SQL Cheat Sheet

This cheat Sheet is designed to assist learners in understanding some of the basic SQL queries. This would be helpful when learning database operations in Spring Boot with suitable examples. Throughout this sheet, we will use a scenario to see how to employ different types of SQL commands.

Scenario: You have a database named "**Company**" with two tables: "**Employees**" and "**Departments**". The "Employees" table contains employee information, such as their ID, name, department ID, and salary. The "Departments" table contains department information, such as their ID and name.

id	name
1	Sales
2	Finance

Department Table

id	name	salary	department_id
1	Ramesh Verma	20000	2
2	Mahesh Anand	25000	1

Employee Table

1. SELECT statement:

a. Retrieve all columns from the Employee table:

```
SELECT * FROM Employees;
```

b. Retrieve specific columns from the Employee table:

```
SELECT name, salary from Employees;
```

c. Select unique salaries from the Employee table:

```
SELECT DISTINCT salary from Employees;
```

d. Select salary as alias compensation:

```
SELECT salary as Compensation from Employees;
```

e. Filter rows based on a condition:

```
SELECT * from Employees WHERE condition;
```

f. Sort the employees based on their salaries:

```
SELECT * from Employees ORDER BY SALARY ASC|DESC;
```

g. Group rows based on a column and apply aggregate functions:

```
SELECT column, function(column) FROM table_name GROUP BY column;
```

The aggregate function can be min(), max(), avg() etc. We will discuss these functions below.

h. Limit the number of rows returned in the result set:

```
SELECT name, salary from Employees LIMIT 1000;
```

2. Matching:

a. Matching data using **LIKE**

```
SELECT * from Employees WHERE name=”J%”;
```

This query returns the list of employees whose name starts with J.

b. We can also use regular expressions to match data.

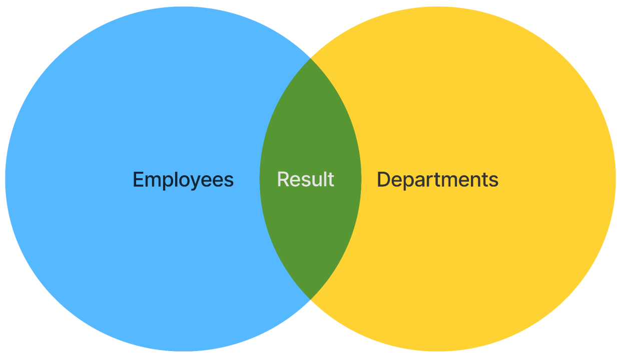
```
SELECT * from Employees WHERE name=”regex”;
```

3. Joins:

a. **Inner Join**: Retrieve employee name and department name for employees in the "IT" department:

```
SELECT Employees.name, Departments.name
FROM Employees
JOIN Departments ON Employees.department_id = Departments.id
WHERE Departments.name = 'IT';
```

The inner join returns rows with a match on both tables.



b. **Outer Join:** The outer join returns all rows from the left table (table1) and the right table (table2). If there is no match, NULL values are returned for the columns of the respective non-matching table.

   Example: Retrieve employee name and department name for all employees and departments, including unmatched records from both tables.

```
SELECT Employees.name, Departments.name
FROM Employees
FULL OUTER JOIN Departments
ON Employees.department_id = Departments.id;
```

c. **Left Join:** The left join returns all rows from the left table (table1) and the matched rows from the right table (table2). If there is no match, NULL values are returned for the columns of the right table.

   Example: Retrieve employee name and department name for all employees, including those without a department assigned.

```
SELECT Employees.name, Departments.name
FROM Employees
LEFT JOIN Departments
ON Employees.department_id = Departments.id;
```

d. **Right Join:** The right join returns all rows from the right table (table 2) and the matched rows from the left table (table 1). If there is no match, NULL values are returned for the columns of the left table.

   Example: Retrieve employee and department names for all departments, including those without employees.

```
SELECT Employees.name, Departments.name
```

```
FROM Employees
RIGHT JOIN Departments
ON Employees.department_id = Departments.id;
```

4. MYSQL Calculation Functions:

a. **Perform Mathematical Calculations:**
   Example: Calculate the total salary for an employee, including a 10% bonus:

   ```
   SELECT salary, salary * 1.1 AS total_salary FROM Employees;
   ```

b. Get the maximum salary in the company:

   ```
   SELECT Max(salary) AS maximum_salary FROM Employees;
   ```

c. Get the minimum salary in the company:

   ```
   SELECT MIN(salary) AS minimum_salary FROM Employees;
   ```

d. Calculate the sum of salaries of all employees in the company:

   ```
   SELECT SUM(salary) AS sum_of_salaries FROM Employees;
   ```

e. Calculate the number of employees in the company

   ```
   SELECT COUNT(*) AS number_of_employees FROM Employees;
   ```

5. String functions in MYSQL

a. **SUBSTR:** The SUBSTR function extracts a substring from a string based on a specified starting position and length.

   Example: Extract the first three characters of an employee's name.

   ```
   SELECT SUBSTR(name, 1, 3) AS initials FROM Employees;
   ```

b. **UPPER**: The UPPER function converts a string to uppercase.

   Example: Convert the employee names to uppercase.

   ```
   SELECT UPPER(name) AS uppercase_name FROM Employees;
   ```

c. **LOWER**: The LOWER function converts a string to lowercase.

Example: Convert the department names to lowercase.

```
SELECT LOWER(name) AS lowercase_name FROM Departments;
```

d. **STRCMP**: The STRCMP function compares two strings and returns 0 if they are equal, a negative value if the first string is less than the second, or a positive value if the first string is greater than the second.

   Example: Compare the names of two employees.

```
SELECT STRCMP(name, 'John Doe') AS name_comparison FROM
Employees;
```

e. **CONCAT**: The CONCAT function concatenates two or more strings together.

   Example: We have first and last names in different columns and want to concatenate them.

```
SELECT CONCAT(first_name, ' ', last_name) AS full_name FROM
Employees;
```