_employees4 where last_name like '%u%');
2
3
4

```

Results

EMP_ID	LAST_NAME	SALARY
110	Vinoth	40400

1 rows returned in 0.01 seconds [Download](#)

EXNO:10 USING THE SET OPERATORS DATE:

1. The HR department needs a list of department IDs for departments that do not contain the job ID ST_CLERK. Use set operators to create this report.

QUERY: SELECT dept_id,job_id FROM dept MINUS SELECT dept_id,job_id FROM dept where job_id='clrk';

OUTPUT:

Language: SQL Rows: 10

```

1   SELECT dept_id,job_id FROM my_employees4 MINUS SELECT dept_id,job_id FROM my_employees4 where job_id='clrk';

```

Results

ORA-01722: invalid number

2. The HR department needs a list of countries that have no departments located in them.

Display the country ID and the name of the countries. Use set operators to create this report.

QUERY: SELECT coun_id,coun_name FROM dept MINUS SELECT coun_id,coun_name FROM dept WHERE id_dept is not NULL;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area contains a SQL command:

```
1 SELECT country_id FROM my_employees4 MINUS SELECT country_id FROM depcse1 WHERE id is not NULL;
2
```

The command is highlighted in red, indicating an error. Below the command, the results tab is selected, showing the following message:

✖ Error at line 1/5: ORA-00904: "COUNTRY_ID": invalid identifier

3. Produce a list of jobs for departments 10, 50, and 20, in that order. Display job ID and department ID using set operators.

QUERY: select job_id,id_dept from dept where dept_id= 30 union all select job_id,id_dept from dept where dept_id=50 union all select job_id,id_dept from dept where dept_id= 20;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL command is identical to the one in the previous screenshot:

```
1 | SELECT country_id,country_name FROM my_employees4 MINUS SELECT country_id,country_name FROM my_employees4 WHERE dept_id is not NULL;
```

The results tab is selected, and the output is displayed as:

no data found

4. Create a report that lists the employee IDs and job IDs of those employees who currently have a job title that is the same as their job title when they were initially hired by the company (that is, they changed jobs but have now gone back to doing their original job).

QUERY: select job_id,hire_job from dept intersect select job_id,hire_job from dept where job_id=hire_job;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user information for 'aarthis' (schema 'WKSP_AARTHI001'), and buttons for Save and Run. Below the header, the SQL command window shows a single line of code: 'select job_id,hire_date from my_employees4 intersect select job_id,hire_date from my_employees4 where job_id=hire_date;'. The results tab is selected, showing the message 'no data found'.

5. The HR department needs a report with the following specifications:

- Last name and department ID of all the employees from the EMPLOYEES table, regardless of whether or not they belong to a department.
- Department ID and department name of all the departments from the DEPARTMENTS table, regardless of whether or not they have employees working in them Write a compound query to accomplish this.

QUERY: SELECT lname,id_dept FROM My_emp UNION SELECT dept_name,id_dept FROM dept;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface again. The SQL command window contains the query: 'SELECT last_name,dept_id FROM My_employees4 UNION SELECT name,id FROM dept;'. The results tab displays the output in a table format:

LAST_NAME	DEPT_ID
Arul	20
Harthik	2002
Ibrahim	20

At the bottom, there are footer links for copyright (Copyright © 1999, 2023, Oracle and/or its affiliates) and version (Oracle APEX 23.2.4).

EXNO:11 CREATING VIEWS DATE: 1. Create a view called EMPLOYEE_VU based on the employee numbers, employee names

and department numbers from the EMPLOYEES table. Change the heading for the employee name to EMPLOYEE.

Query: create view EMPLOYEE_VU as select emp_id, lname as "EMPLOYEE", id_dept from My_emp;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user information (aarthi s aarthi001) are also present. The main workspace is titled 'SQL Commands' and contains a single line of SQL code: 'create view EMPLOYEE_VU as select emp_id, last_name as "EMPLOYEE", dept_id from My_employees4'. Below the code, the 'Results' tab is selected, showing the message 'View created.' and a execution time of '0.02 seconds'.

2. Display the contents of the EMPLOYEES_VU view.

Query: select * from EMPLOYEE_VU;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar and workspace are identical to the previous screenshot. The main workspace contains the query 'select * from EMPLOYEE_VU;'. The results are displayed in a table format below:

EMP_ID	EMPLOYEE	DEPT_ID
110	Vinod	20
23	sam	20
1	sai	20

At the bottom of the results page, there are footer links for '220701001@rajalakshmi.edu.in', 'aarthi001', and 'en', along with copyright and Oracle APEX version information.

3. Using your EMPLOYEES_VU view, enter a query to display all employees names and department.

Query: select employee,id_dept from EMPLOYEE_VU;

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands Schema WKSP_AARTHI001

Language SQL Rows 10 Clear Command Find Tables Save Run

1 select employee,dept_id from EMPLOYEE_VU;

2

Results

EMPLOYEE	DEPT_ID
Vinoth	20
sam	20
sai	20

Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

4. Create a view named DEPT50 that contains the employee number, employee last names and department numbers for all employees in department 50. Label the view columns EMPNO, EMPLOYEE and DEPTNO. Do not allow an employee to be reassigned to another department through the view.

Query: create view DEPT50 as select lname as "EMPLOYEE",id_dept as "DEPTNO" from My_emp;

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands Schema WKSP_AARTHI001

Language SQL Rows 10 Clear Command Find Tables Save Run

1 create view DEPT50 as select last_name as "EMPLOYEE",id_dept as "DEPTNO" from My_employees4;

Error at line 1/54: ORA-00904: "ID_DEPT": invalid identifier

1. create view DEPT50 as select last_name as "EMPLOYEE",id_dept as "DEPTNO" from My_employees4;

5. Display the structure and contents of the DEPT50 view.

Query: select * from DEPT50;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there are search, user profile, and schema selection (WKSP_AARTHI001) buttons. The main area has a toolbar with icons for SQL, rows, clear command, and find tables. Below the toolbar, a code editor shows the following SQL command:

```
1 select * from DEPT50;
2
```

Under the results tab, a table is displayed with three columns: EMPNO, EMPLOYEE, and DEPTNO. One row is shown with values 1, priya, and 50. At the bottom left, it says "1 rows returned in 0.02 seconds".

6. Attempt to reassign Matos to department 80.

Query: UPDATE DEPT50 SET DEPTNO = 30 WHERE EMPLOYEE = 'Matos';

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The code editor contains the following SQL command:

```
1 select * from DEPT50; UPDATE DEPT50 SET DEPTNO = 30 WHERE EMPLOYEE = 'Matos';
2
```

Below the code editor, an error message is displayed: "Error at line 1/22: ORA-00933: SQL command not properly ended".

7. Create a view called SALARY_VU based on the employee last names, department names, salaries, and salary grades for all employees. Use the Employees, DEPARTMENTS and JOB_GRADE tables. Label the column Employee, Department, salary, and Grade respectively.

Query: create view SALARY_VU as select lname as "employee", job_title as "DEPARTMENT", salary from My_emp;

OUTPUT:

EXNO:13 CREATING VIEWS DATE:

1.Create a simple view called view_d_songs that contains the ID, title and artist from the DJs on Demand table for each “New Age” type code. In the subquery, use the alias “Song Title” for the title

column.

QUERY: CREATE VIEW view_d_songs AS SELECT id, title as "Song Title", artist from d_songs;

OUTPUT:



2. SELECT * FROM view_d_songs. What was returned?

QUERY: select * from d_songs;

OUTPUT:



3. REPLACE

view_d_songs. Add type_code to the column list. Use aliases for all columns.

QUERY: CREATE OR REPLACE VIEW view_d_songs AS SELECT d_songs.id, d_songs.title "Song Title", d_songs.artist, d_songs.type_code from d_songs;

OUTPUT:



4. Jason Tsang, the disk jockey for DJs on Demand, needs a list of the past events and those planned for the coming months so he can make arrangements for each event's equipment setup. As the company manager, you do not want him to have access to the price that clients paid for their events. Create a view for Jason to use that displays the name of the event, the event date, and the theme description. Use aliases for each column name.

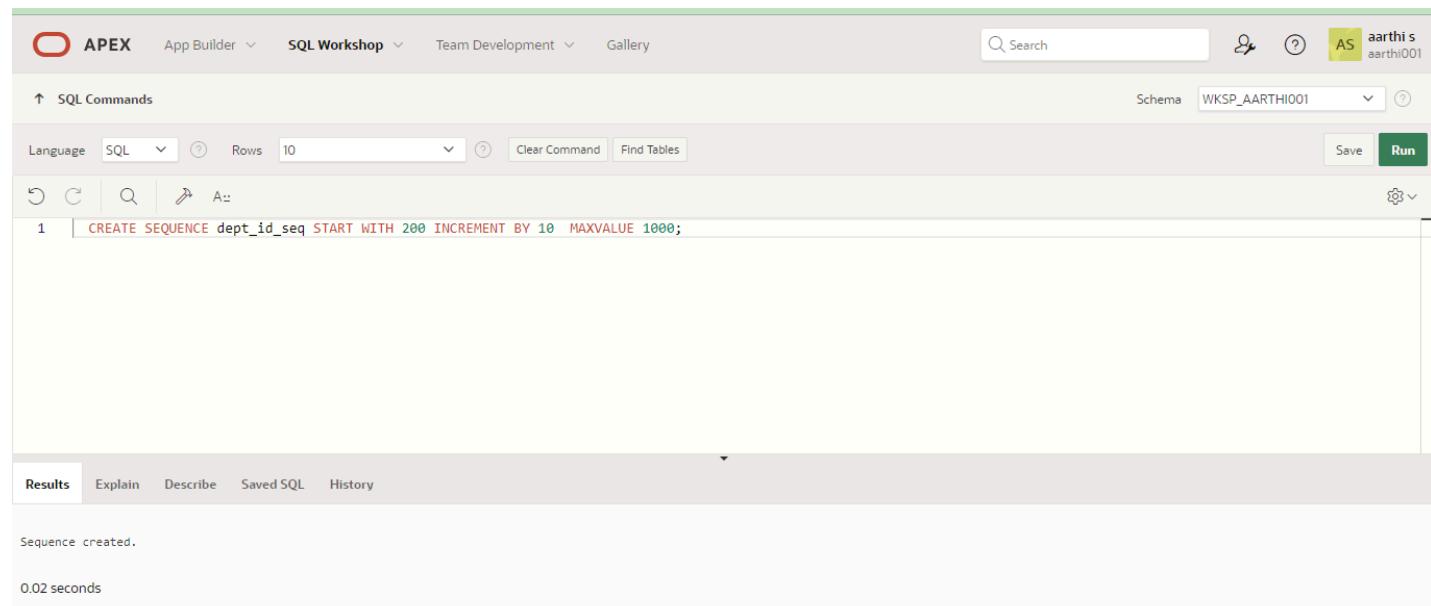
QUERY: CREATE OR REPLACE VIEW view_d_events_pkgs AS SELECT d_name as "Name of Event", TO_CHAR(d_date, 'dd-Month-yyyy') as "Event date", theme as "Theme description" FROM d_events;

OUTPUT:

EXNO:14 OTHER DATABASE OBJECTS DATE:

1. Create a sequence to be used with the primary key column of the DEPT table. The sequence should start at 200 and have a maximum value of 1000. Have your sequence increment by ten numbers. Name the sequence DEPT_ID_SEQ.

Query: CREATE SEQUENCE dept_id_seq START WITH 200 INCREMENT BY 10 MAXVALUE 1000;
OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there are user profile icons and a search bar. The main workspace is titled 'SQL Commands' and shows the following command in the editor:

