

NAME-HARSHIT SELARKA
PRN-23070521131
SEC-B
BATCH B-2

Practical 3. Part 1.

Write and execute SQL functions- aggregate, numeric, date, string, and conversion.

SQL Aggregate functions

SQL Aggregate Functions are used to perform calculations on a set of rows and return a single value. They are often used with the **GROUP BY** clause in SQL to summarize data for each group. Commonly used aggregate functions include **COUNT()**, **SUM()**, **AVG()**, **MIN()**, and **MAX()**.

Common SQL Aggregate Functions

Count()

- **Count(*)**: Returns the total number of records.
- **Count(salary)**: Return the number of Non-Null values over the column salary.
- **Count(Distinct Salary)**: Return the number of distinct Non-Null values over the column salary.

Sum()

- **sum(salary)**: Sum all Non-Null values of Column salary for example 3120.
- **sum(Distinct salary)**: Sum of all distinct Non-Null values for example 3120.

Avg()

- **Avg(salary)** = $\text{Sum}(\text{salary}) / \text{count}(\text{salary}) = 3120 / 5 = 624$
- **Avg(Distinct salary)** = $\text{sum}(\text{Distinct salary}) / \text{Count}(\text{Distinct Salary}) = 3120 / 5 = 624$

Min()

- **Min(salary)**: Minimum value in the salary column except NULL i.e., 403.

Max():

- **Max(salary)**: Maximum value in the salary i.e., 802.

SQL Aggregate Functions with Examples (Oracle SQL Plus)

Create the Employee table

```
CREATE TABLE Employee (  
  Id NUMBER PRIMARY KEY,  
  Name CHAR(1),
```

Salary NUMBER(10, 2)

The screenshot shows the Online Oracle DB Compiler interface. The SQL editor contains a script that creates an 'Employee' table with columns 'Id' (NUMBER PRIMARY KEY), 'Name' (CHAR(1)), and 'Salary' (NUMBER(10, 2)). It then inserts six rows of data. The output section shows the status 'Successfully executed', execution time of 0.0000 secs, and memory usage of 4.184 Mb. The 'Your Output' section displays the results of the queries: 6 rows inserted, total salary 624.0, average salary 802, and a list of employees with their salaries.

```
1 CREATE TABLE Employee (  
2   Id NUMBER PRIMARY KEY,  
3   Name CHAR(1),  
4   Salary NUMBER(10, 2)  
5 );  
6 INSERT INTO Employee (Id, Name, Salary)  
7 VALUES (1, 'A', 802);  
8  
9 INSERT INTO Employee (Id, Name, Salary)  
10 VALUES (2, 'B', 403);  
11  
12 INSERT INTO Employee (Id, Name, Salary)  
13 VALUES (3, 'C', 604);  
14  
15 INSERT INTO Employee (Id, Name, Salary)  
16 VALUES (4, 'D', 705);  
17  
18 INSERT INTO Employee (Id, Name, Salary)  
19 VALUES (5, 'E', 606);  
20  
21 INSERT INTO Employee (Id, Name, Salary)  
22 VALUES (6, 'F', NULL);  
23 SELECT COUNT(*) AS TotalEmployees FROM Employee;  
24 -- Calculate the total salary  
25 SELECT SUM(Salary) AS TotalSalary FROM Employee;  
26 -- Find the average salary  
27 SELECT AVG(Salary) AS AverageSalary FROM Employee;  
28 -- Get the highest salary  
29 SELECT MAX(Salary) AS HighestSalary FROM Employee;  
30 -- Determine the lowest salary  
31 SELECT MIN(Salary) AS LowestSalary FROM Employee;  
32 SELECT Name, SUM(Salary) AS TotalSalary  
33 FROM Employee  
34 GROUP BY Name;  
35 SELECT Name, SUM(Salary) AS TotalSalary  
36 FROM Employee  
37 GROUP BY Name  
38 HAVING SUM(Salary) > 600;  
39
```

Output

Status : Successfully executed

Time: 0.0000 secs Memory: 4.184 Mb

Your Output

```
6  
3120  
624.0  
802  
403  
A|802  
B|403  
C|604  
D|705  
E|606  
F|606
```

