Синтаксис SQL / SQL syntax

Усі оператори SQL починаються з будь-якого з ключових слів, таких як SELECT, INSERT, UPDATE, DELETE, ALTER, DROP, CREATE, USE, SHOW, і всі твердження закінчуються крапкою з комою (;).

SQL є нечутливим до регістру, тобто SELECT і select мають однакове значення в операторах SQL.

All the SQL statements start with any of the keywords like SELECT, INSERT, UPDATE, DELETE, ALTER, DROP, CREATE, USE, SHOW and all the statements end with a semicolon (;).

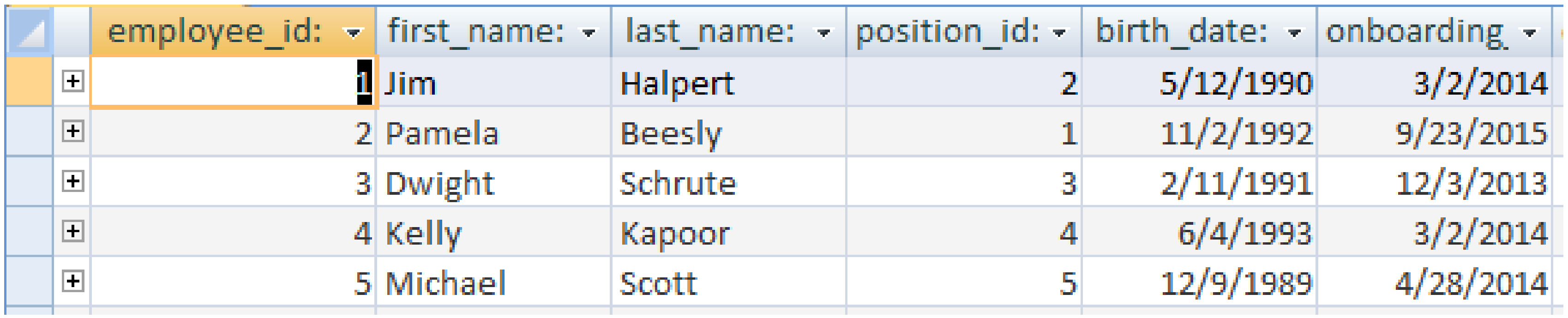
SQL is case insensitive, which means SELECT and select have same meaning in SQL statements.

SQL SELECT

SELECT
    [**DISTINCT** | **ALL** | **TOP** {unsigned_integer}]
    select_expression,...
FROM table_references
[**WHERE** where_definition]
[**GROUP BY** {unsigned_integer | col_name |
    formula}]
[**HAVING** where_definition]
[**ORDER BY** {unsigned_integer | col_name |
    formula} [**ASC** | **DESC**], ...]

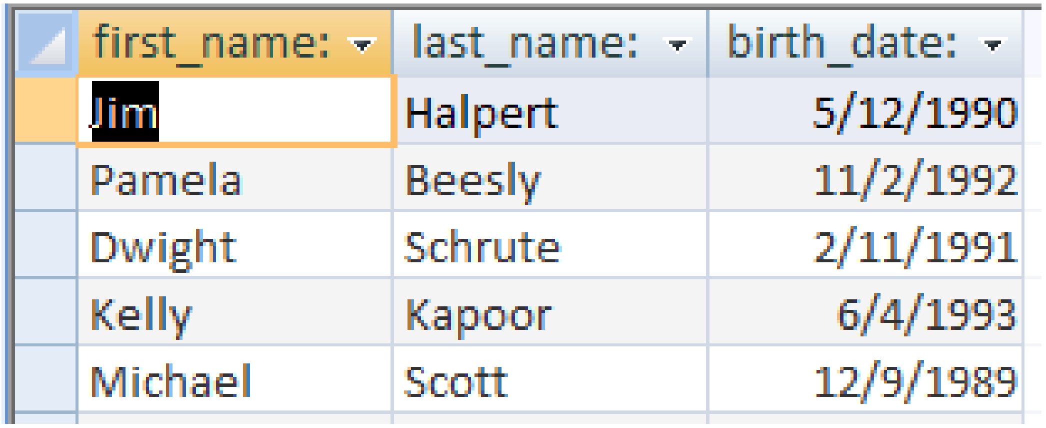
SQL SELECT statement

```
SELECT
column1, column2, ..., columnN
FROM table_name;
```

