SQL SELECT – LIKE

Позиція SQL LIKE використовується для порівняння значення з аналогічними значеннями, використовуючи оператори підстановки. Є два символи підстановки, які використовуються спільно з оператором LIKE:

- * – замість одного або декількох символів (або %);
- ? – замість одного символу (або _).

The SQL LIKE clause is used to compare a value to similar values using wildcard operators. There are two wildcards used in conjunction with the LIKE operator:

- * – Matches one or more characters (or %).
- ? – Matches one character (or _)

SQL LIKE clause

```
SELECT
column1, column2, ..., columnN
FROM table_name
WHERE column_name LIKE { pattern };
```

employee_id:	first_name:	last_name:	position_id:	birth_date:	onboarding_
1	Jim	Halpert	2	5/12/1990	3/2/2014
2	Pamela	Beesly	1	11/2/1992	9/23/2015
3	Dwight	Schrute	3	2/11/1991	12/3/2013
4	Kelly	Kapoor	4	6/4/1993	3/2/2014
5	Michael	Scott	5	12/9/1989	4/28/2014

```
SELECT
first_name, last_name, onboarding_date
FROM employee
WHERE last_name LIKE "S*";
```

first_name:	last_name:	onboarding_date:
Dwight	Schrute	12/3/2013
Michael	Scott	4/28/2014

SQL SELECT – AND & OR

Оператори SQL AND та OR використовуються для об'єднання декількох умов для фільтрації даних у операторі SQL. Ці два оператори називаються кон'юнктивними операторами.

The SQL AND & OR operators are used to combine multiple conditions to narrow data in an SQL statement. These two operators are called as the conjunctive operators.

SQL AND/OR clause

```
SELECT
column1, column2, ..., columnN
FROM table_name
WHERE CONDITION1 { AND|OR }
   CONDITION 2;
```

employee_id:	first_name:	last_name:	position_id:	birth_date:	onboarding_
1	Jim	Halpert	2	5/12/1990	3/2/2014
2	Pamela	Beesly	1	11/2/1992	9/23/2015
3	Dwight	Schrute	3	2/11/1991	12/3/2013
4	Kelly	Kapoor	4	6/4/1993	3/2/2014
5	Michael	Scott	5	12/9/1989	4/28/2014

```
SELECT
first_name, last_name, onboarding_date
FROM employee
WHERE first_name = "Jim" OR first_name = "Kelly";
```

first_name:	last_name:	onboarding_date:
Jim	Halpert	3/2/2014
Kelly	Kapoor	3/2/2014

employee_id:	first_name:	last_name:	position_id:	birth_date:	onboarding_
1	Jim	Halpert	2	5/12/1990	3/2/2014
2	Pamela	Beesly	1	11/2/1992	9/23/2015
3	Dwight	Schrute	3	2/11/1991	12/3/2013
4	Kelly	Kapoor	4	6/4/1993	3/2/2014
5	Michael	Scott	5	12/9/1989	4/28/2014

```
SELECT
first_name, last_name, onboarding_date
FROM employee
WHERE department_id = 1 AND position_id = 3;
```

first_name:	last_name:	onboarding_date:
Dwight	Schrute	12/3/2013

SQL IN clause

```
SELECT
column1, column2, ..., columnN
FROM table_name
WHERE column_name IN (val-1, val-2,
   …, val-N);
```

SQL BETWEEN clause

```
SELECT
column1, column2, ..., columnN
FROM table_name
WHERE column_name BETWEEN val-1
  AND val-2;
```

SQL SELECT – GROUP BY

Позиція SQL GROUP BY використовується у співпраці з оператором SELECT для організації однакових даних у групи. Ця позиція GROUP BY слідує за позицією WHERE в операторі SELECT і передує позиції ORDER BY.

The SQL GROUP BY clause is used in collaboration with the SELECT statement to arrange identical data into groups. This GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.