```
CREATE SEQUENCE dept_id_seq START WITH 200 INCREMENT BY 10 MAXVALUE 1000;
```

Below the editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is active and displays the output: "Sequence created." and "0.02 seconds".

2. Write a query in a script to display the following information about your sequences: sequence name, maximum value, increment size, and last number.

QUERY: SELECT sequence_name, max_value, increment_by, last_number FROM

user_sequences;

OUTPUT:

A screenshot of the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, user profile 'aarthi s aarthi001', and a 'Run' button. The main area shows a SQL command in the editor:

```
1 | sequence_name, max_value, increment_by, last_number FROM user_sequences;
2 |
```

The results tab is selected, showing a yellow error message box with the text 'ORA-00900: invalid SQL statement'. Below the message, it says '0.01 seconds'.

3. Write a script to insert two rows into the DEPT table. Name your script lab12_3.sql. Be sure to use the sequence that you created for the ID column. Add two departments named Education and Administration. Confirm your additions. Run the commands in your script.

QUERY: INSERT INTO dept VALUES (dept_id_seq.nextval, 'Education'); INSERT INTO dept VALUES (dept_id_seq.nextval, 'Administration');

OUTPUT:

A screenshot of the Oracle SQL Workshop interface, identical to the previous one but with different results. The SQL command is the same:

```
1 | INSERT INTO dept VALUES (dept_id_seq.nextval, 'Education'); INSERT INTO dept VALUES (dept_id_seq.nextval, 'Administration');
```

The results tab shows a yellow error message box with the text 'Error at line 1/59: ORA-00933: SQL command not properly ended'. Inside the box, the full SQL command is displayed:

```
1. INSERT INTO dept VALUES (dept_id_seq.nextval, 'Education'); INSERT INTO dept VALUES (dept_id_seq.nextval, 'Administration');
```

4. Create a nonunique index on the foreign key column (DEPT_ID) in the EMP table. QUERY:

CREATE INDEX emp_dept_id_idx ON emp (dept_id); **OUTPUT:**



5. Display the indexes and uniqueness that exist in the data dictionary for the EMP table. QUERY:
SELECT index_name, table_name, uniqueness FROM user_indexes WHERE table_name = 'EMP';

OUTPUT:

Exno:15 CONTROLLING USER ACCESS date:

1. What privilege should a user be given to log on to the Oracle Server? Is this a system or an object privilege?

Ans: The CREATE SESSION system privilege

2. What privilege should a user be given to create tables?

Ans : The CREATE TABLE privilege

3. If you create a table, who can pass along privileges to other users on your table?

Ans: You can, or anyone you have given those privileges to by using the WITH GRANT OPTION.

4. You are the DBA. You are creating many users who require the same system privileges. What should you use to make your job easier?

Ans: Create a role containing the system privileges and grant the role to the users

5. What command do you use to change your password?

Ans: The ALTER USER statement

6. Grant another user access to your DEPARTMENTS table. Have the user grant you query access to his or her DEPARTMENTS table. Ans: Team 2 executes the GRANT statement.

GRANT select
ON departments
TO <user1>;

Team 1 executes the GRANT statement.

```
GRANT select  
ON departments  
TO <user2>;
```

WHERE user1 is the name of team 1 and user2 is the name of team 2. **7. Query all the rows in your DEPARTMENTS table.**

Ans: SELECT * FROM departments;

8. Add a new row to your DEPARTMENTS table. Team 1 should add Education as department number 500. Team 2 should add Human Resources department number 510. Query the other team's table.

Ans: Team 1 executes this INSERT statement.

```
INSERT INTO departments(department_id, department_name) VALUES (500, 'Education');  
COMMIT;
```

Team 2 executes this INSERT statement.

```
INSERT INTO departments(department_id, department_name) VALUES (510,  
'Administration');  
COMMIT;
```

9. Create a synonym for the other team's DEPARTMENTS table.

Ans: Team 1 creates a synonym named team2.

```
CREATE SYNONYM team2  
FOR <user2>.DEPARTMENTS;
```

Team 2 creates a synonym named team1.

```
CREATE SYNONYM team1  
FOR <user1>. DEPARTMENTS;
```

10. Query all the rows in the other team's DEPARTMENTS table by using your synonym.

Ans: Team 1 executes this SELECT statement.

```
SELECT *  
FROM team2;
```

Team 2 executes this SELECT statement.