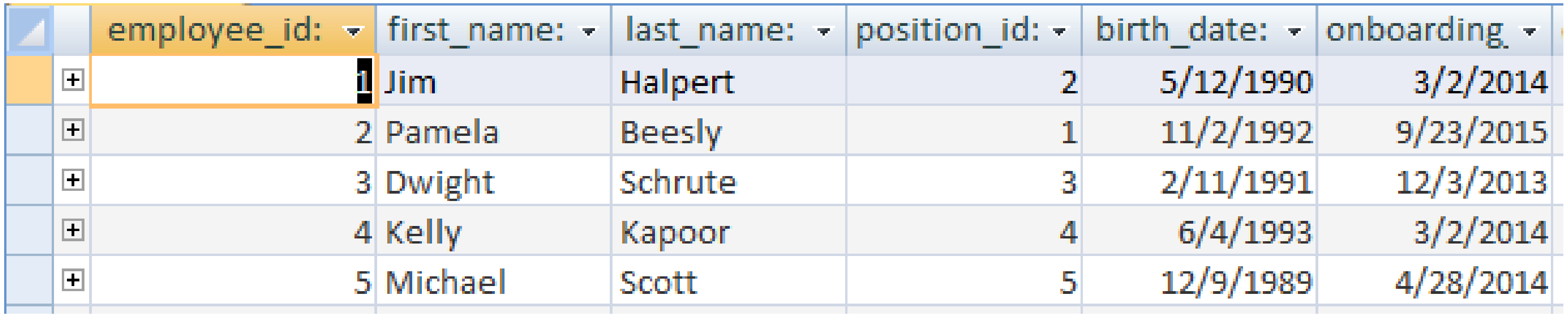


employee_id:	first_name:	last_name:	position_id:	birth_date:	onboarding
1	Jim	Halpert	2	5/12/1990	3/2/2014
2	Pamela	Beesly	1	11/2/1992	9/23/2015
3	Dwight	Schrute	3	2/11/1991	12/3/2013
4	Kelly	Kapoor	4	6/4/1993	3/2/2014
5	Michael	Scott	5	12/9/1989	4/28/2014

```
SELECT
first_name, last_name, birth_date
FROM employee;
```

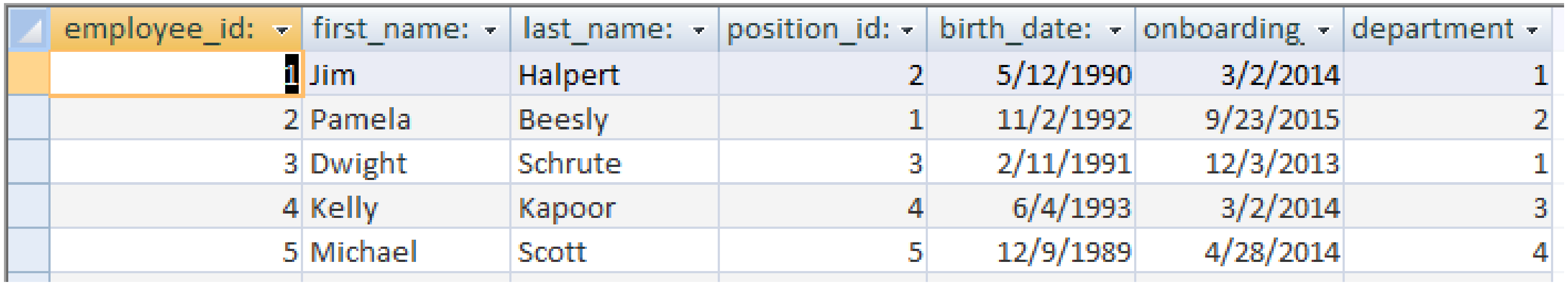


first_name:	last_name:	birth_date:
Jim	Halpert	5/12/1990
Pamela	Beesly	11/2/1992
Dwight	Schrute	2/11/1991
Kelly	Kapoor	6/4/1993
Michael	Scott	12/9/1989