The screenshot shows the Online Oracle DB Compiler interface with a partial SQL script. The output section shows the status 'Successfully executed', execution time of 0.0000 secs, and memory usage of 4.184 Mb. The 'Your Output' section displays the results of the queries: 6 rows inserted, total salary 624.0, average salary 802, and a list of employees with their salaries.

```
6 INSERT INTO Employee (Id, Name, Salary)  
7 VALUES (1, 'A', 802);  
8  
9 INSERT INTO Employee (Id, Name, Salary)  
10 VALUES (2, 'B', 403);  
11  
12 INSERT INTO Employee (Id, Name, Salary)  
13 VALUES (3, 'C', 604);  
14  
15 INSERT INTO Employee (Id, Name, Salary)  
16 VALUES (4, 'D', 705);  
17  
18 INSERT INTO Employee (Id, Name, Salary)  
19 VALUES (5, 'E', 606);  
20  
21 INSERT INTO Employee (Id, Name, Salary)  
22 VALUES (6, 'F', NULL);  
23 SELECT COUNT(*) AS TotalEmployees FROM Employee;  
24 -- Calculate the total salary  
25 SELECT SUM(Salary) AS TotalSalary FROM Employee;  
26 -- Find the average salary  
27 SELECT AVG(Salary) AS AverageSalary FROM Employee;  
28 -- Get the highest salary  
29 SELECT MAX(Salary) AS HighestSalary FROM Employee;  
30 -- Determine the lowest salary  
31 SELECT MIN(Salary) AS LowestSalary FROM Employee;  
32 SELECT Name, SUM(Salary) AS TotalSalary  
33 FROM Employee  
34 GROUP BY Name;  
35 SELECT Name, SUM(Salary) AS TotalSalary  
36 FROM Employee  
37 GROUP BY Name  
38 HAVING SUM(Salary) > 600;  
39  
40  
41  
42  
43  
44  
45
```

Output

Status : Successfully executed

Time: 0.0000 secs Memory: 4.184 Mb

Your Output

```
A|802  
B|403  
C|604  
D|705  
E|606  
F|606  
A|802  
B|403  
C|604  
D|705  
E|606
```

);

-- Insert data into the Employee table

```
INSERT INTO Employee (Id, Name, Salary)  
VALUES (1, 'A', 802);
```

```
INSERT INTO Employee (Id, Name, Salary)  
VALUES (2, 'B', 403);
```

```
INSERT INTO Employee (Id, Name, Salary)
VALUES (3, 'C', 604);
```

```
INSERT INTO Employee (Id, Name, Salary)
VALUES (4, 'D', 705);
```

```
INSERT INTO Employee (Id, Name, Salary)
VALUES (5, 'E', 606);
```

```
INSERT INTO Employee (Id, Name, Salary)
VALUES (6, 'F', NULL);
```

```
-- Commit the changes to make them permanent [OPTIONAL]
COMMIT;
```

In the following example, we will use multiple aggregate functions on the data.

```
--Count the number of employees
SELECT COUNT(*) AS TotalEmployees FROM Employee;
```

```
-- Calculate the total salary
SELECT SUM(Salary) AS TotalSalary FROM Employee;
```

```
-- Find the average salary
SELECT AVG(Salary) AS AverageSalary FROM Employee;
```

```
-- Get the highest salary
SELECT MAX(Salary) AS HighestSalary FROM Employee;
```

```
-- Determine the lowest salary
SELECT MIN(Salary) AS LowestSalary FROM Employee;
```

Using Aggregate Functions with GROUP BY

GROUP BY allows you to group rows that share a property, enabling you to perform **aggregate calculations** on each group. This is commonly used with the COUNT(), SUM(), AVG(), MIN(), and MAX() functions.

Example: Total Salary by Each Employee

```
SELECT Name, SUM(Salary) AS TotalSalary
FROM Employee
GROUP BY Name;
```

Using HAVING with Aggregate Functions

The **HAVING** clause is used to filter results after applying aggregate functions. Unlike WHERE, which filters rows before aggregation, **HAVING filters groups after aggregation**.

