Basic Commands
++++++++++++++

more queries:

```
SELECT name AS first_name
FROM employee;

SELECT DISTINCT gender
FROM employee;
```

+++++++++++++++++++++++++

SQL aggregation is the task of collecting a set of values to return a single value. It is done with the help of aggregate functions, such as SUM, COUNT, and AVG.

built-in functions:

```
SELECT COUNT(supervisor_id)
FROM employee;

SELECT AVG(salary)
FROM employee;

SELECT SUM(salary)
FROM employee;

SELECT COUNT(supervisor_id)
FROM employee
WHERE gender = 'F';
```

group what you get back by gender

```
SELECT COUNT(gender), gender
FROM employee
GROUP BY gender;

SELECT SUM(total_sales), employee_id
FROM works_with
GROUP BY client_id;
```

+++++++++++++++++++++++++

what are wildcards?
wildcards are a way of defining a patterns that we wanna match specific pieces of data to.

% means any number of characters
_ means one character

```
SELECT *
FROM branch
WHERE branch_name LIKE '%ford';

SELECT *
FROM branch
WHERE branch_name LIKE '%for%';
```

+++++++++++++++++++++++++

what is UNION?
the UNION operator is used to combine the result-set of two or more SELECT statements
UNION rules:
every SELECT statement within UNION must have the same number of columns
the columns must also have similar data types
the columns in every SELECT statement must also be in the same order

```
SELECT employee.first_name AS Employee_Branch_Names
FROM employee
UNION
SELECT branch.branch_name
FROM branch;
```

+++++++++++++++++++++++++

what is JOIN?
the JOIN operator is used to combine rows from two or more tables, based on a related column between them

```
SELECT employee.emp_id, employee.first_name, branch.branch_name
FROM employee
JOIN branch    -- LEFT JOIN, RIGHT JOIN
ON employee.emp_id = branch.mgr_id;
```

display employee.employee_id, employee.name, branch.branch_name after matching employee.employee_id to branch.manager_id(both of them contains employee_id)

+++++++++++++++++++++++++

nestet queries:

```
read from inside to outside

SELECT employee.name
FROM employee
WHERE employee.employee_id IN (
                    SELECT works_with.employee_id
                    FROM works_with
                    WHERE works_with.total_sales > 14);

SELECT client.client_id, client.client_name
FROM client
WHERE client.branch_id = (
                    SELECT branch.branch_id
                    FROM branch
                    WHERE branch.manager_id = (
                                    SELECT employee.employee_id
                                    FROM employee
                                    WHERE employee.name = 'Kim'
                                    LIMIT 1));

+++++++++++++++++++++++++

what is a trigger?
a trigger is a stored procedure in a database that automatically invokes
whenever a special event in the database occurs

what is a delimeter?
a special character used to signal the end of a SQL statement

CREATE TABLE trigger_test (
     message VARCHAR(100)
);

creating triggers:
first change the delimeter before creating a new trigger
DELIMITER $$

then create a trigger ending it with the delimeter you sat
CREATE
    TRIGGER my_trigger BEFORE INSERT
    ON employee
    FOR EACH ROW BEGIN
        INSERT INTO trigger_test VALUES('added new employee');
    END$$

finally change back the delimeter to semicolon
DELIMITER ;

DELIMITER $$
```

```
CREATE
    TRIGGER my_trigger BEFORE INSERT
    ON employee
    FOR EACH ROW BEGIN
        INSERT INTO trigger_test VALUES(NEW.first_name);
    END$$
DELIMITER ;

DELIMITER $$
CREATE
    TRIGGER my_trigger BEFORE INSERT
    ON employee
    FOR EACH ROW BEGIN
         IF NEW.gender = 'M' THEN
               INSERT INTO trigger_test VALUES('added male employee');
         ELSEIF NEW.gender = 'F' THEN
               INSERT INTO trigger_test VALUES('added female');
         ELSE
               INSERT INTO trigger_test VALUES('added other employee');
         END IF;
    END$$
DELIMITER ;
```

deleting triggers:

```
DROP TRIGGER my_trigger;
```