	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
*
FROM 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

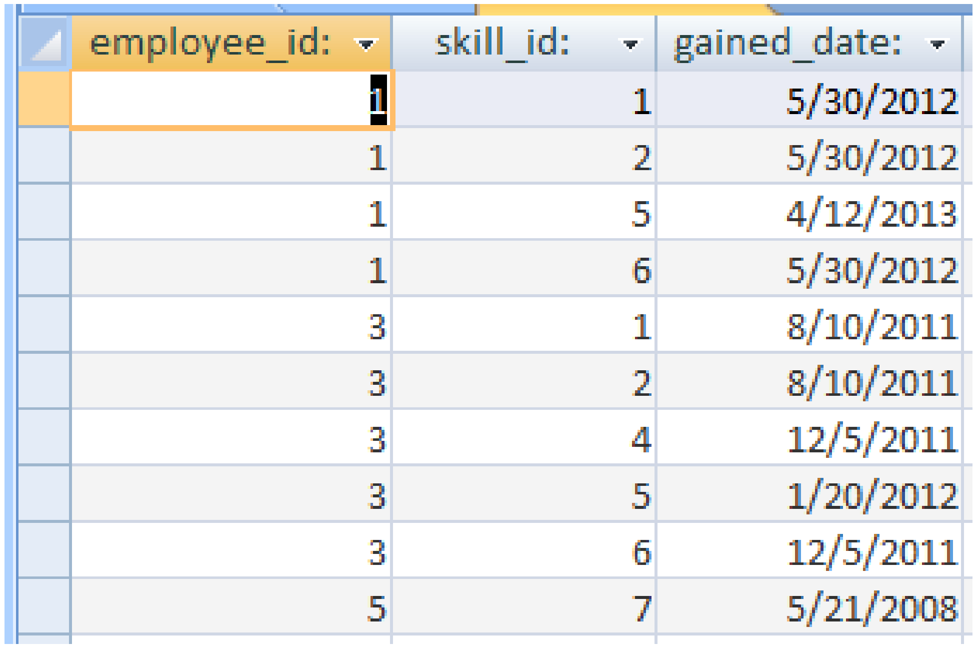
SQL SELECT – DISTINCT

Ключове слово SQL DISTINCT використовується разом із оператором SELECT для виключення всіх дубльованих записів та отримання лише унікальних записів.

The SQL DISTINCT keyword is used in conjunction with the SELECT statement to eliminate all the duplicate records and fetching only unique records.

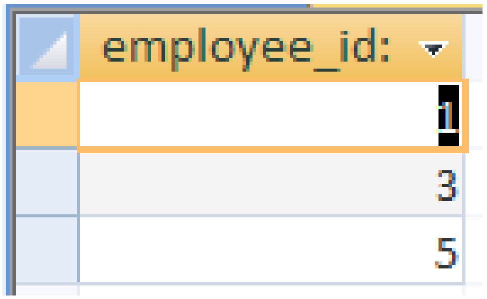
SQL DISTINCT clause

```
SELECT DISTINCT
column1, column2, ..., columnN
FROM table_name;
```



employee_id:	skill_id:	gained_date:
1	1	5/30/2012
1	2	5/30/2012
1	5	4/12/2013
1	6	5/30/2012
3	1	8/10/2011
3	2	8/10/2011
3	4	12/5/2011
3	5	1/20/2012
3	6	12/5/2011
5	7	5/21/2008

