Operators and functions

Select statement with where clause

The WHERE clause is used to specify a selection criterion.

The WHERE Clause

To conditionally select data from a table, a WHERE clause can be added to the SELECT statement.

```
SELECT column FROM table WHERE column operator
value;
```

```
SELECT column1,column2 FROM table WHERE column
operator value;
```

Select clause can contain arithmetic expressions involving the operation +, -, *, and / also operating on constants or attributes of tuples.

Find ID, name and one-fifth of the salary of the employees

```
Select ID, name, salary/5 from employees;
```

Using the WHERE Clause

To select only the persons living in the city "Sandnes", we add a WHERE clause to the SELECT statement:

```
SELECT * FROM Persons  WHERE City='Sandnes'
```

LastName	FirstName	Address	City	Year
Hansen	Ola	Timoteivn 10	Sandnes	1951
Svendson	Tove	Borgvn 23	Sandnes	1978
Svendson	Stale	Kaivn 18	Sandnes	1980
Pettersen	Kari	Storgt 20	Stavanger	1960

Output

LastName	FirstName	Address	City	Year
Hansen	Ola	Timoteivn 10	Sandnes	1951
Svendson	Tove	Borgvn 23	Sandnes	1978
Svendson	Stale	Kaivn 18	Sandnes	1980

Where Clause :

Specifies the conditions that the result most satisfy corresponds to the selection predicate of the relational algebra
Example : Find the names of all the employees of SAS school

```
select name from employees where school_name = 'SAS';
```

Between comparison operator

Example : Find the names of all the employees with salary between 60,000 and 1,00,000

```
Select name from employees where salary between 60000 and 100000;
```

In operator : allows to specify multiple values in a where clause; shorthand for multiple or conditions

Example : Find the names of all the employees who are either from 'SCOPE', or 'SAS'

```
Select name from employees where school_name in
('SCOPE', 'SAS');
```

With the WHERE clause, the following operators can be used:

Operator	Description
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern

Note: In some versions of SQL the <> operator may be written as !=

Using Quotes

Note that we have used single quotes around the conditional values in the examples.

SQL uses single quotes around text values (most database systems will also accept double quotes). Numeric values should not be enclosed in quotes.

For text values:

This is correct:

```
SELECT * FROM Persons WHERE FirstName='Tove';
```

This is wrong:

```
SELECT * FROM Persons WHERE FirstName=Tove;
```

And /or

```
Select city from weather where humidity>60 and
humidity<80;
```

```
Select * from weather where temp!=80;
```

Can be used with the logical connectives.

Example : Find the names of all the employees of SAS school with salary greater than 60000

```
select name from employees where school_name = 'SAS' and
salary > 60000;
```

Order of clause

```
Select city from weather order by temp;
```

(by default it is ascending order otherwise we can put asc.)

```
Select city, temp from weather order by temp desc;
```

```
Select  * from weather order by humidity, temp desc;
```

```
Select * from weather where humidity > 70 order by temp;
```

Order of clause

select--------→from----------→where-------→order by

The LIKE Condition

The LIKE condition is used to specify a search for a pattern in a column.

```
SELECT column FROM table WHERE column LIKE pattern;
```

A "%" sign can be used to define wildcards (missing letters in the pattern) both before and after the pattern.

Using LIKE

The following SQL statement will return persons with first names that start with an 'O':

```
SELECT * FROM Persons WHERE FirstName LIKE 'O%';
```

The following SQL statement will return persons with first names that end with an 'a':

```
SELECT * FROM Persons WHERE FirstName LIKE '%a';
```

Using LIKE

The following SQL statement will return persons with first names that contain atleast 3 characters

```
SELECT * FROM Persons WHERE FirstName LIKE '___%';
```

The following SQL statement will return persons with first names that contain third character to be 's'

```
SELECT * FROM Persons WHERE FirstName LIKE '__s%';
```

Mismatching

- To find the mismatching characters we will use, not like operator.

```
Select * from weather where city not like '%ad%';
```

- We can use like operator to compare with numbers.

```
Select * from weather where temp like 50;
```

Null / not null

- Find the name of employees who do not get commission.

```
Select ename from emp where comm is null;
```

- Find the name of employees who get commission.

```
Select ename from emp where comm is not null;
```

```
Select ename from emp where mgr is null;
```