Example: Find Employees with Salary Greater Than 600

```
SELECT Name, SUM(Salary) AS TotalSalary
FROM Employee
GROUP BY Name
HAVING SUM(Salary) > 600;
```

SQL | String functions

SQL String Functions are powerful tools that allow you to manipulate, format, and extract specific parts of text data in your database. These functions are essential for tasks like cleaning up data, comparing strings, and combining text fields. Whether you're working with names, addresses, or any form of textual data, mastering SQL string functions is crucial for efficient data handling and analysis.

String functions are used to perform an operation on input string and return an output string. Following are the string functions defined in SQL:

1. CONCAT(): Concatenate Strings

The CONCAT() function is used to concatenate (combine) two or more strings into one string. It is useful when you want to merge fields like first and last names into a full name.

Query:

```
SELECT CONCAT('John', ' ', 'Doe') AS FullName;
```

Output:

John Doe

2. CHAR_LENGTH() / CHARACTER_LENGTH(): Find String Length

The CHAR_LENGTH() or LENGTH() function returns the length of a string in characters. It's essential for validating or manipulating text data, especially when you need to know how many characters a string contains.

Query:

```
SELECT CHAR_LENGTH('Hello') AS StringLength;
```

Output:

5

3. UPPER() and LOWER(): Convert Text Case

These functions convert the text to uppercase or lowercase, respectively. They are useful for normalizing the case of text in a database.

Query:

```
SELECT UPPER('hello') AS UpperCase;
```

```
SELECT LOWER('HELLO') AS LowerCase;
```

Output:

HELLO

hello

4. LENGTH(): Length of String in Bytes

LENGTH() returns the length of a string in bytes. This can be useful for working with multi-byte character sets.

Query:

```
SELECT LENGTH('Hello') AS LengthInBytes;
```

Output:

5

5. REPLACE(): Replace Substring in String

The REPLACE() function replaces occurrences of a substring within a string with another substring. This is useful for cleaning up data, such as replacing invalid characters or formatting errors.

Query:

```
SELECT REPLACE('Hello World', 'World', 'SQL') AS UpdatedString;
```

Output:

Hello SQL

6. SUBSTRING() / SUBSTR(): Extract Part of a String

The SUBSTRING() (or SUBSTR()) function is used to extract a substring from a string, starting from a specified position. It is especially useful when you need to extract a specific part of a string, like extracting the domain from an email address.

Query:

```
SELECT SUBSTRING('Hello World', 1, 5) AS SubStringExample;
```

Output:

Hello

7. LEFT() and RIGHT(): Extract Substring from Left or Right

The LEFT() and RIGHT() functions allow you to extract a specified number of characters from the left or right side of a string, respectively.

Query:

```
SELECT LEFT('Hello World', 5) AS LeftString;
```

```
SELECT RIGHT('Hello World', 5) AS RightString;
```

Output:

Hello

World

8. INSTR(): Find Position of Substring

INSTR() finds the position of the first occurrence of a substring in a string. This is useful when you need to locate the position of specific characters within a string.

Query:

```
SELECT INSTR('Hello World', 'World') AS SubstringPosition;
```

Output:

7

9. TRIM(): Remove Leading and Trailing Spaces

TRIM() finds the position of the first occurrence of a substring in a string. This is useful when you need to locate the position of specific characters within a string.

```
TRIM([[LEADING | TRAILING | BOTH] [character] FROM string])
```

Query:

```
SELECT TRIM(' ' FROM ' Hello World ') AS TrimmedString;
```

Output:

Hello World

10. REVERSE(): Reverse the String

The REVERSE() function reverses the characters in a string. It's useful in situations where you need to process data backward, such as for password validation or certain pattern matching.

Query:

```
SELECT REVERSE('Hello') AS ReversedString;
```

Output:

olleH

codechef.com

Practical 3 part 1 study c... Assignment details Practical 3 part 1 study c... Online Oracle DB Compiler Running SQL Queries Ora...

```
37 GROUP BY Name
38 HAVING SUM(Salary) > 600;
39 -- Combine the Name of the employee with a custom string
40 SELECT Name || ' works at CodeChef' AS CombinedString FROM Employee;
41
42
43
44
45
46
47
48
49
50
```

```
A works at CodeChef
B works at CodeChef
C works at CodeChef
D works at CodeChef
E works at CodeChef
F works at CodeChef
```

Online SQL Compiler

Welcome to our online SQL compiler and interpreter, the perfect platform to run and test your SQL queries efficiently. Our tool makes coding easy for developers of any skill level, whether you're a beginner or experienced.