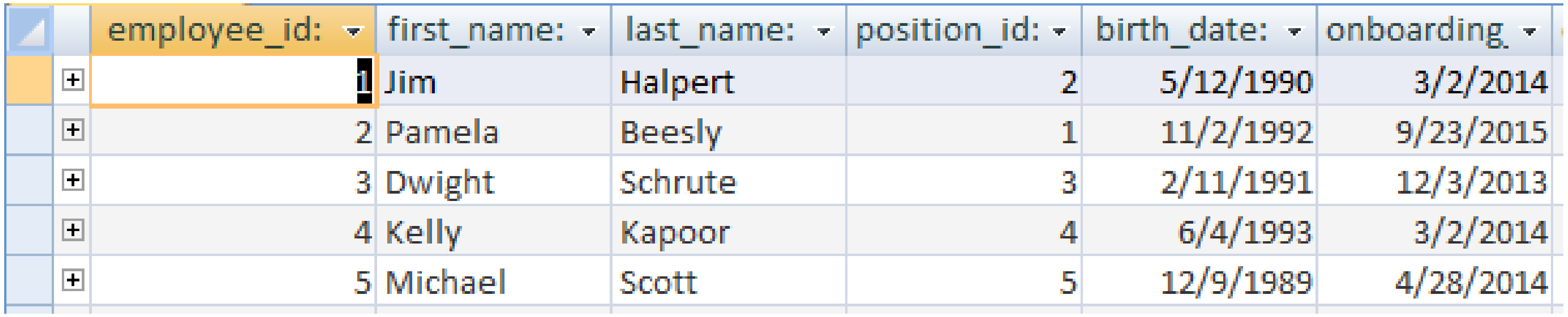
SELECT **DISTINCT** employee_id FROM skills_matrix;



employee_id:
1
3
5

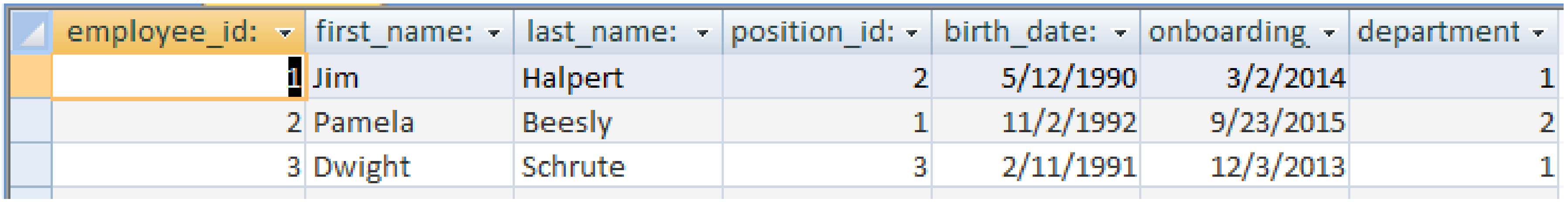
ЯКИЙ БУДЕ РЕЗУЛЬТАТ ЯКЩО ЗАМІСТЬ **DISTINCT** БУДЕ **ALL**?

WHAT WILL BE THE OUTPUT IF WE USE THE KEYWORD **ALL** INSTEAD OF **DISTINCT**?



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 **TOP 3**
*
FROM 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

