EXPERIMENT-4

Lab 4: To implement and use Inbuilt Functions

Objective: To understand the use of SQL Inbuilt functions.

Inbuilt Functions

1. **Aggregate Functions**
2. **String Functions**
3. **Mathematical Functions**
4. **Date and Time Functions**
5. **Conditional Functions**

1. Aggregate Functions

COUNT(): Counts the number of rows that satisfy the specified condition.
Example: SELECT COUNT(name) FROM employee;

SUM(): Calculates the sum of the specified column values.
Example: SELECT SUM(working_hours) AS "Total working hours" FROM employee;

AVG(): Calculates the average of the specified column values.
Example: SELECT AVG(working_hours) AS "Average working hours" FROM employee;

MIN(): Retrieves the minimum value of the specified column.
Example: SELECT MIN(working_hours) AS Minimum_working_hours FROM employee;

MAX(): Retrieves the maximum value of the specified column.
SELECT MAX(working_hours) AS Maximum_working_hours FROM employee;

LIMIT: Limits the number of rows returned by a query.
Example: SELECT working_date FROM employee LIMIT 1;

ORDER BY with LIMIT: Orders the result set by a specified column and limits the number of rows returned.
Example: SELECT working_hours FROM employee ORDER BY name DESC LIMIT 1;

GROUP_CONCAT(): Concatenates values from multiple rows into a single string within each group.
Example: SELECT emp_id, emp_fname, emp_lname, dept_id, GROUP_CONCAT(designation) AS designation FROM employee2 GROUP BY emp_id, emp_fname, emp_lname, dept_id;

GROUP_CONCAT() with DISTINCT: Concatenates distinct values from multiple rows into a single string within each group.
Example: SELECT emp_fname, dept_id, GROUP_CONCAT(DISTINCT designation) AS designation FROM employee2 GROUP BY emp_id, emp_fname, emp_lname, dept_id;

GROUP_CONCAT() with Separator: Concatenates values using a specified separator within each group.
Example: SELECT emp_fname, GROUP_CONCAT(DISTINCT designation SEPARATOR ';') AS designation FROM employee2 GROUP BY emp_id, emp_fname, emp_lname, dept_id;

GROUP_CONCAT() with CONCAT_WS() and Separator: Concatenates values using a specified separator and a specified separator between values within each group.
Example: SELECT GROUP_CONCAT(CONCAT_WS(',', emp_lname, emp_fname) SEPARATOR ';') AS employeename FROM employee2;

2. MySQL String Function

CONCAT_WS(): Concatenates arguments using a specified separator.
CONCAT(): Concatenates all the arguments into a string.
CHARACTER_LENGTH(): Returns the size of the specified string.
ELT(): Returns the Nth element from a list of strings.
EXPORT_SET(): Returns a string for each bit set.
FIELD(): Returns the index of a string.
FIND_IN_SET(): Returns the value of a string given its position in the argument.
FORMAT(): Formats a number with a specified number of decimal places.
FROM_BASE64(): Encodes a given string to binary format.
HEX(): Returns a specified number or string in hexadecimal format.
INSERT(): Inserts a character into a string at a specified position.
INSTR(): Returns the first occurrence of a substring in a string.
LCASE(): Returns the given string in lowercase.
LEFT(): Returns a specified number of characters from the left side of a string.
LENGTH(): Returns the length of the specified string in bytes.
LIKE(): Checks for pattern matching and returns either 1 or 0.
LOAD_FILE(): Returns the content of a file.
LOCATE(): Returns the position of a given substring in a string.
LOWER(): Returns the given string in lowercase.
LPAD(): Left-pads a string to a specified length.
LTRIM(): Removes leading spaces from a string.
MAKE_SET(): Returns values from a set for the given bit.
MID(): Extracts a substring from a string.
OCTET_LENGTH(): Returns the length of a given string.
OCT(): Returns the octal representation of a number.
ORD(): Returns the code for the leftmost character if it is multi-byte.
POSITION(): Returns the position of a given substring in a string.
QUOTE(): Returns the string passed in single quotes.
REPEAT(): Repeats a string for a specified number of times.
REPLACE(): Replaces all occurrences of a substring within a string.
REVERSE(): Reverses a string.