Our compiler will allow you to

- Run your code fast
- Get detailed output and error description after each run
- You can also check the time and memory usage of your code
- Write your code faster using the auto-complete feature
- Use and import libraries
- Customize the editor with your favorite theme

What is an online compiler?

Online compilers are online code editors that let you run and test your code in a web browser easily. Use our code SQL editor above to write your queries, and execute them.

About SQL

SQL (Structured Query Language) is a standard programming language used for managing and manipulating relational databases. It is widely used for querying, updating, and managing data in databases.

codechef.com

Practical 3 part 1 study c... Assignment details Practical 3 part 1 study c... Online Oracle DB Compiler Running SQL Queries Ora...

```
39 -- Combine the Name of the employee with a custom string
40 SELECT Name || ' works at CodeChef' AS CombinedString FROM Employee;
41 -- Get the length of each employee's Name
42 SELECT Name, LENGTH(Name) AS StringLength FROM Employee;
43
44
45
46
47
48
49
50
51
52
```

```
A|1
B|1
C|1
D|1
E|1
F|1
```

Online SQL Compiler

Welcome to our online SQL compiler and interpreter, the perfect platform to run and test your SQL queries efficiently. Our tool makes coding easy for developers of any skill level, whether you're a beginner or experienced.

Our compiler will allow you to

- Run your code fast
- Get detailed output and error description after each run
- You can also check the time and memory usage of your code
- Write your code faster using the auto-complete feature
- Use and import libraries
- Customize the editor with your favorite theme

What is an online compiler?

Online compilers are online code editors that let you run and test your code in a web browser easily. Use our code SQL editor above to write your queries, and execute them.

About SQL

SQL (Structured Query Language) is a standard programming language used for managing and manipulating relational databases. It is widely used for querying,

codechef.com

(no subject) - harshitselar... Practical 3 part 1 study c... Assignment details Practical 3 part 1 study c... Online Oracle DB Compiler Running SQL Queries Ora...

```
41 -- Get the length of each employee's Name
42 SELECT Name, LENGTH(Name) AS StringLength FROM Employee;
43 -- Convert each employee's Name to upper and lower case
44 SELECT Name, UPPER(Name) AS UpperCase, LOWER(Name) AS LowerCase FROM Employee;
45
46
47
48
49
50
51
52
53
54
```

A|A|a
B|B|b
C|C|c
D|D|d
E|E|e
F|F|f

Online SQL Compiler

Welcome to our online SQL compiler and interpreter, the perfect platform to run and test your SQL queries efficiently. Our tool makes coding easy for developers of any skill level, whether you're a beginner or experienced.

Our compiler will allow you to

- Run your code fast
- Get detailed output and error description after each run
- You can also check the time and memory usage of your code
- Write your code faster using the auto-complete feature
- Use and import libraries
- Customize the editor with your favorite theme

What is an online compiler?

Online compilers are online code editors that let you run and test your code in a web browser easily. Use our code SQL editor above to write your queries, and execute them.

About SQL

SQL (Structured Query Language) is a standard programming language used for managing and manipulating relational databases. It is widely used for querying, updating, and managing data in databases.

codechef.com

(no subject) - harshitselar... Practical 3 part 1 study c... Assignment details Practical 3 part 1 study c... Online Oracle DB Compiler Running SQL Queries Ora...

```
42 SELECT Name, LENGTH(Name) AS StringLength FROM Employee;
43 -- Convert each employee's Name to upper and lower case
44 SELECT Name, UPPER(Name) AS UpperCase, LOWER(Name) AS LowerCase FROM Employee;
45 -- Replace a specific character in the employee's Name
46 SELECT Name, REPLACE(Name, 'A', 'Z') AS UpdatedName FROM Employee;
47
48
49
50
51
52
53
54
55
```

A|Z
B|B
C|C
D|D
E|E
F|F

Online SQL Compiler

Welcome to our online SQL compiler and interpreter, the perfect platform to run and test your SQL queries efficiently. Our tool makes coding easy for developers of any skill level, whether you're a beginner or experienced.