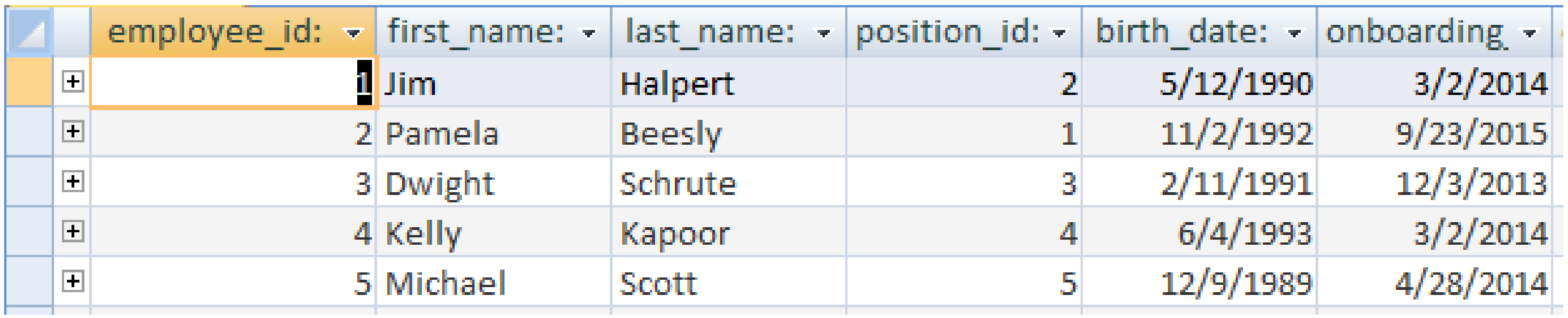
SQL SELECT – ORDER BY

За замовчуванням SQL ORDER BY використовується для сортування даних в порядку зростання або зменшення на основі одного або декількох стовпців. Деякі бази даних сортують результати запиту за зростанням.

The SQL ORDER BY clause is used to sort the data in ascending or descending order, based on one or more columns. Some databases sort the query results in an ascending order by default.

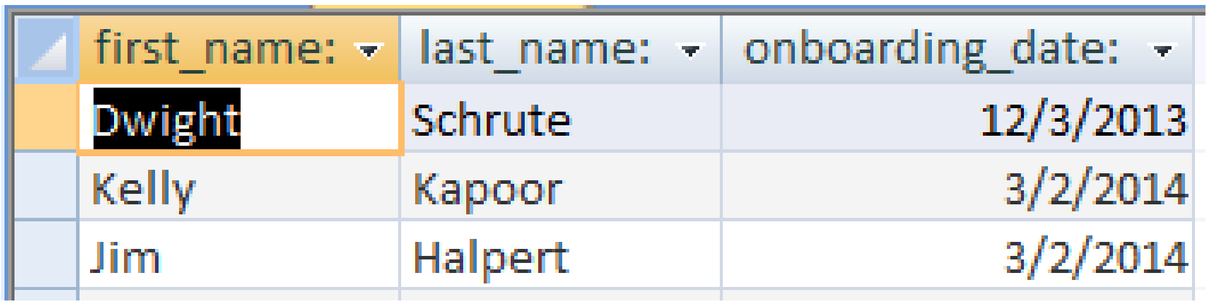
SQL ORDER BY clause

```
SELECT
column1, column2, ..., columnN
FROM table_name
ORDER BY column_name {ASC | DESC};
```

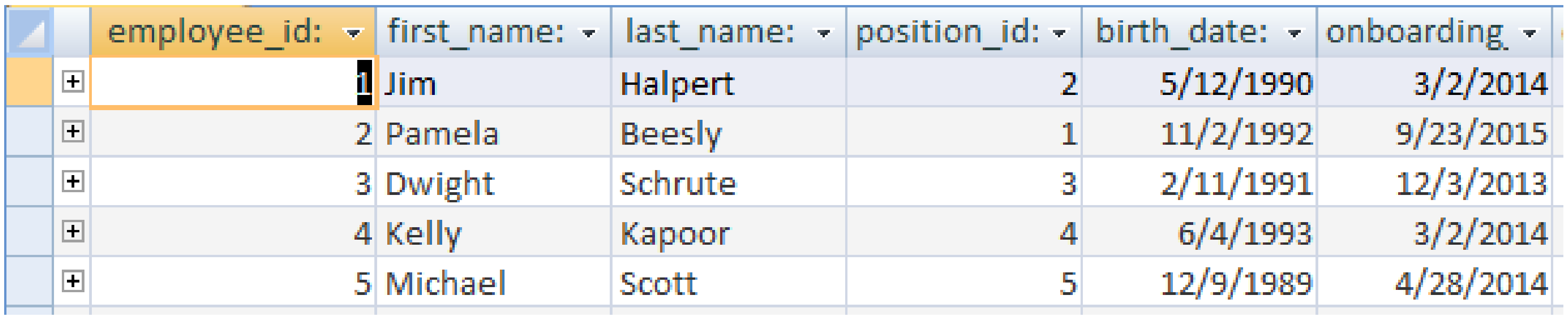


employee_id:	first_name:	last_name:	position_id:	birth_date:	onboarding
1	Jim	Halpert	2	5/12/1990	3/2/2014
2	Pamela	Beesly	1	11/2/1992	9/23/2015
3	Dwight	Schrute	3	2/11/1991	12/3/2013
4	Kelly	Kapoor	4	6/4/1993	3/2/2014
5	Michael	Scott	5	12/9/1989	4/28/2014