RIGHT(): Extracts a specified number of characters from the right side of a string.
RPAD(): Pads the specified strings from the right.
RTRIM(): Removes trailing spaces from a specified string.
SOUNDEX(): Returns the soundex string for the specified string.

3. Mathematical Functions

ABS(): Returns the absolute value of the specified number.
ACOS(): Returns the arc cosine of the given number.
SIGN(): Returns the sign of the specified number.
SIN(): Returns the sine value of the given number.
SQRT(): Returns the square root of the given number.
SUM(): Sums the values of given expressions.
TAN(): Returns the tangent of the given number.
TRUNCATE(): Truncates the given number up to a certain number of decimal places.
ASIN(): Returns the arc sine of the given number.
ATAN2(): Returns the arc tangent of the specified numbers n and m.
ATAN(): Returns the arc tangent of the specified number.
AVG(): Calculates the average value from the given expression.
CEIL() (or CEILING()): Returns the smallest value greater than or equal to the specified number.
COS(): Returns the cosine of the given number.
COT(): Returns the cotangent of the given number.
COUNT(): Gets the total count for the specified column of the table.
DEGREES(): Converts the given radian number into a degree.
DIV(): Finds the integer division by dividing the number n by the number m.
EXP(): Finds e raised to the power of the given number.
FLOOR(): Finds the greatest integer less than or equal to the specified number.
GREATEST(): Gets the largest number from the list.
LEAST(): Gets the smallest number from the list.
LN(): Gets the natural logarithm for the specified number.
LOG10(): Gets the base-10 logarithm for the specified number.
LOG(): Returns either the natural logarithm or the specified base logarithm of the given number.
LOG2(): Gets the base-2 logarithm for the specified number.
MAX(): Gets the maximum number from the given column.
MIN(): Gets the minimum number from the given column.
MOD(): Gets the remainder for the specified values.
PI(): Gets the value of pi up to 6 decimal places.
POWER(): Gets the power for the specified values.
POW(): Gets the power for the specified values.
RADIANS(): Converts the given degrees to radians.
RAND(): Generates a random number.
ROUND(): Rounds off the specified number.

Lab Performance Questions

```
CREATE TABLE employee(
   name varchar(45) NOT NULL,
   occupation varchar(35) NOT NULL,
   working_date date,
   working_hours varchar(10)
);

desc employee;

INSERT INTO employee VALUES
('Robin', 'Scientist', '2020-10-04', 12),
('Warner', 'Engineer', '2020-10-04', 10),
('Peter', 'Actor', '2020-10-04', 13),
('Marco', 'Doctor', '2020-10-04', 14),
('Brayden', 'Teacher', '2020-10-04', 12),
('Antonio', 'Business', '2020-10-04', 11);

select * from employee;
```

1. `SELECT COUNT(name) FROM employee;`
2. `SELECT SUM(working_hours) AS "Total working hours" FROM employee;`
3. `SELECT AVG(working_hours) AS "Average working hours" FROM employee;`
4. `SELECT MIN(working_hours) AS Minimum_working_hours FROM employee;`
5. `SELECT MAX(working_hours) AS Maximum_working_hours FROM employee;`
6. `SELECT working_date FROM employee LIMIT 1;`
7. `SELECT working_hours FROM employee ORDER BY name DESC LIMIT 1;`

```
CREATE TABLE employee2(
   emp_id varchar(45),
   emp_fname varchar(10),
   emp_lname varchar(10),
   dept_id varchar(10),
   designation varchar(25)
);

desc employee2;

INSERT INTO employee2 VALUES
(1, 'David', 'Miller', 2, 'Engineer'),
(2, 'Peter', 'Watson', 3, 'Manager'),
(3, 'Mark', 'Boucher', 1, 'Scientist'),
(2, 'Peter', 'Watson', 3, 'BDE'),
(1, 'David', 'Miller', 2, 'Developer'),
(4, 'Adam', 'Warner', 4, 'Receptionist'),
(3, 'Mark', 'Boucher', 1, 'Engineer'),
(4, 'Adam', 'Warner', 4, 'Clerk');

select * from employee2;
```

