Table 1: employee Note: We'll refer this table for all the queries in this sheet					
employee_id [integer]	**first_name [characters]**	**last_name [characters]**	**salary [integer]**	**department [characters]**	**city [characters]**
1	John	Abraham	1000000	Banking	Delhi
2	Michael	Clarke	800000	Insurance	Bangalore
3	Roy	Thomas	700000	Banking	Gujarat
4	Tom	Jose	600000	Insurance	Delhi
5	Jerry	Pinto	650000	Insurance	Bangalore
6	Philip	Mathew	750000	Services	Chandigarh
7	Amir	Khan	650000	Services	Delhi

Column Aliases

Column aliases are used to give temporary name to column. Column alias do not modify actual table.

Query Syntax:

```
SELECT column_name
AS column_alias
FROM table_name;
```

Example 1: Get department of employee with alias name "employee_dept".

Query:

```
1  SELECT department
2  AS employee_dept
3  FROM employee;
```

Result:

```
+---------------+
| employee_dept |
+---------------+
| Banking       |
| Insurance     |
| Banking       |
| Insurance     |
| Insurance     |
| Services      |
| Services      |
+---------------+
7 rows in set (0.00 sec)
```

Explanation: Here you can see in the output that "department" name come with different name (or alias name) as "employee_dept".

Keyword: DISTINCT

Distinct keyword removes duplicates and returns result with unique entries.

Query Syntax [With SELECT]:

```
SELECT DISTINCT column_1, column_2, ... column_n
FROM table_name;
```

Example 2: Get unique department names of employee table.

Query:

```
1   SELECT DISTINCT department
2   FROM employee;
```

Result:

```
+------------+
| department |
+------------+
| Banking    |
| Insurance  |
| Services   |
+------------+
3 rows in set (0.00 sec)
```

Explanation: Here you can see in the output that "department" column contains only three entries after removal of duplicates.

Aggregate Functions

Aggregate functions perform a calculation on a set of values and return a single value. They are frequently used with different clauses in SQL. Some common aggregate functions are:

Function	Description
AVG()	Returns the average value
COUNT()	Returns the number of rows
MAX()	Returns the largest value
MIN()	Returns the smallest value
SUM()	Returns the sum
LENGTH()	Returns the length

Example 3: Get average of salary from employee table.

Query:

```
1   SELECT AVG(salary)
2   AS Salary_Average
3   FROM employee;
```

Result:

```
+----------------+
| Salary_Average |
+----------------+
|    735714.2857 |
+----------------+
1 row in set (0.00 sec)
```

Example 4: Obtain a count of all employees from the Employee table.

Query:

```
1  SELECT COUNT(employee_id)
2  FROM employee;
```

Result:

```
+--------------------+
| COUNT(employee_id) |
+--------------------+
|                  7 |
+--------------------+
1 row in set (0.00 sec)
```

Example 5: Count all rows from the Employee table.

Query:

```
1  SELECT COUNT(*)
2  AS RowCount
3  FROM employee;
```

Result:

```
+----------+
| RowCount |
+----------+
|        7 |
+----------+
1 row in set (0.00 sec)
```

Example 6: Count different department from the employee table.

Query:

```
1  SELECT COUNT(DISTINCT department)
2  AS department
3  FROM employee;
```

Result:

```
+------------+
| department |
+------------+
|          3 |
+------------+
1 row in set (0.00 sec)
```

Example 7: Return highest salary from the Employee table.

Query:

```
1  SELECT MAX(salary)
2  AS MaximumSalary
3  FROM employee;
```

Result:

```
+---------------+
| MaximumSalary |
+---------------+
|       1000000 |
+---------------+
1 row in set (0.00 sec)
```

Example 8: Return sum of a salary of all employees from employee table.

Query:

```
1  SELECT SUM(salary)
2  AS Sum_Of_Salary
```

Result:

```
+---------------+
| Sum_Of_Salary |
```

```
3  FROM employee;
```

```
+---------------+
|       5150000 |
+--------------+
1 row in set (0.00 sec)
```

Example 9: Get name of the employee from employee table whose salary is greater than average salary.

Query:

```
1  SELECT first_name
2  FROM employee
3  WHERE salary > (SELECT AVG(salary) FROM employee);
```

Result:

```
+------------+
| first_name |
+------------+
| John       |
+------------+
| Michael    |
+------------+
| Philip     |
+------------+
3 rows in set (0.00 sec)
```

Problem 1	**Refer table structure given below.**

Table Structure

Table 2: student

student_id [integer]	first_name [characters]	last_name [characters]	age [integer]	city [characters]	fees [integer]
1	Vikas	Kumar	18	Delhi	2000
2	Neha	Jain	14	Gurugram	4500
3	Girish	Sharma	15	Delhi	3000

Problem 1.1	**Get average age from student table.**
Problem 1.2	**Get count of all students present in table as Total_Student_Count.**
Problem 1.3	**Get unique cities from student table.**
Problem 1.4	**Get sum of fees of all students as Total_Fees.**
Problem 1.5	**Print the minimum fees from students table.**
Problem 1.6	**Get count of different cities as city_count from student table.**
Problem 1.7	**Get names of those students who pay more than the average fees of all students.**
Problem 1.8	**Get details of all students who have to pay fees more than minimum fee or less than maximum fee.**
Problem 1.9	**Get id and first name of those students having age not in the range of 0 and average age of students.**
Problem 1.10	**Get details of students having age between average age and maximum age of students.**
Problem 1.11	**Get names of students having maximum age and minimum salary.**