```
SELECT *  
FROM team1;
```

11. Query the USER_TABLES data dictionary to see information about the tables that you own.

Ans: SELECT table_name FROM user_tables;

12. Query the ALL_TABLES data dictionary view to see information about all the tables that you can access. Exclude tables that you own.

Ans: SELECT table_name, owner FROM all_tables
WHERE owner <> <your account>;

13. Revoke the SELECT privilege from the other team.

Ans: Team 1 revokes the privilege.

```
REVOKE select  
ON departments  
FROM user2;
```

Team 2 revokes the privilege.

```
REVOKE select  
ON departments  
FROM user1;
```

14. Remove the row you inserted into the DEPARTMENTS table in step 8 and save the changes.

Team 1 executes this INSERT statement.

Ans:

```
DELETE FROM departments  
WHERE department_id = 500;  
COMMIT;
```

Team 2 executes this INSERT statement.

```
DELETE FROM departments  
WHERE department_id = 510;  
COMMIT;
```

EXNO:16 PL/SQL CONTROL STRUCTURES

1. Write a PL/SQL block to calculate the incentive of an employee whose ID is 110.

QUERY:

```
DECLARE  
incentive NUMBER(8,2);  
BEGIN  
SELECT salary*0.12 INTO incentive  
FROM employees  
WHERE employee_id = 110;
```

```
DBMS_OUTPUT.PUT_LINE('Incentive = ' || TO_CHAR(incentive));
END;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'aarthi s' (aarthi001). The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_AARTHI001'. The code editor contains the following PL/SQL block:

```
1 DECLARE
2   incentive NUMBER(8,2);
3   BEGIN
4     SELECT salary*0.12 INTO incentive
5     FROM my_employees4
6     WHERE emp_id = 110;
7     DBMS_OUTPUT.PUT_LINE('Incentive = ' || TO_CHAR(incentive));
8   END;
```

The results tab shows the output of the query:

```
Incentive = 4848
Statement processed.

0.02 seconds
```

2.) Write a PL/SQL block to show an invalid case-insensitive reference to a quoted and without quoted user-defined identifier

QUERY:

```
DECLARE
WELCOME varchar2(10) := 'welcome';
BEGIN
DBMS_Output.Put_Line("Welcome");
END;
/
```

```
/
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. On the right, there's a search bar, a help icon, and a session identifier 'aarthi s aarthi001'. The main workspace is titled 'SQL Commands'. It has dropdowns for 'Language' (set to 'SQL') and 'Rows' (set to 10), along with buttons for 'Clear Command' and 'Find Tables'. On the far right are 'Save' and 'Run' buttons. The code area contains the following PL/SQL block:

```

1  DECLARE
2  |  "WELCOME" varchar2(10) := 'welcome';
3  BEGIN
4  |  DBMS_Output.Put_Line(Welcome); A
5  END;

```

Below the code, the results tab is active, showing the output of the command:

```

welcome
Statement processed.

0.00 seconds

```

3.) Write a PL/SQL block to adjust the salary of the employee whose ID 122.

QUERY:

```

DECLARE
    salary_of_emp NUMBER(8,2);
PROCEDURE approx_salary (
    emp NUMBER,
    empsal IN OUT NUMBER,
    addless NUMBER
) IS
BEGIN
    empsal := empsal + addless;
END;

BEGIN
    SELECT salary INTO salary_of_emp
    FROM employees
    WHERE employee_id = 122;
    DBMS_OUTPUT.PUT_LINE
    ('Before invoking procedure, salary_of_emp: ' || salary_of_emp);
    approx_salary (100, salary_of_emp, 1000);
    DBMS_OUTPUT.PUT_LINE
    ('After invoking procedure, salary_of_emp: ' || salary_of_emp);
END;
/

```

OUTPUT:



4.) Write a PL/SQL block to create a procedure using the "IS [NOT] NULL Operator" and show AND operator returns TRUE if and only if both operands are TRUE.

QUERY:

```
CREATE OR REPLACE PROCEDURE pri_bool(
  boo_name VARCHAR2,
  boo_val BOOLEAN
) IS
BEGIN
  IF boo_val IS NULL THEN
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = NULL');
  ELSIF boo_val = TRUE THEN
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = TRUE');
  ELSE
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = FALSE');
  END IF;
END;
/
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'App Builder', 'SQL Workshop' (which is active), 'Team Development', and 'Gallery' are visible. On the right, there's a search bar, a user icon for 'aarthi s', and a schema dropdown set to 'WKSP_AARTHI001'. The main area is titled 'SQL Commands' and contains a code editor with the following PL/SQL code:

```

1 CREATE OR REPLACE PROCEDURE pri_bool(
2   | boo_name VARCHAR2,
3   | boo_val BOOLEAN
4 ) IS
5 BEGIN
6   IF boo_val IS NULL THEN
7     DBMS_OUTPUT.PUT_LINE( boo_name || ' = NULL');
8   ELSIF boo_val = TRUE THEN
9     DBMS_OUTPUT.PUT_LINE( boo_name || ' = TRUE');
10  ELSE
11    DBMS_OUTPUT.PUT_LINE( boo_name || ' = FALSE');
12 END IF;

```

Below the code editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the output: 'Procedure created.' and '0.04 seconds'.

5.) Write a PL/SQL block to describe the usage of LIKE operator including wildcard characters and escape character.

