SLQ CHEAT SHEET

Sample Data

id	name	department	salary	hire_date
1	Alice	HR	60000	2018-06-15
2	Bob	IT	75000	2019-07-01
3	Charlie	Finance	55000	2020-09-23
4	David	IT	80000	2017-12-11
5	Emma	HR	62000	2021-01-20

1. Basic SQL Commands

- **SELECT – Retrieve data from a table**

SELECT column1, column2 FROM table_name;

- **FROM – Specifies the table to query**

SELECT * FROM table_name;

- **WHERE – Filter records based on a condition**

SELECT * FROM table_name WHERE column1 = 'value';
EXAMPLE: SELECT * FROM employees WHERE salary >= 60000;

- **ORDER BY – Sort the result set**

SELECT * FROM table_name ORDER BY column1 ASC|DESC;
EXAMPLE: SELECT * FROM employees ORDER BY salary DESC;

- **LIMIT – Restrict the number of records returned**

SELECT * FROM table_name LIMIT 10;
EXAMPLE: SELECT * FROM employees LIMIT 3;

- **DISTINCT – Retrieve unique values**

SELECT DISTINCT column1 FROM table_name;
EXAMPLE: SELECT DISTINCT

2. Filtering

- **LIKE – Pattern matching**

SELECT * FROM table_name WHERE column1 LIKE 'A%';
EXAMPLE: SELECT * FROM employees WHERE name LIKE 'A%';

- **IN / NOT IN – Match a list of values**

SELECT * FROM table_name WHERE column1 IN ('value1', 'value2');
EXAMPLE: SELECT * FROM employees WHERE department IN ('IT', 'HR');

- **OR / AND – Combine conditions**

SELECT * FROM table_name WHERE column1 = 'value' AND column2 > 10;

- **ANY, ALL → Compare against multiple values**

EXAMPLE:
SELECT *
FROM employees
WHERE salary > ANY (SELECT salary FROM employees WHERE department = 'HR');

3. Joins (Combining Tables)

- **INNER JOIN – Match records in both tables**

SELECT a.column1, b.column2 FROM tableA a INNER JOIN tableB b ON a.id = b.id;
EXAMPLE:
SELECT employees.name, employees.salary, departments.department_name
FROM employees
INNER JOIN departments ON employees.department = departments.department_name;

- **LEFT JOIN – All records from the left table + matching records from the right table**

SELECT a.column1, b.column2 FROM tableA a LEFT JOIN tableB b ON a.id = b.id;

- **RIGHT JOIN – All records from the right table + matching records from the left table**

SELECT a.column1, b.column2 FROM tableA a RIGHT JOIN tableB b ON a.id = b.id;

- **FULL JOIN – All records from both tables**

SELECT a.column1, b.column2 FROM tableA a FULL JOIN tableB b ON a.id = b.id;

- **IMPLICIT JOIN – Joining tables without JOIN keyword**

SELECT a.column1, b.column2 FROM tableA a, tableB b WHERE a.id = b.id;
EXAMPLE:
SELECT employees.name, departments.department_name
FROM employees, departments
WHERE employees.department = departments.department_name;

4. Aggregations & Grouping

- **COUNT() – Count records**

SELECT COUNT(*) FROM table_name;
EXAMPLE: SELECT department, COUNT(*) FROM employees GROUP BY department;

- **SUM() – Total sum of a column**

SELECT SUM(column_name) FROM table_name;

- **AVG() – Average value**

SELECT AVG(column_name) FROM table_name;

- **MIN() / MAX() – Minimum & Maximum values**

SELECT MIN(column_name), MAX(column_name) FROM table_name;

- **GROUP BY – Group data by specific columns**

SELECT column1, COUNT(*) FROM table_name GROUP BY column1;

- **HAVING – Filter grouped records**

SELECT column1, COUNT(*) FROM table_name GROUP BY column1 HAVING COUNT(*) > 1;
EXAMPLE: SELECT department, AVG(salary) FROM employees GROUP BY department HAVING AVG(salary) > 60000;

5. Datetime Functions

SELECT CURRENT_DATE, CURRENT_TIME;
SELECT EXTRACT(YEAR FROM column_name) FROM table_name;
DATE_TRUNC() – Truncate date values
SELECT DATE_TRUNC('month', column_name) FROM table_name;
EXAMPLE:
SELECT NOW(), CURRENT_DATE, EXTRACT(YEAR FROM hire_date) FROM employees;

6. Set Operations

- **UNION – Combine results of two queries (removes duplicates)**

SELECT column1 FROM tableA UNION SELECT column1 FROM tableB;

- **INTERSECT – Return common records between queries**

SELECT column1 FROM tableA INTERSECT SELECT column1 FROM tableB;

- **EXCEPT – Return records from first query not in second**

SELECT column1 FROM tableA EXCEPT SELECT column1 FROM tableB;

7. Window Functions

- **PARTITION BY – Define partitions for window functions**

SELECT column1, column2, RANK() OVER (PARTITION BY column1 ORDER BY column2) FROM table_name;

- **ROW_NUMBER() – Assign row numbers**

SELECT column1, ROW_NUMBER() OVER (ORDER BY column2) FROM table_name;

- **RANK() / DENSE_RANK() – Rank records with ties**

SELECT column1, RANK() OVER (ORDER BY column2) FROM table_name;
EXAMPLE:
SELECT name, department, salary,
RANK() OVER (PARTITION BY department ORDER BY salary DESC) as rank
FROM employees;

8. Conditional Logic & Existence Checks

- **CASE WHEN – Conditional statements**

```
SELECT column1,
    CASE WHEN column2 > 10 THEN 'High' ELSE 'Low' END AS category
FROM table_name;
EXAMPLE:
SELECT name, salary,
    CASE
      WHEN salary > 70000 THEN 'High'
      WHEN salary > 60000 THEN 'Medium'
      ELSE 'Low'
    END AS salary_category
FROM employees;
```

- **EXISTS – Check if a subquery returns any results**

SELECT * FROM table_name WHERE EXISTS (SELECT 1 FROM another_table WHERE condition);

- **ANY / ALL – Compare against a subquery**

SELECT * FROM table_name WHERE column1 > ANY (SELECT column1 FROM another_table);

9. Common Table Expressions (CTEs)

```
WITH temp_table AS (
   SELECT column1, column2 FROM table_name WHERE condition
)
SELECT * FROM temp_table;
EXAMPLE:
WITH high_earners AS (
   SELECT * FROM employees WHERE salary > 70000
)
SELECT * FROM high_earners;
```