Our compiler will allow you to

- Run your code fast
- Get detailed output and error description after each run
- You can also check the time and memory usage of your code
- Write your code faster using the auto-complete feature
- Use and import libraries
- Customize the editor with your favorite theme

What is an online compiler?

Online compilers are online code editors that let you run and test your code in a web browser easily. Use our code SQL editor above to write your queries, and execute them.

About SQL

SQL (Structured Query Language) is a standard programming language used for managing and manipulating relational databases. It is widely used for querying,

codechef.com

(no subject) - harshitselar...

Practical 3 part 1 study c...

Assignment details

Practical 3 part 1 study c...

Online Oracle DB Compiler

Running SQL Queries Ora...

```
42 SELECT Name, LENGTH(Name) AS StringLength FROM Employee;
43 -- Convert each employee's Name to upper and lower case
44 SELECT Name, UPPER(Name) AS UpperCase, LOWER(Name) AS LowerCase FROM Employee;
45 -- Replace a specific character in the employee's Name
46 SELECT Name, REPLACE(Name, 'A', 'Z') AS UpdatedName FROM Employee;
47 -- Extract the first character of each employee's Name
48 SELECT Name, SUBSTR(Name, 1, 1) AS FirstCharacter FROM Employee;
49
50
51
52
53
54
55
56
```

A|A
B|B
C|C
D|D
E|E
F|F

Online SQL Compiler

Welcome to our online SQL compiler and interpreter, the perfect platform to run and test your SQL queries efficiently. Our tool makes coding easy for developers of any skill level, whether you're a beginner or experienced.

Our compiler will allow you to

- Run your code fast
- Get detailed output and error description after each run
- You can also check the time and memory usage of your code
- Write your code faster using the auto-complete feature
- Use and import libraries
- Customize the editor with your favorite theme

What is an online compiler?

Online compilers are online code editors that let you run and test your code in a web browser easily. Use our code SQL editor above to write your queries, and execute them.

About SQL

SQL (Structured Query Language) is a standard programming language used for managing and manipulating relational databases. It is widely used for querying, updating, and managing data in databases.

```
46 SELECT Name, REPLACE(Name, 'A', 'Z') AS UpdatedName FROM Employee;
47 -- Extract the first character of each employee's Name
48 SELECT Name, SUBSTR(Name, 1, 1) AS FirstCharacter FROM Employee;
49 -- Find the position of a specific character in the employee's Name
50 SELECT Name, INSTR(Name, 'A') AS CharacterPosition FROM Employee;
51
52
53
54
55
56
57
58
59
```

```
A|1
B|0
C|0
D|0
E|0
F|0
```

Online SQL Compiler

Welcome to our online SQL compiler and interpreter, the perfect platform to run and test your SQL queries efficiently. Our tool makes coding easy for developers of any skill level, whether you're a beginner or experienced.

Our compiler will allow you to


- Run your code fast
- Get detailed output and error description after each run
- You can also check the time and memory usage of your code
- Write your code faster using the auto-complete feature
- Use and import libraries
- Customize the editor with your favorite theme

What is an online compiler?

Online compilers are online code editors that let you run and test your code in a web browser easily. Use our code SQL editor above to write your queries, and execute them.

About SQL

SQL (Structured Query Language) is a standard programming language used for managing and manipulating relational databases. It is widely used for querying,



[Courses](#) [Practice](#) [Compete](#) [Upgrade to Pro](#) [Login](#) [Sign Up](#)