QUERY:

```

DECLARE
PROCEDURE pat_match (
test_string VARCHAR2,
pattern VARCHAR2
) IS
BEGIN
IF test_string LIKE pattern THEN
DBMS_OUTPUT.PUT_LINE ('TRUE');
ELSE
DBMS_OUTPUT.PUT_LINE ('FALSE');
END IF;
END;
BEGIN
pat_match('Blweate', 'B%a_e');
pat_match('Blweate', 'B%A_E');
END;
/

```

OUTPUT:

A screenshot of the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema is set to WKSP_AARTHI001. The main area shows a PL/SQL code editor with the following content:

```

8  DBMS_OUTPUT.PUT_LINE ('TRUE');
9  ELSE
10 DBMS_OUTPUT.PUT_LINE ('FALSE');
11 END IF;
12 END;
13 BEGIN
14  pat_match('Blweate', 'B%a_e');
15  pat_match('Blweate', 'B%A_E');
16 END;
17 /
18

```

The code is run, and the results panel shows:

Results Explain Describe Saved SQL History

TRUE
FALSE

Statement processed.

0.01 seconds

6.) Write a PL/SQL program to arrange the number of two variable in such a way that the small number will store in num_small variable and large number will store in num_large variable

QUERY:

DECLARE

num_small NUMBER := 8;

num_large NUMBER := 5;

num_temp NUMBER;

BEGIN

IF num_small > num_large THEN

num_temp := num_small;

num_small := num_large;

num_large := num_temp;

END IF;

DBMS_OUTPUT.PUT_LINE ('num_small ='||num_small);

DBMS_OUTPUT.PUT_LINE ('num_large ='||num_large);

END;

/

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is active. The main area contains a code editor with the following PL/SQL script:

```

8  num_small := num_large;
9  num_large := num_temp;
10 END IF;
11 DBMS_OUTPUT.PUT_LINE ('num_small = '||num_small);
12 DBMS_OUTPUT.PUT_LINE ('num_large = '||num_large);
13 END;
14 /
15

```

Below the code editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, displaying the output of the executed script:

```

num_small = 5
num_large = 8
Statement processed.

0.00 seconds

```

7.) Write a PL/SQL procedure to calculate the incentive on a target achieved and display the message either the record updated or not.

QUERY:

```

DECLARE
  PROCEDURE test1 (
    sal_achieve NUMBER,
    target_qty NUMBER,
    emp_id NUMBER
  )
  IS
    incentive NUMBER := 0;
    updated VARCHAR2(3) := 'No';
  BEGIN
    IF sal_achieve > (target_qty + 200) THEN
      incentive := (sal_achieve - target_qty)/4;
      UPDATE employees
        SET salary = salary + incentive
        WHERE employee_id = emp_id;
      updated := 'Yes';
    END IF;
    DBMS_OUTPUT.PUT_LINE (
      'Table updated? ' || updated || ',' ||
      'incentive = ' || incentive || !
    );
  END test1;
BEGIN
  test1(2300, 2000, 144);
  test1(3600, 3000, 145);
END;

```

/

OUTPUT:



8.) Write a PL/SQL procedure to calculate incentive achieved according to the specific sale limit

QUERY:

```
DECLARE
  PROCEDURE test1 (sal_achieve NUMBER)
  IS
    incentive NUMBER := 0;
  BEGIN
    IF sal_achieve > 44000 THEN
      incentive := 1800;
    ELSIF sal_achieve > 32000 THEN
      incentive := 800;
    ELSE
      incentive := 500;
    END IF;
    DBMS_OUTPUT.NEW_LINE;
    DBMS_OUTPUT.PUT_LINE (
      'Sale achieved : ' || sal_achieve || ', incentive : ' || incentive || '!');
  END test1;
BEGIN
  test1(45000);
  test1(36000);
  test1(28000);
END;
/
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. The code entered is:

```
13  DBMS_OUTPUT.NEW_LINE;
14  DBMS_OUTPUT.PUT_LINE (
15  'Sale achieved : ' || sal_achieve || ', incentive : ' || incentive || '.' );
16  END test1;
17  BEGIN
18    test1(45000);
19    test1(36000);
20    test1(28000);
21  END;
22 /
23 |
```

In the 'Results' tab, the output is displayed:

```
Sale achieved : 45000, incentive : 1800.
Sale achieved : 36000, incentive : 800.
Sale achieved : 28000, incentive : 500.
Statement processed.
```

9.) Write a PL/SQL program to count number of employees in department 50 and check whether this department have any vacancies or not. There are 45 vacancies in this department.

QUERY:

```
SET SERVEROUTPUT ON
DECLARE
  tot_emp NUMBER;
  get_dep_id NUMBER;

BEGIN
  get_dep_id := 80;
  SELECT Count(*)
  INTO tot_emp
  FROM employees e
  join departments d
  ON e.department_id = d.department_id
  WHERE e.department_id = get_dep_id;
  dbms_output.Put_line ('The employees are in the department ''||get_dep_id||'' is: '
  ||To_char(tot_emp));
  IF tot_emp >= 45 THEN
    dbms_output.Put_line ('There are no vacancies in the department ''||get_dep_id||''');
  ELSE
    dbms_output.Put_line ('There are ''||to_char(45-tot_emp)||'' vacancies in department ''||
    get_dep_id ''');
  END IF;
END;
```

/

OUTPUT:



10.) Write a PL/SQL program to count number of employees in a specific department and check whether this department have any vacancies or not. If any vacancies, how many vacancies are in that department.

QUERY:

DECLARE

