Week 3: WHERE and ORDER BY and Built-in commands

Order of typing command

SELECT column_list
FROM source_table_or_view
WHERE filter_condition
GROUP BY grouping_condition
HAVING grouping_filter_condition
ORDER BY sort_order;

Order of execution of commands

FROM
WHERE
GROUP BY
HAVING
SELECT
ORDER BY

Select command is used for displaying the entire table data

```
SELECT *
FROM employee;
```

For Sorting the data we can use the ORDER BY command

```
SELECT parameter_1, parameter_2, parameter_3
FROM table_name
ORDER BY parameter_2;
-- or
SELECT parameter_1, parameter_2, parameter_3
FROM table_name
ORDER BY parameter_2 ASC;
-- or
SELECT parameter_1, parameter_2, parameter_3
```

```
FROM table_name
ORDER BY parameter_2 DESC;
-- or
SELECT parameter_1, parameter_2, parameter_3
FROM table_name
ORDER BY parameter_1 ASC, parameter_2 ASC;
-- or
SELECT parameter_1, parameter_2, parameter_3
FROM table_name
ORDER BY parameter_1 ASC, parameter_2 DESC;
```

Column name alias are used to change the column names displayed on the output window

```
SELECT parameter_1 as "First", parameter_2 "Second Parameter", parameter_3 "Parameter-3"
FROM table_name
ORDER BY parameter_1 ASC, parameter_2 ASC;
```

There are two ways of concatenating two or more columns

```
-- Concatenating 'only' two columns
SELECT CONCAT((CONCAT(parameter_1, ' '), parameter_2)) "Column Alias", parameter_3 "Parameter 3"
FROM table_name
ORDER BY parameter_1 ASC, parameter_2 ASC;

-- Concatenating more than two columns
SELECT parameter_1 ||  ' ' || parameter_2 "Column Alias", parameter_3 "Parameter 3"
FROM table_name
ORDER BY parameter_1 ASC, parameter_2 ASC;
```

Removing duplicates from output window

```
SELECT DISTINCT parameter_1
FROM table_name
ORDER BY parameter_1 ASC;

SELECT DISTINCT parameter_1, parameter_2
FROM table_name
ORDER BY parameter_1 ASC;
```

Using WHERE statement

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_2 = 3;
```

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_2 = 3 or parameter_2 = 2;
```

```
-- Using OR condition
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_2 = 3 OR parameter_2 = 2
ORDER BY parameter_2;

-- Using AND condition
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_2 = 3 OR parameter_2 = 2
ORDER BY parameter_2;
```

Using pattern finding by LIKE command

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_1 LIKE 'A%'
ORDER BY parameter_2;
```

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_1 LIKE 'A%95\%' -- slashing % character
ORDER BY parameter_2;
```

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_1 LIKE 'A%'
ORDER BY parameter_2;
```

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_1 LIKE '%land'
ORDER BY parameter_2;
```

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_1 LIKE '%land%'
ORDER BY parameter_2;
```

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_1 LIKE '_a%'
ORDER BY parameter_2;
```

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE parameter_1 LIKE '_a%a'
ORDER BY parameter_2;
```

```
SELECT parameter_1, parameter_2
FROM table_name
WHERE (parameter_1 LIKE '%land%' OR parameter_1 LIKE '%pore%') AND parameter_2 = 3
ORDER BY parameter_2;
```

Built-in functions

```
SELECT UPPER(parameter_1), LOWER(parameter_2)
FROM table_name
WHERE (parameter_1 LIKE '%land%' OR parameter_1 LIKE '%pore%') AND parameter_2 = 3
ORDER BY parameter_2;
```

Using alias and WHERE is not possible because SELECT is executed after WHERE

```
SELECT first_name ||  ' ' || last_name "Employee Name", job_id
FROM employee
```

```
WHERE "Employee Name" LIKE '%a%' -- this will show an error
ORDER BY first_name;
```