

EXERCISE-6

Single Row Functions

Name: Vedhasree S

Register Number: 240701580

Department: CSE

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

The screenshot shows the SQL Developer interface. The SQL Commands window contains the following query:

```
1 SELECT SYSDATE AS "Date"
2 FROM dual;
```

The Results window shows a single row with the date 10/12/2025 under the column header Date.

Date
10/12/2025

1 rows returned in 0.01 seconds

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.

The screenshot shows the SQL Developer interface. The SQL Commands window contains the following query:

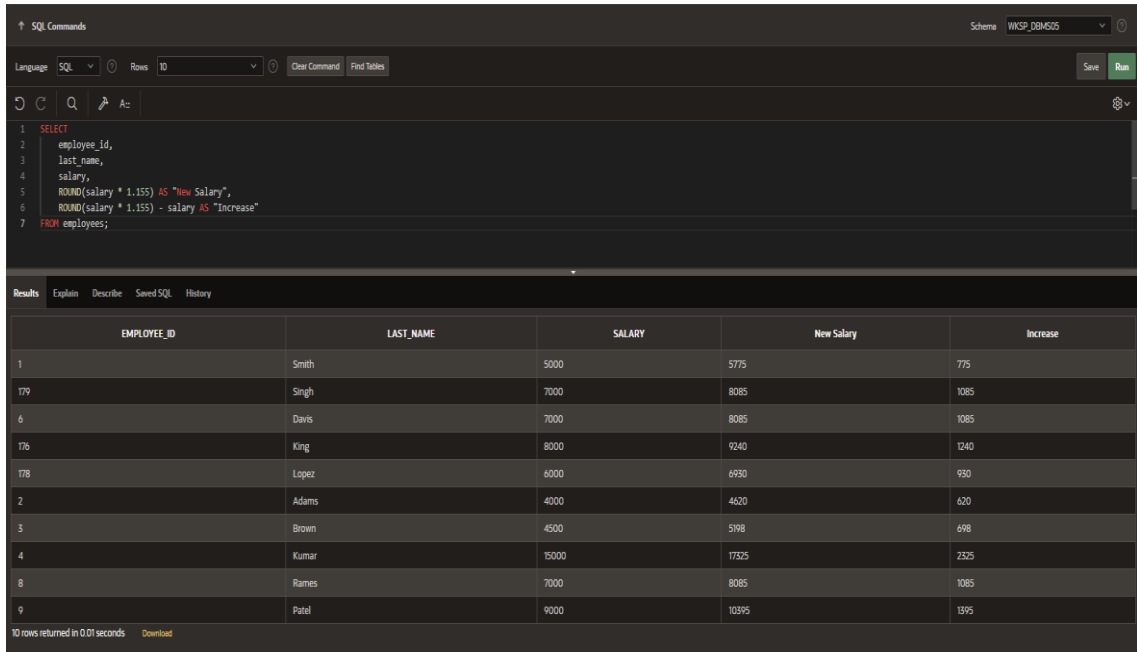
```
1 SELECT
2   employee_id,
3   last_name,
4   salary,
5   ROUND(salary * 1.155) AS "New Salary"
6 FROM employees;
```

The Results window shows a table with 10 rows of employee data. The columns are EMPLOYEE_ID, LAST_NAME, SALARY, and New Salary.

EMPLOYEE_ID	LAST_NAME	SALARY	New Salary
1	Smith	5000	5775
779	Singh	7000	8085
6	Davis	7000	8085
776	King	8000	9240
778	Lopez	6000	6930
2	Adams	4000	4620
3	Brown	4500	5198
4	Kumar	15000	17325
8	Rames	7000	8085
9	Patel	9000	10395

10 rows returned in 0.02 seconds

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.