```
tot_emp NUMBER;  
get_dep_id NUMBER;
```

BEGIN

```
get_dep_id := 80;  
SELECT Count(*)  
INTO tot_emp  
FROM employees e  
join departments d  
ON e.department_id = d.dept_id  
WHERE e.department_id = get_dep_id;
```

```
dbms_output.Put_line ('The employees are in the department ''||get_dep_id||' is: '  
||To_char(tot_emp));
```

IF tot_emp >= 45 THEN

```
dbms_output.Put_line ('There are no vacancies in the department ''||get_dep_id);  
ELSE
```

```
dbms_output.Put_line ('There are ''||to_char(45-tot_emp)||'' vacancies in department ''||  
get_dep_id );
```

```
END IF;  
END;  
/  
/
```

OUTPUT:



11.) Write a PL/SQL program to display the employee IDs, names, job titles, hire dates, and salaries of all employees

QUERY:

```
DECLARE  
v_employee_id employees.employee_id%TYPE;  
v_full_name employees.first_name%TYPE;  
v_job_id employees.job_id%TYPE;  
v_hire_date employees.hire_date%TYPE;  
v_salary employees.salary%TYPE;  
CURSOR c_employees IS  
SELECT employee_id, first_name || ' ' || last_name AS full_name, job_id, hire_date, salary  
FROM employees;  
BEGIN  
DBMS_OUTPUT.PUT_LINE('Employee ID | Full Name | Job Title | Hire Date | Salary');  
DBMS_OUTPUT.PUT_LINE('-----');  
OPEN c_employees;  
FETCH c_employees INTO v_employee_id, v_full_name, v_job_id, v_hire_date, v_salary;  
WHILE c_employees%FOUND LOOP  
DBMS_OUTPUT.PUT_LINE(v_employee_id || ' ' || v_full_name || ' ' || v_job_id || ' ' ||  
v_hire_date || ' ' || v_salary);  
FETCH c_employees INTO v_employee_id, v_full_name, v_job_id, v_hire_date,  
v_salary;  
END LOOP;  
CLOSE c_employees;
```

```
END;
```

```
/
```

OUTPUT:



12.) Write a PL/SQL program to display the employee IDs, names, and department names of all employees.

QUERY:

```
DECLARE
  CURSOR emp_cursor IS
    SELECT e.employee_id, e.first_name, m.first_name AS manager_name
    FROM employees e
    LEFT JOIN employees m ON e.manager_id = m.employee_id;
  emp_record emp_cursor%ROWTYPE;
BEGIN
  OPEN emp_cursor;
  FETCH emp_cursor INTO emp_record;
  WHILE emp_cursor%FOUND LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp_record.employee_id);
    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || emp_record.first_name);
    DBMS_OUTPUT.PUT_LINE('Manager Name: ' || emp_record.manager_name);
    DBMS_OUTPUT.PUT_LINE('-----');
    FETCH emp_cursor INTO emp_record;
  END LOOP;
  CLOSE emp_cursor;
END;
/
```

OUTPUT:



14.) Write a PL/SQL program to display the employee IDs, names, and job history start dates of all employees.

QUERY:

```
DECLARE
  CURSOR employees_cur IS
    SELECT employee_id, last_name, job_id, start_date
    FROM employees NATURAL JOIN job_history;
    emp_start_date DATE;
BEGIN
  dbms_output.Put_line(Rpad('Employee ID', 15) || Rpad('Last Name', 25) || Rpad('Job Id', 35)
    || 'Start Date');
  dbms_output.Put_line('-----'
    -----);
  FOR emp_sal_rec IN employees_cur LOOP
    -- find out most recent end_date in job_history
    SELECT Max(end_date) + 1
    INTO emp_start_date
    FROM job_history
    WHERE employee_id = emp_sal_rec.employee_id;
    IF emp_start_date IS NULL THEN
      emp_start_date := emp_sal_rec.start_date;
    END IF;
    dbms_output.Put_line(Rpad(emp_sal_rec.employee_id, 15)
      || Rpad(emp_sal_rec.last_name, 25)
      || Rpad(emp_sal_rec.job_id, 35)
      || To_char(emp_start_date, 'dd-mon-yyyy'));
  END LOOP;
END;
```

/OUTPUT:



EXNO:17 PROCEDURES AND FUNCTIONS DATE: 1.)Factorial of a number using function.

QUERY:

```
DECLARE
  fac NUMBER := 1;
  n NUMBER := :1;
BEGIN
  WHILE n > 0 LOOP
    fac := n * fac;
    n := n - 1;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE(fac);
END;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'aarthi s aarthi001', and a 'Run' button. The main workspace displays the PL/SQL code for calculating a factorial. Below the code, the 'Results' tab is selected, showing the output: '6' and 'Statement processed.' The bottom status bar indicates '0.01 seconds'.

```
1  DECLARE
2    fac NUMBER := 1;
3    n NUMBER := :1;
4  BEGIN
5    WHILE n > 0 LOOP
6      fac := n * fac;
7      n := n - 1;
8    END LOOP;
9    DBMS_OUTPUT.PUT_LINE(fac);
10   END;
11  
```

2.) Write a PL/SQL program using Procedures IN,INOUT,OUT parameters to retrieve the corresponding book information in library.