SELECT TOP 3
first_name, last_name, onboarding_date
FROM employee
ORDER BY onboarding_date;

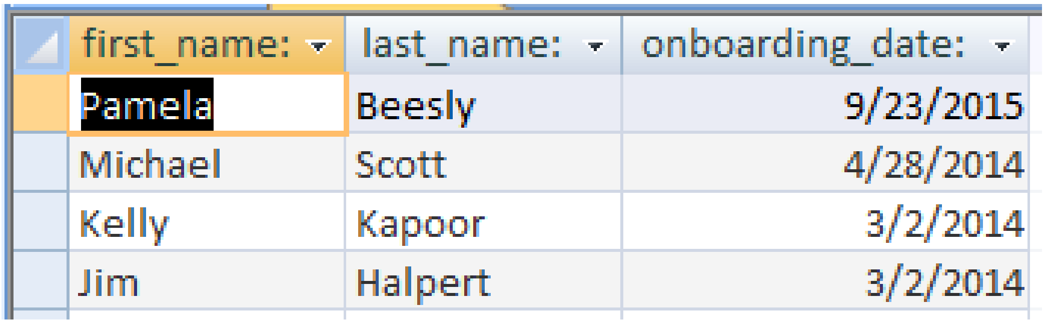


first_name:	last_name:	onboarding_date:
Dwight	Schrute	12/3/2013
Kelly	Kapoor	3/2/2014
Jim	Halpert	3/2/2014



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 TOP 3
first_name, last_name, onboarding_date
FROM employee
ORDER BY onboarding_date **DESC**;



first_name:	last_name:	onboarding_date:
Pamela	Beesly	9/23/2015
Michael	Scott	4/28/2014
Kelly	Kapoor	3/2/2014
Jim	Halpert	3/2/2014

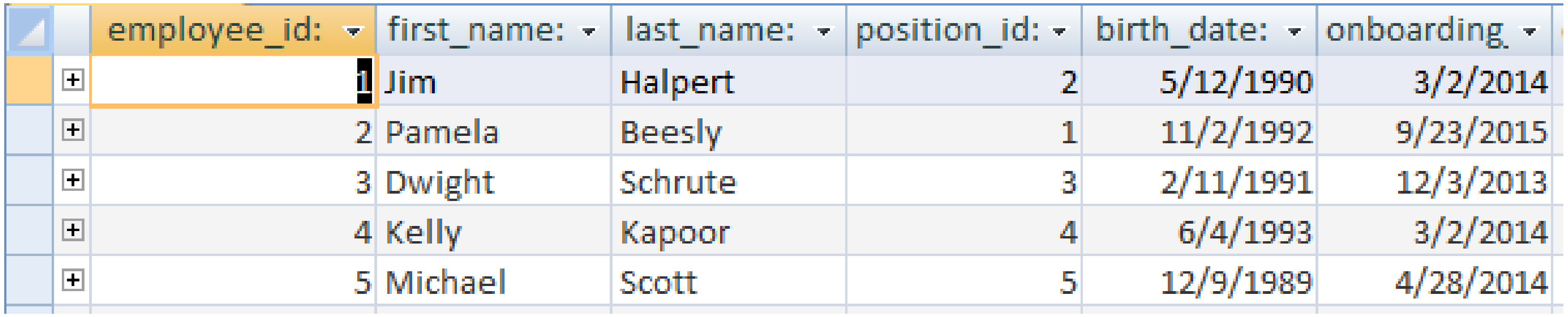
SQL SELECT – WHERE

Позиція SQL WHERE використовується для вказування умови для отримання даних з однієї таблиці або шляхом об'єднання з кількома таблицями. Якщо ця умова виконується, то тільки вона повертає певне значення з таблиці. WHERE використовують щоб відфільтрувати записи та отримувати лише необхідні записи.