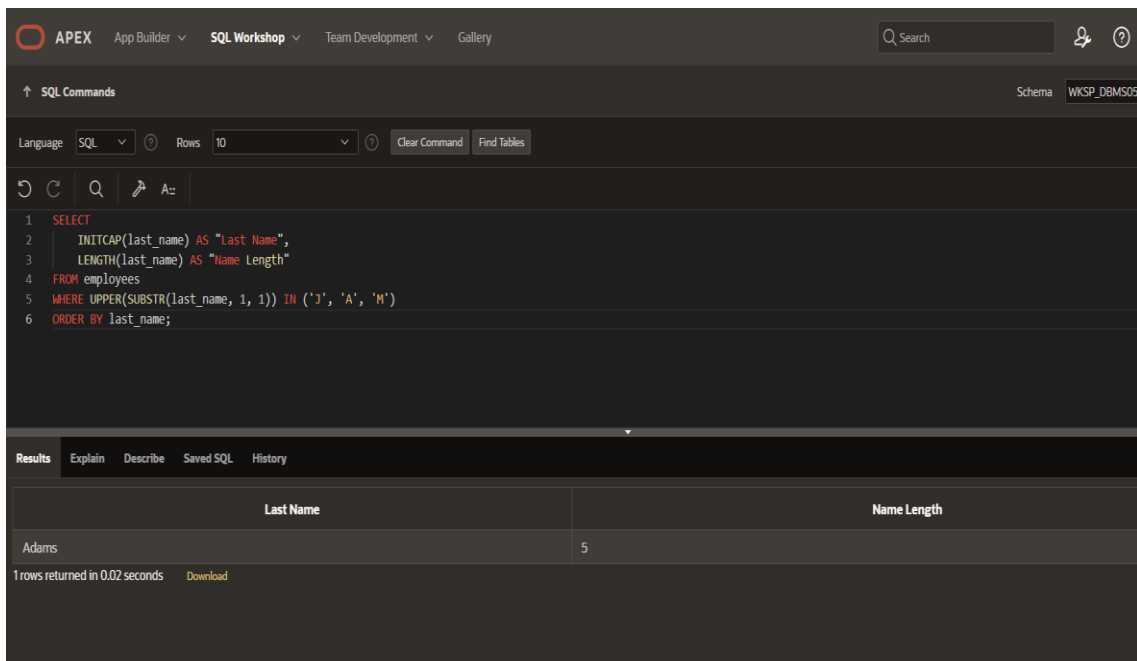
The screenshot shows the SQL Developer interface with a query window and a results grid. The query calculates a new salary (1.155 times the old salary) and the increase (the difference between the new and old salaries).

```
1 SELECT
2   employee_id,
3   last_name,
4   salary,
5   ROUND(salary * 1.155) AS "New Salary",
6   ROUND(salary * 1.155) - salary AS "Increase"
7 FROM employees;
```

EMPLOYEE_ID	LAST_NAME	SALARY	New Salary	Increase
1	Smith	5000	5775	775
179	Singh	7000	8085	1085
6	Davis	7000	8085	1085
176	King	8000	9240	1240
178	Lopez	6000	6930	930
2	Adams	4000	4620	620
3	Brown	4500	5198	698
4	Kumar	15000	17325	2325
8	Rames	7000	8085	1085
9	Patel	9000	10395	1395

10 rows returned in 0.01 seconds Download

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 whose name starts with the letters J, A, or M. Give each column an appropriate label. Sort the results by the employees' last names.



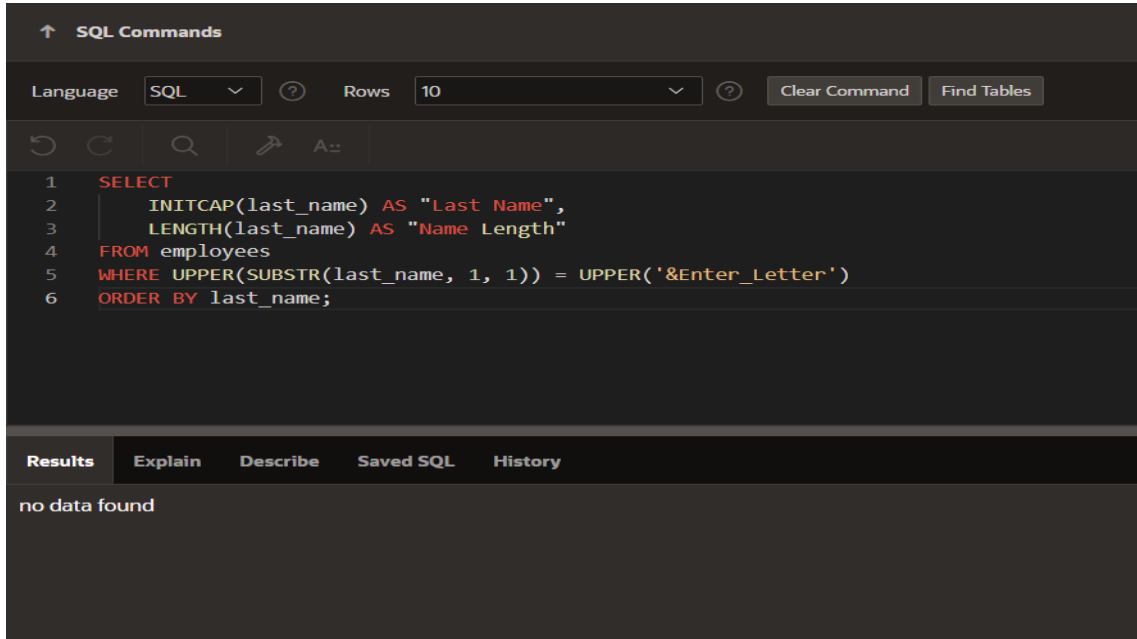
The screenshot shows the APEX SQL Workshop interface with a query window and a results grid. The query uses INITCAP to format the last name and LENGTH to get the name length, filtering for names starting with J, A, or M.

```
1 SELECT
2   INITCAP(last_name) AS "Last Name",
3   LENGTH(last_name) AS "Name Length"
4 FROM employees
5 WHERE UPPER(SUBSTR(last_name, 1, 1)) IN ('J', 'A', 'M')
6 ORDER BY last_name;
```