QUERY:

```
CREATE OR REPLACE PROCEDURE get_book_info (
    p_book_id IN NUMBER,
    p_title IN OUT VARCHAR2,
    p_author OUT VARCHAR2,
    p_year_published OUT NUMBER)
AS
BEGIN
    SELECT title, author, year_published INTO p_title, p_author, p_year_published
    FROM books
    WHERE book_id = p_book_id;
    p_title := p_title || ' - Retrieved';
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        p_title := NULL;
        p_author := NULL;
        p_year_published := NULL;
END;
DECLARE
    v_book_id NUMBER := 1;
    v_title VARCHAR2(100);
    v_author VARCHAR2(100);
    v_year_published NUMBER;
BEGIN
    v_title := 'Initial Title';
    get_book_info(p_book_id => v_book_id, p_title => v_title, p_author =>
    v_author, p_year_published => v_year_published);
    DBMS_OUTPUT.PUT_LINE('Title: ' || v_title);
    DBMS_OUTPUT.PUT_LINE('Author: ' || v_author);
    DBMS_OUTPUT.PUT_LINE('Year Published: ' || v_year_published); END;
OUTPUT:
```

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected, followed by 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The search bar contains 'Search'. On the right, it says 'AS aarthi s aarthi001'. Below the navigation, there's a toolbar with icons for Undo, Redo, Search, Find Tables, Save, and Run. The main area is titled 'SQL Commands' with a sub-section 'Language SQL'. It shows the following PL/SQL code:

```

1 CREATE OR REPLACE PROCEDURE get_book_info (
2   p_book_id IN NUMBER,
3   p_title IN OUT VARCHAR2,
4   p_author OUT VARCHAR2,
5   p_year_published OUT NUMBER)
6 AS
7 BEGIN
8   SELECT title, author, year_published INTO p_title, p_author, p_year_published FROM books
9   WHERE book_id = p_book_id;
10  p_title := p_title || ' - Retrieved';
11 EXCEPTION
12 WHEN NO_DATA_FOUND THEN

```

The code is highlighted in blue and red. A yellow callout box highlights the error at line 17: 'Error at line 17: PLS-00103: Encountered the symbol "DECLARE"'. The code within the error box is:

```

1. CREATE OR REPLACE PROCEDURE get_book_info (
2.   p_book_id IN NUMBER,
3.   p_title IN OUT VARCHAR2,

```

EXNO:18 TRIGGER DATE:

1.) Write a code in PL/SQL to develop a trigger that enforces referential integrity by preventing the deletion of a parent record if child records exist

QUERY:

```

CREATE OR REPLACE TRIGGER prevent_parent_deletion
BEFORE DELETE ON parent_table
FOR EACH ROW
DECLARE
  child_exists EXCEPTION;
  PRAGMA EXCEPTION_INIT(child_exists, -20001);
  v_child_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_child_count FROM child_table WHERE parent_id =
  :OLD.parent_id;
  IF v_child_count > 0 THEN
    RAISE child_exists;
  END IF;
EXCEPTION
  WHEN child_exists THEN
    RAISE_APPLICATION_ERROR(-20001, 'Cannot delete parent record while child records
exist.');
END;

```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the code editor, there is a PL/SQL trigger definition:

```

7 | v_child_count NUMBER;
8 | BEGIN
9 |   SELECT COUNT(*) INTO v_child_count FROM child_table WHERE parent_id = :OLD.parent_id;
10|   IF v_child_count > 0 THEN
11|     RAISE child_exists;
12|   END IF;
13| EXCEPTION
14|   WHEN child_exists THEN
15|     RAISE_APPLICATION_ERROR(-20001, 'Cannot delete parent record while child records exist.');
16| END;
17|

```

The code fails at line 2/18 due to the error: ORA-00942: table or view does not exist. A yellow box highlights the error message and the following four steps for resolution:

1. CREATE OR REPLACE TRIGGER prevent_parent_deletion
2. BEFORE DELETE ON parent_table
3. FOR EACH ROW
4. DECLARE

2.) Write a code in PL/SQL to create a trigger that checks for duplicate values in a specific column and raises an exception if found

QUERY:

```

CREATE OR REPLACE TRIGGER check_duplicates
BEFORE INSERT OR UPDATE ON unique_values_table
FOR EACH ROW
DECLARE
  duplicate_found EXCEPTION;
  PRAGMA EXCEPTION_INIT(duplicate_found, -20002);
  v_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_count FROM unique_values_table
  WHERE unique_col = :NEW.unique_col AND id != :NEW.id;
  IF v_count > 0 THEN
    RAISE duplicate_found;
  END IF;
EXCEPTION
  WHEN duplicate_found THEN
    RAISE_APPLICATION_ERROR(-20002, 'Duplicate value found in unique_col.');
END;

```

OUTPUT:



3.) Write a code in PL/SQL to create a trigger that restricts the insertion of new rows if the total of a column's values exceeds a certain threshold

QUERY:

```
CREATE OR REPLACE TRIGGER check_threshold
BEFORE INSERT OR UPDATE ON threshold_table
FOR EACH ROW
DECLARE
    threshold_exceeded EXCEPTION;
    PRAGMA EXCEPTION_INIT(threshold_exceeded, -20003);
    v_sum NUMBER;
    v_threshold NUMBER := 10000; -- Set your threshold here
BEGIN
    SELECT SUM(value_col) INTO v_sum FROM threshold_table;
    v_sum := v_sum + :NEW.value_col;
    IF v_sum > v_threshold THEN
        RAISE threshold_exceeded;
    END IF;
EXCEPTION
    WHEN threshold_exceeded THEN
        RAISE_APPLICATION_ERROR(-20003, 'Threshold exceeded for value_col.');
END;
```

OUTPUT:



4.) Write a code in PL/SQL to design a trigger that captures changes made to specific columns and logs them in an audit table.