8. SELECT emp_id, emp_fname, emp_lname, dept_id, GROUP_CONCAT(designation) AS designation FROM employee2 GROUP BY emp_id, emp_fname, emp_lname, dept_id;

9. SELECT emp_fname, dept_id, GROUP_CONCAT(DISTINCT designation) as designation FROM employee2 group by emp_id, emp_fname, emp_lname, dept_id;

10. SELECT emp_fname, GROUP_CONCAT(DISTINCT designation SEPARATOR ';') as designation FROM employee2 group by emp_id, emp_fname, emp_lname, dept_id;

11. SELECT GROUP_CONCAT(CONCAT_WS(',', emp_lname, emp_fname) SEPARATOR ';') as employeename FROM employee2;

MySQL String Function

```
CREATE TABLE sample_table (
   id INT,
   first_name VARCHAR(50),
   last_name VARCHAR(50)
);
 desc sample_table;

INSERT INTO sample_table (id, first_name, last_name)
VALUES
   (1, 'John', 'Doe'),
   (2, 'Jane', 'Smith'),
   (3, 'Alice', 'Johnson');
```

12. SELECT id FROM sample_table;

13. SELECT first_name FROM sample_table;

14. SELECT last_name FROM sample_table;

15. SELECT CONCAT_WS('-', first_name, last_name) AS concat_ws_result FROM sample_table;

16. SELECT CONCAT(first_name, ' ', last_name) AS concat_result FROM sample_table;

17. SELECT CHARACTER_LENGTH(first_name) AS char_length_result FROM sample_table;

18. SELECT ELT(1, first_name, last_name, 'Other') AS elt_result FROM sample_table;

19. SELECT EXPORT_SET(id, 1, ',', '') AS export_set_result FROM sample_table;

20. SELECT FIELD(first_name, 'Alice', 'Jane', 'John') AS field_result FROM sample_table;

21. SELECT FIND_IN_SET(first_name, 'John,Jane,Alice') AS find_in_set_result FROM sample_table;

22. SELECT FORMAT(id, 2) AS format_result FROM sample_table;

23. SELECT FROM_BASE64('SGVsbG8gd29ybGQh') AS from_base64_result FROM sample_table;

24. SELECT HEX(id) AS hex_result FROM sample_table;

25. SELECT INSERT(first_name, 2, 0, 'X') AS insert_result FROM sample_table;

26. SELECT INSTR(first_name, 'hn') AS instr_result FROM sample_table;

27. SELECT LCASE(first_name) AS lcase_result FROM sample_table;

28. SELECT LEFT(first_name, 2) AS left_result FROM sample_table;

29. SELECT LENGTH(first_name) AS length_result FROM sample_table;

30. SELECT first_name LIKE 'J%' AS like_result FROM sample_table;

31. SELECT LOAD_FILE('/path/to/file') AS load_file_result FROM sample_table;

32. SELECT LOCATE('a', first_name) AS locate_result FROM sample_table;

33. SELECT LOWER(first_name) AS lower_result FROM sample_table;

34. SELECT LPAD(first_name, 10, '*') AS lpad_result FROM sample_table;

35. SELECT LTRIM(' ' || first_name) AS ltrim_result FROM sample_table;

36. SELECT MAKE_SET(2, 'Red', 'Green', 'Blue') AS make_set_result FROM sample_table;

37. SELECT MID(first_name, 2, 2) AS mid_result FROM sample_table;

38. SELECT OCTET_LENGTH(first_name) AS octet_length_result FROM sample_table;

39. SELECT OCT(id) AS oct_result FROM sample_table;

40. SELECT ORD(LEFT(first_name, 1)) AS ord_result FROM sample_table;

41. SELECT POSITION('oh' IN first_name) AS position_result FROM sample_table;

42. SELECT QUOTE(first_name) AS quote_result FROM sample_table;