Last Name	Name Length
Adams	5

1 rows returned in 0.02 seconds Download

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 whose last name starts with the letter H.

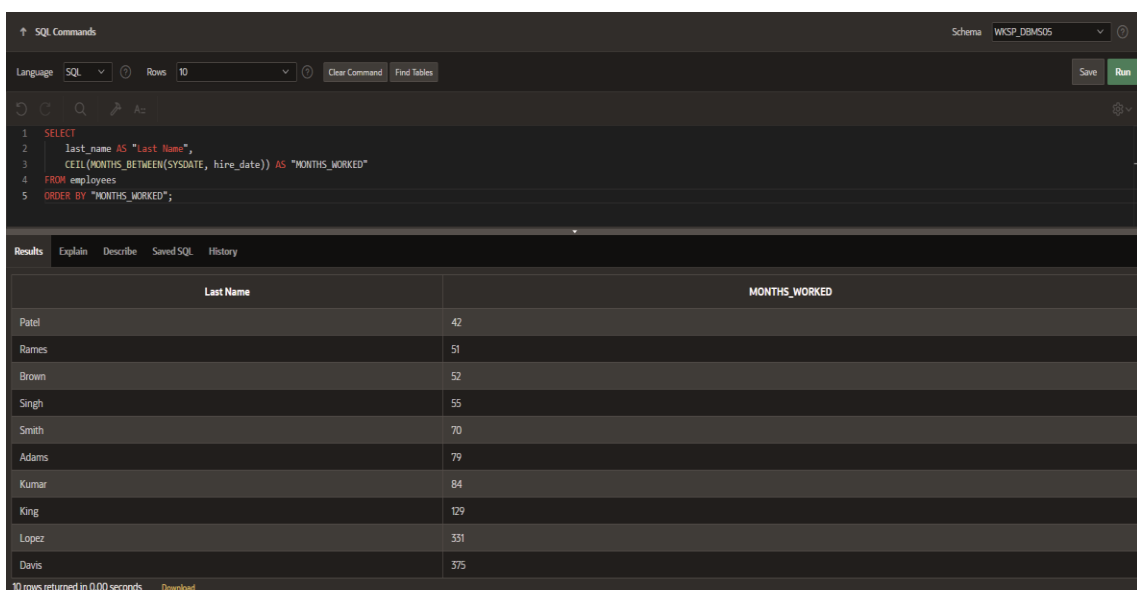


The screenshot shows a SQL interface with a query editor and a results pane. The query is as follows:

```
1 SELECT
2     INITCAP(last_name) AS "Last Name",
3     LENGTH(last_name) AS "Name Length"
4 FROM employees
5 WHERE UPPER(SUBSTR(last_name, 1, 1)) = UPPER('&Enter_Letter')
6 ORDER BY last_name;
```

The results pane shows "no data found".

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.



The screenshot shows a SQL interface with a query editor and a results pane. The query is as follows:

```
1 SELECT
2     last_name AS "Last Name",
3     CEIL(MONTHS_BETWEEN(SYSDATE, hire_date)) AS "MONTHS_WORKED"
4 FROM employees
5 ORDER BY "MONTHS_WORKED";
```

The results pane shows a table with 10 rows:

Last Name	MONTHS_WORKED
Patel	42
Rames	51
Brown	52
Singh	55
Smith	70
Adams	79
Kumar	84
King	129
Lopez	331
Davis	375

10 rows returned in 0.00 seconds

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.