The SQL WHERE clause is used to specify a condition while fetching the data from a single table or by joining with multiple tables. If the given condition is satisfied, then only it returns a specific value from the table. WHERE is used to filter the records and fetching only the necessary records.

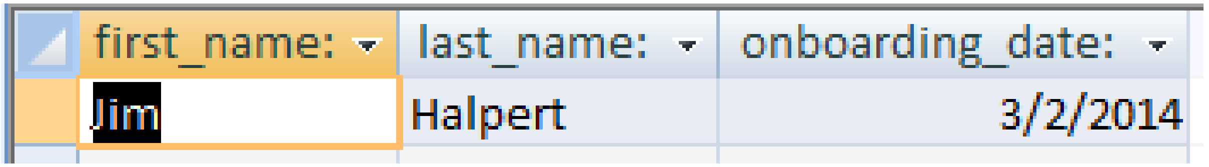
SQL WHERE clause

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



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"



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

SQL – Operators

Оператор є зарезервованим словом або символом, який використовується, перш за все, в позиції WHERE оператора SQL для виконання операцій, таких як порівняння, та арифметичних операцій. Ці оператори використовуються для того, щоб вказувати умови в операторі SQL та виступати в якості сполучень для декількох умов у твердженні.

An operator is a reserved word or a character used primarily in an SQL statement's WHERE clause to perform operation(s), such as comparisons and arithmetic operations. These Operators are used to specify conditions in an SQL statement and to serve as conjunctions for multiple conditions in a statement.

SQL arithmetic operators

Operator	Description	Example
+ (Addition)	Adds values on either side of the operator.	a + b will give 30
- (Subtraction)	Subtracts right hand operand from left hand operand.	a - b will give -10
* (Multiplication)	Multiplies values on either side of the operator.	a * b will give 200
/ (Division)	Divides left hand operand by right hand operand.	b / a will give 2
% (Modulus)	Divides left hand operand by right hand operand and returns remainder.	b % a will give 0

SQL comparison operators (1)

Operator	Description	Example
=	Checks if the values of two operands are equal or not, if yes then condition becomes true.	(a = b) is not true.
!=	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(a != b) is true.
<>	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(a <> b) is true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(a > b) is not true.
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(a < b) is true.

SQL comparison operators (2)

Operator	Description	Example
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(a >= b) is not true.
<=	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(a <= b) is true.
!<	Checks if the value of left operand is not less than the value of right operand, if yes then condition becomes true.	(a !< b) is false.
!>	Checks if the value of left operand is not greater than the value of right operand, if yes then condition becomes true.	(a !> b) is true.

SQL logical operators (1)

No.	Operator & Description
1	**ALL** The ALL operator is used to compare a value to all values in another value set.
2	**AND** The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause.
3	**ANY** The ANY operator is used to compare a value to any applicable value in the list as per the condition.
4	**BETWEEN** The BETWEEN operator is used to search for values that are within a set of values, given the minimum value and the maximum value.
5	**EXISTS** The EXISTS operator is used to search for the presence of a row in a specified table that meets a certain criterion.

SQL logical operators (2)

No.	Operator & Description
6	**IN** The IN operator is used to compare a value to a list of literal values that have been specified.
7	**LIKE** The LIKE operator is used to compare a value to similar values using wildcard operators.
8	**NOT** The NOT operator reverses the meaning of the logical operator with which it is used.
9	**OR** The OR operator is used to combine multiple conditions in an SQL statement's WHERE clause.
10	**IS NULL** The NULL operator is used to compare a value with a NULL value.
11	**UNIQUE** The UNIQUE operator searches every row of a specified table for uniqueness (no duplicates).