43. SELECT REPEAT(first_name, 2) AS repeat_result FROM sample_table;

44. SELECT REPLACE(first_name, 'Jo', 'X') AS replace_result FROM sample_table;

45. SELECT REVERSE(first_name) AS reverse_result FROM sample_table;

46. SELECT RIGHT(last_name, 2) AS right_result FROM sample_table;

47. SELECT RPAD(first_name, 10, '*') AS rpad_result FROM sample_table;

48. SELECT RTRIM(first_name) AS rtrim_result FROM sample_table;

49. SELECT SOUNDEX(first_name) AS soundex_result FROM sample_table;

Mathematical Functions

```
-- Creating a sample employee table
CREATE TABLE employee (
    employee_id INT PRIMARY KEY,
    salary DECIMAL(10, 2),
    age INT
);

-- Inserting sample data into the employee table
INSERT INTO employee (employee_id, salary, age)
VALUES
    (1, 50000.25, 30),
    (2, 60000.75, 35),
    (3, 75000.50, 28);

-- Using various functions in SQL queries on the employee table
```

50. SELECT employee_id FROM employee;

51. SELECT ABS(salary) AS abs_salary FROM employee;

52. SELECT ACOS(salary / 100000) AS acos_salary FROM employee;

53. SELECT SIGN(salary) AS sign_salary FROM employee;

54. SELECT SIN(salary / 100000) AS sin_salary FROM employee;

55. SELECT SQRT(salary) AS sqrt_salary FROM employee;

56. SELECT salary AS sum_salary FROM employee;

57. SELECT TAN(salary / 100000) AS tan_salary FROM employee;

58. SELECT TRUNCATE(salary, 2) AS truncated_salary FROM employee;

59. SELECT ASIN(salary / 100000) AS asin_salary FROM employee;

60. SELECT ATAN2(salary, age) AS atan2_salary FROM employee;

61. SELECT ATAN(salary / 100000) AS atan_salary FROM employee;

62. SELECT salary AS avg_salary FROM employee;

63. SELECT CEIL(salary) AS ceil_salary FROM employee;

64. SELECT CEILING(salary) AS ceiling_salary FROM employee;

65. SELECT COS(salary / 100000) AS cos_salary FROM employee;

66. SELECT COT(salary / 100000) AS cot_salary FROM employee;

67. SELECT 1 AS count_employee FROM employee;

68. SELECT DEGREES(salary / 100000) AS degrees_salary FROM employee;

69. SELECT salary DIV 2 AS div_salary FROM employee;

70. SELECT EXP(salary / 100000) AS exp_salary FROM employee;

71. SELECT FLOOR(salary) AS floor_salary FROM employee;

72. SELECT GREATEST(salary, 0, -50000) AS greatest_salary FROM employee;

73. SELECT LEAST(salary, 0, -50000) AS least_salary FROM employee;

74. SELECT LN(salary / 100000) AS ln_salary FROM employee;

75. SELECT LOG10(salary / 100000) AS log10_salary FROM employee;

76. SELECT LOG(salary / 100000, 2) AS log_salary FROM employee;

77. SELECT LOG2(salary / 100000) AS log2_salary FROM employee;

78. SELECT salary AS max_salary FROM employee;

79. SELECT salary AS min_salary FROM employee;

80. SELECT MOD(salary, 3) AS mod_salary FROM employee;

81. SELECT PI() AS pi_value FROM employee;

82. SELECT POWER(salary, 2) AS power_salary FROM employee;

83. SELECT POW(salary, 2) AS pow_salary FROM employee;

84. SELECT RADIANS(salary / 100000) AS radians_salary FROM employee;

85. SELECT RAND() AS rand_value FROM employee;

86. SELECT ROUND(salary, 2) AS round_salary FROM employee;