SQL GROUP BY clause

```
SELECT
column1, column2, ..., columnN
FROM table_name
WHERE CONDITION
GROUP BY column_name;
```

project_id:	employee_i	from_date:	to_date:	role_id:
1	1	2/28/2016	12/18/2016	1
1	2	4/20/2016	12/18/2016	3
1	3	2/25/2016	12/15/2016	2
1	4	4/20/2016	12/19/2016	4
1	5	2/14/2016	12/19/2016	5
2	5	10/17/2016	4/22/2017	5
4	3	5/15/2017	12/31/2018	4
4	5	5/11/2017	12/31/2018	5
5	4	12/5/2017	2/20/2018	4

SELECT
employee_id, *COUNT*(project_id) AS [number_of_projects]
FROM resources_allocation
GROUP BY employee_id;

employee_id:	number_of_projects
1	1
2	1
3	2
4	2
5	3

Функції агрегації / Aggregate functions

Number	Function	Description
1	**Avg**	Calculates the arithmetic mean of a set of values contained in a specified field on a query. Середнє арифметичне.
2	**Count**	Calculates the number of records returned by a query. Кількість записів.
3	**First**	Return a field value from the first or last record in the result set returned by a query. Значення з першого/останнього запису.
4	**Last**	
5	**Min**	Return the minimum or maximum of a set of values contained in a specified field on a query. Максимальне/мінімальне значення.
6	**Max**	
7	**Sum**	Returns the sum of a set of values contained in a specified field on a query. Сума значень.

SQL COUNT clause

```
SELECT
COUNT(column_name)
FROM table_name
WHERE CONDITION;
```

project_id:	employee_i	from_date:	to_date:	role_id:
1	1	2/28/2016	12/18/2016	1
1	2	4/20/2016	12/18/2016	3
1	3	2/25/2016	12/15/2016	2
1	4	4/20/2016	12/19/2016	4
1	5	2/14/2016	12/19/2016	5
2	5	10/17/2016	4/22/2017	5
4	3	5/15/2017	12/31/2018	4
4	5	5/11/2017	12/31/2018	5
5	4	12/5/2017	2/20/2018	4

```
SELECT
employee_id, Sum(to_date - from_date) AS [days_on_projects]
FROM resources_allocation
GROUP BY employee_id;
```

employee_id:	days_on_projects
1	294
2	242
3	889
4	320
5	1095

SQL SELECT – HAVING

Позиція HAVING дозволяє визначити умови, які фільтрують групи, що з'являються у результатах.

Позиція WHERE містить умови для вибраних стовпців, тоді як позиція HAVING містить умови для груп, створених у пункті GROUP BY.

The HAVING Clause enables you to specify conditions that filter which group results appear in the results.

The WHERE clause places conditions on the selected columns, whereas the HAVING clause places conditions on groups created by the GROUP BY clause.

SQL HAVING clause

```
SELECT
SUM(column_name)
FROM table_name
WHERE CONDITION
GROUP BY column_name
HAVING CONDITION;
```

project_id:	employee_i	from_date:	to_date:	role_id:
1	1	2/28/2016	12/18/2016	1
1	2	4/20/2016	12/18/2016	3
1	3	2/25/2016	12/15/2016	2
1	4	4/20/2016	12/19/2016	4
1	5	2/14/2016	12/19/2016	5
2	5	10/17/2016	4/22/2017	5
4	3	5/15/2017	12/31/2018	4
4	5	5/11/2017	12/31/2018	5
5	4	12/5/2017	2/20/2018	4

```
SELECT
employee_id, Sum(to_date - from_date) AS [days_on_projects]
FROM resources_allocation
GROUP BY employee_id
HAVING Sum(to_date - from_date) >= 300;
```