The screenshot shows the APEX SQL Workshop interface. The SQL command is:

```
1 SELECT
2   last_name || ' earns ' || salary || ' monthly but wants ' || (salary*3) AS "Dream Salaries"
3 FROM employees;
```

The results are displayed in a table with the column header "Dream Salaries".

Dream Salaries
Smith earns 5000 monthly but wants 15000
Singh earns 7000 monthly but wants 21000
Davis earns 7000 monthly but wants 21000
King earns 8000 monthly but wants 24000
Lopez earns 6000 monthly but wants 18000
Adams earns 4000 monthly but wants 12000
Brown earns 4500 monthly but wants 13500
Kumar earns 15000 monthly but wants 45000
Rames earns 7000 monthly but wants 21000
Patel earns 9000 monthly but wants 27000

10 rows returned in 0.01 seconds

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

The screenshot shows the APEX SQL Workshop interface. The SQL command is:

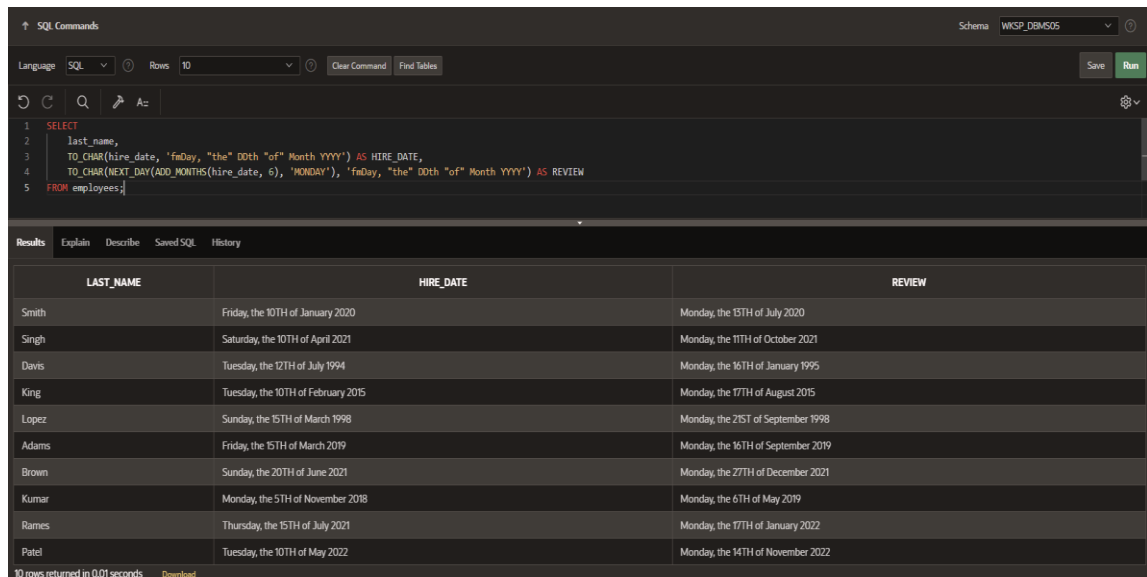
```
1 SELECT
2   last_name,
3   LPAD(TO_CHAR(salary, '$999,999,999.00'), 15) AS SALARY
4 FROM employees;
```

The results are displayed in a table with columns "LAST_NAME" and "SALARY".

LAST_NAME	SALARY
Smith	\$5,000.0
Singh	\$7,000.0
Davis	\$7,000.0
King	\$8,000.0
Lopez	\$6,000.0
Adams	\$4,000.0
Brown	\$4,500.0
Kumar	\$15,000.0
Rames	\$7,000.0
Patel	\$9,000.0

10 rows returned in 0.01 seconds

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."