Online SQL Compiler

Learning SQL? Check out our complete SQL roadmap

SQL

```
26 -- Find the average salary
27 SELECT AVG(Salary) AS AverageSalary FROM Employee;
28 -- Get the highest salary
29 SELECT MAX(Salary) AS HighestSalary FROM Employee;
30 -- Determine the lowest salary
31 SELECT MIN(Salary) AS LowestSalary FROM Employee;
32 SELECT Name, SUM(Salary) AS TotalSalary
33 FROM Employee
34 GROUP BY Name;
35 SELECT Name, SUM(Salary) AS TotalSalary
36 FROM Employee
37 GROUP BY Name
38 HAVING SUM(Salary) > 600;
39 -- Combine the Name of the employee with a custom string
40 SELECT Name || ' works at CodeChef' AS CombinedString FROM Employee;
41 -- Get the length of each employee's Name
42 SELECT Name, LENGTH(Name) AS StringLength FROM Employee;
43 -- Convert each employee's Name to upper and lower case
44 SELECT Name, UPPER(Name) AS UpperCase, LOWER(Name) AS LowerCase FROM Employee;
45 -- Replace a specific character in the employee's Name
46 SELECT Name, REPLACE(Name, 'A', 'Z') AS UpdatedName FROM Employee;
47 -- Extract the first character of each employee's Name
48 SELECT Name, SUBSTR(Name, 1, 1) AS FirstCharacter FROM Employee;
49 -- Find the position of a specific character in the employee's Name
50 SELECT Name, INSTR(Name, 'A') AS CharacterPosition FROM Employee;
51 -- Reverse each employee's Name
52 SELECT Name, LISTAGG(SUBSTR(Name, LEVEL, 1)) WITHIN GROUP (ORDER BY LEVEL DESC
53 ) AS ReversedName
54 FROM Employee
55 CONNECT BY LEVEL <= LENGTH(Name)
56 AND PRIOR Name = Name
57 AND PRIOR SYS_GUID() IS NOT NULL
58 GROUP BY Name;
```

Run

Enter Input here

If your code takes input, add it in the above box before running.

Output

Your Output

```
C|1
D|0
E|E
F|F
A|1
B|0
C|0
D|0
E|0
F|0
```

Runtime Error

```
SIGHUP
```

Error

```
Parse error near line 52: near "(": syntax error
```

Some Other String Function

These are the some other SQL Functions.

ASCII(): This function is used to find the ASCII value of a character.

Syntax: SELECT ascii('t');

Output: 116

CONCAT_WS(): This function is used to add two words or strings with a symbol as concatenating symbol.

Syntax: SELECT CONCAT_WS('_', 'sitnagpur', 'for', 'sitnagpur');

Output: sitnagpur_for_sitnagpur

FIND_IN_SET(): This function is used to find a symbol from a set of symbols.

Syntax: SELECT FIND_IN_SET('b', 'a, b, c, d, e, f');

Output: 2

FORMAT(): This function is used to display a number in the given format.

Syntax: SELECT FORMAT(0.981 * 100, 'N2') + '%' AS PercentageOutput;

Output: '98.10%'

INSTR(): This function is used to find the occurrence of an alphabet.

Syntax: INSTR('sitnagpur for sitnagpur', 'e');

Output: 2 (the first occurrence of 'e')

LCASE(): This function is used to convert the given string into lower case.

Syntax: LCASE ("SitnagpurFor Sitnagpur To Learn");

Output: sitnagpurforsitnagpur to learn

LOCATE(): This function is used to find the nth position of the given word in a string.

Syntax: SELECT LOCATE('for', 'sitnagpurforsitnagpur', 1);

Output: 6

LPAD(): This function is used to make the given string of the given size by adding the given symbol.

Syntax: LPAD('sitnagpur', 8, '0');

Output:

000sitnagpur

MID(): This function is to find a word from the given position and of the given size.

Syntax: Mid ("sitnagpurforsitnagpur", 6, 2);

Output: for

POSITION(): This function is used to find position of the first occurrence of the given alphabet.

Syntax: SELECT POSITION('e' IN 'sitnagpurforsitnagpur');

Output: 2

REPEAT(): This function is used to write the given string again and again till the number of times mentioned.

Syntax: SELECT REPEAT('sitnagpur', 2);

Output: sitnagpursitnagpur

REPLACE(): This function is used to cut the given string by removing the given sub string.

Syntax: REPLACE('123sitnagpur123', '123');

Output: sitnagpur

RPAD(): This function is used to make the given string as long as the given size by adding the given symbol on the right.

Syntax: RPAD('sitnagpur', 8, '0');

Output: 'sitnagpur000'

RTRIM(): This function is used to cut the given sub string from the original string.

Syntax: RTRIM('sitnagpurxyzzyy', 'xyz');

Output: 'sitnagpur'

SPACE(): This function is used to write the given number of spaces.

Syntax: SELECT SPACE(7);

Output: ' '

STRCMP(): This function is used to compare 2 strings.

Syntax: SELECT STRCMP('google.com', 'sitnagpur.edu.in');

Output: -1