QUERY:

```
CREATE OR REPLACE TRIGGER log_changes
AFTER UPDATE ON main_table
FOR EACH ROW
BEGIN
  INSERT INTO audit_table (audit_id, changed_id, old_col1, new_col1,
old_col2, new_col2, change_time)
  VALUES (audit_seq.NEXTVAL, :OLD.id, :OLD.col1, :NEW.col1, :OLD.col2,
:NEW.col2, SYSTIMESTAMP);
END;
```

OUTPUT:



5.) Write a code in PL/SQL to implement a trigger that records user activity (inserts, updates, deletes) in an audit log for a given set of tables.

QUERY:

```
CREATE OR REPLACE TRIGGER log_user_activity
AFTER INSERT OR UPDATE OR DELETE ON activity_table
FOR EACH ROW
BEGIN
  IF INSERTING THEN
    INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time) VALUES
    (activity_log_seq.NEXTVAL, 'INSERT', 'activity_table', :NEW.id, SYSTIMESTAMP); ELSIF
  UPDATING THEN
    INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time) VALUES
    (activity_log_seq.NEXTVAL, 'UPDATE', 'activity_table', :NEW.id, SYSTIMESTAMP); ELSIF
  DELETING THEN
    INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time) VALUES
    (activity_log_seq.NEXTVAL, 'DELETE', 'activity_table', :OLD.id, SYSTIMESTAMP); END IF;
END;
```

OUTPUT:



6.) Write a code in PL/SQL to implement a trigger that automatically calculates and updates a running total column for a table whenever new rows are inserted

QUERY:

```
CREATE OR REPLACE TRIGGER update_running_total
BEFORE INSERT ON running_total_table
FOR EACH ROW
DECLARE
    v_total NUMBER;
BEGIN
    SELECT NVL(SUM(amount), 0) INTO v_total FROM running_total_table;
    :NEW.running_total := v_total + :NEW.amount;
END;
```

OUTPUT:

MONGO DB

EX_NO: 19 DATE:

1.)Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'.

QUERY:

```
db.restaurants.find( { $or: [{ name: /^Wil/ }, { cuisine: { $nin: ['American', 'Chinese'] } } ] , { restaurant_id: 1, name: 1, borough: 1, cuisine: 1 } );
```

OUTPUT:

The screenshot shows a MongoDB shell interface. On the left, there is a search bar labeled "Enter a title...". Below it, a dropdown menu says "MongoDB" with a dropdown arrow, and a small info icon. To the right are two buttons: "Run" (green) and "Save" (blue). In the main area, a command is typed: `i db.movies.find({ year: 1893 })`. To the right, under the heading "Output", the response is shown: `mycompiler_mongodb>`, followed by two blank lines, and then `[Execution complete with exit code 0]`.

2.) Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates.

QUERY:

```
db.restaurants.find( { grades: { $elemMatch: { grade: "A", score: 11, date: ISODate("2014-08-11T00:00:00Z") } } }, { restaurant_id: 1, name: 1, grades: 1 } );
```

OUTPUT:

The screenshot shows a MongoDB shell interface. At the top, there is a search bar labeled "Enter a title...". Below it, a toolbar has a "MongoDB" dropdown, a help icon, and two buttons: "Run" and "Save". In the main area, a command is entered: `i db.movies.find({ runtime: { $gt: 120 } })`. To the right, under the "Output" section, the response is shown: `mycompiler_mongodb>`, followed by two blank lines, and then "[Execution complete with exit code 0]".

3.) Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".

QUERY:

```
db.restaurants.find( { "grades.1.grade": "A", "grades.1.score": 9, "grades.1.date": ISODate("2014-08-11T00:00:00Z") }, { restaurant_id: 1, name: 1, grades: 1 } );
```

OUTPUT:

The screenshot shows a MongoDB shell interface. At the top, there is a search bar labeled "Enter a title...". Below it, a toolbar has a "MongoDB" dropdown, a help icon, and two buttons: "Run" and "Save". In the main area, a command is entered: `i db.movies.find({ genres: 'short' })`. To the right, under the "Output" section, the response is shown: `mycompiler_mongodb>`, followed by two blank lines, and then "[Execution complete with exit code 0]".

4.) Write a MongoDB query to find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and upto 52

QUERY:

```
db.restaurants.find({$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, {_id:0, restaurant_id:1, name:1, address:1})
```

OUTPUT:

The screenshot shows a MongoDB shell interface. On the left, there is an input field with placeholder text "Enter a title...". Below it, a dropdown menu is set to "MongoDB" and a "Run" button is visible. The main area contains a command line with the text "db.movies.find({ directors: 'William K.L. Dickson' })". To the right, a "Output" panel displays the results of the query execution. The output shows the prompt "mycompiler_mongodb>" followed by "[Execution complete with exit code 0]".

- 5.) Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns.

QUERY:

```
db.restaurants.find({}, { _id: 0 }).sort({ name: 1 });
```

Enter a title...

MongoDB ▾



Ctrl+S

▶ Run

Save

```
1 db.movies.find({ countries: 'USA' })
```

Output

```
mycompiler_mongodb>
mycompiler_mongodb>
```

```
[Execution complete with exit code 0]
```