The screenshot shows the SQL Developer interface with the following SQL query:

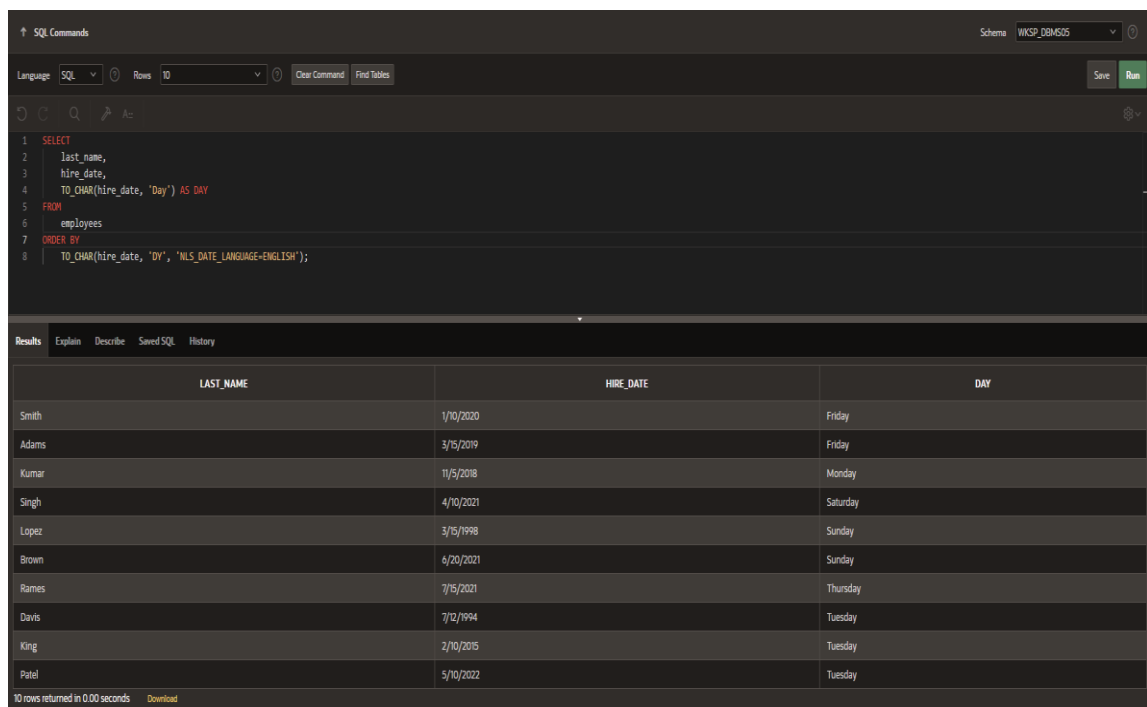
```
1 SELECT
2   last_name,
3   TO_CHAR(hire_date, 'fmDay, "the" DDth "of" Month YYYY') AS HIRE_DATE,
4   TO_CHAR(NEXT_DAY(ADD_MONTHS(hire_date, 6), 'MONDAY'), 'fmDay, "the" DDth "of" Month YYYY') AS REVIEW
5 FROM employees;
```

The results table displays the following data:

LAST_NAME	HIRE_DATE	REVIEW
Smith	Friday, the 10TH of January 2020	Monday, the 13TH of July 2020
Singh	Saturday, the 10TH of April 2021	Monday, the 11TH of October 2021
Davis	Tuesday, the 12TH of July 1994	Monday, the 16TH of January 1995
King	Tuesday, the 10TH of February 2015	Monday, the 17TH of August 2015
Lopez	Sunday, the 15TH of March 1998	Monday, the 21ST of September 1998
Adams	Friday, the 15TH of March 2019	Monday, the 16TH of September 2019
Brown	Sunday, the 20TH of June 2021	Monday, the 27TH of December 2021
Kumar	Monday, the 5TH of November 2018	Monday, the 6TH of May 2019
Rames	Thursday, the 15TH of July 2021	Monday, the 17TH of January 2022
Patel	Tuesday, the 10TH of May 2022	Monday, the 14TH of November 2022

10 rows returned in 0.01 seconds

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.



The screenshot shows the SQL Developer interface with the following SQL query:

```
1 SELECT
2   last_name,
3   hire_date,
4   TO_CHAR(hire_date, 'Day') AS DAY
5 FROM
6   employees
7 ORDER BY
8   TO_CHAR(hire_date, 'DY', 'NLS_DATE_LANGUAGE=ENGLISH');
```

The results table displays the following data:

LAST_NAME	HIRE_DATE	DAY
Smith	1/10/2020	Friday
Adams	3/15/2019	Friday
Kumar	11/5/2018	Monday
Singh	4/10/2021	Saturday
Lopez	3/15/1998	Sunday
Brown	6/20/2021	Sunday
Rames	7/15/2021	Thursday
Davis	7/12/1994	Tuesday
King	2/10/2015	Tuesday
Patel	5/10/2022	Tuesday

10 rows returned in 0.00 seconds