employee_id:	days_on_projects
3	889
4	320
5	1095

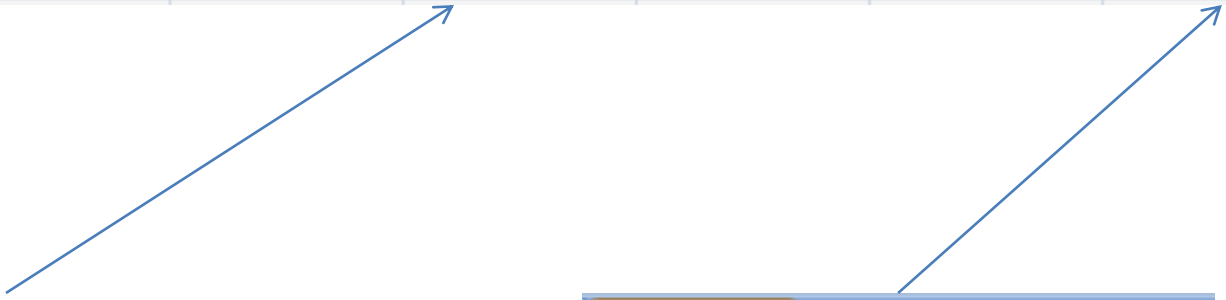
SQL SELECT – JOIN

Позиція SQL JOIN використовується для об'єднання записів з двох або більше таблиць у базі даних. JOIN використовується для об'єднання атрибутів з двох таблиць, використовуючи спільні для кожної таблиці значення.

The SQL JOIN clause is used to combine records from two or more tables in a database. A JOIN is a means for combining fields from two tables by using values common to each.

employee

employee_id:	first_name:	last_name:	position_id:	birth_date:	onboarding	department
1	Jim	Halpert	2	5/12/1990	3/2/2014	1
2	Pamela	Beesly	1	11/2/1992	9/23/2015	2
3	Dwight	Schrute	3	2/11/1991	12/3/2013	1
4	Kelly	Kapoor	4	6/4/1993	3/2/2014	3
5	Michael	Scott	5	12/9/1989	4/28/2014	4



position

position_id:	position_tit
1	QA Eng.
2	SW Eng.
3	Sr. SW Eng.
4	Syst. Eng.
5	PM

department

department	department_title:
1	Dev. Dept.
2	QA Dept.
3	Maintenance Dept.
4	Management Dept.
5	Accounting Dept.
6	HR Dept.

Query 01

```
SELECT
    employee.employee_id,
    employee.first_name,
    employee.last_name,
    position.position_title,
    department.department_title
FROM
   employee, position, department;
```

Query1

employee_id:	first_name:	last_name:	position_tit	department_title:
1	Jim	Halpert	QA Eng.	Dev. Dept.
2	Pamela	Beesly	QA Eng.	Dev. Dept.
3	Dwight	Schrute	QA Eng.	Dev. Dept.
4	Kelly	Kapoor	QA Eng.	Dev. Dept.
5	Michael	Scott	QA Eng.	Dev. Dept.
1	Jim	Halpert	SW Eng.	Dev. Dept.
2	Pamela	Beesly	SW Eng.	Dev. Dept.
3	Dwight	Schrute	SW Eng.	Dev. Dept.
4	Kelly	Kapoor	SW Eng.	Dev. Dept.
5	Michael	Scott	SW Eng.	Dev. Dept.
1	Jim	Halpert	Sr. SW Eng.	Dev. Dept.
2	Pamela	Beesly	Sr. SW Eng.	Dev. Dept.
3	Dwight	Schrute	Sr. SW Eng.	Dev. Dept.
4	Kelly	Kapoor	Sr. SW Eng.	Dev. Dept.
5	Michael	Scott	Sr. SW Eng.	Dev. Dept.
1	Jim	Halpert	Syst. Eng.	Dev. Dept.
2	Pamela	Beesly	Syst. Eng.	Dev. Dept.
3	Dwight	Schrute	Syst. Eng.	Dev. Dept.
4	Kelly	Kapoor	Syst. Eng.	Dev. Dept.
5	Michael	Scott	Syst. Eng.	Dev. Dept.
1	Jim	Halpert	PM	Dev. Dept.
2	Pamela	Beesly	PM	Dev. Dept.
3	Dwight	Schrute	PM	Dev. Dept.
4	Kelly	Kapoor	PM	Dev. Dept.
5	Michael	Scott	PM	Dev. Dept.

Record: 1 of 150 No Filter Search

Query result:
150 records (!!!)
Instead of 5

WHY?

Декартове з’єднання / Cartesian or cross join

Декартове з’єднання повертає декартов добуток з наборів записів з двох або більше об'єднаних таблиць. Таким чином, він прирівнюється до внутрішнього з'єднання, де умова приєднання завжди оцінюється як істина, або де умова з'єднання відсутня у твердженні.

The CARTESIAN JOIN or CROSS JOIN returns the Cartesian product of the sets of records from two or more joined tables. Thus, it equates to an inner join where the join-condition always evaluates to either True or where the join-condition is absent from the statement.