

VSB – Technical University of Ostrava
Faculty of Electrical Engineering
Department of Computer Science
Database Research Group



Introduction to Database Systems

Tutorial 3

Ing. Petr Lukáš
petr.lukas@vsb.cz
EA440
Ostrava, 2017

SQL

Structured Query Language



Student

login	f_name	l_name	date_of_birth
bla005	Adam	Black	5 Sep, 1987
whi007	Bob	White	7 Jun, 1988
gra065	John	Gray	2 Aug, 1991

Select all Students.

Select all students born after 1 Jan, 1988.

Select first and last names of all students.

Select all students sorted by their date of birth in a descending order.



Student

login	f_name	l_name	date_of_birth
bla005	Adam	Black	5 Sep, 1987
whi007	Bob	White	7 Jun, 1988
gra065	John	Gray	2 Aug, 1991

Select all Students.

```
SELECT *  
FROM Student
```

Select all students born after 1 Jan, 1988.

```
SELECT *  
FROM Student  
WHERE date_of_birth > '1988-01-01'
```

Select first and last names of all students.

```
SELECT f_name, l_name  
FROM Student
```

Select all students sorted by their date of birth in a descending order.

```
SELECT *  
FROM Student  
ORDER BY date_of_birth DESC
```

Student

login	f_name	univerzity
bla005	Adam	1
whi007	Bob	1
gra065	John	2

Univerzity

id	name
1	VSB – TUO
2	University of Ostrava



Select all names of students and the universities they study.



Student

login	f_name	univerzity
bla005	Adam	1
whi007	Bob	1
gra065	John	2

Univerzity

id	name
1	VSB – TUO
2	University of Ostrava



Select all names of students and the universities they study.

```
SELECT f_name, name  
FROM Student, University  
WHERE Student.univerzity = University.id
```



f_name	name
Adam	VSB – TUO
Bob	VSB – TUO
John	University of Ostrava



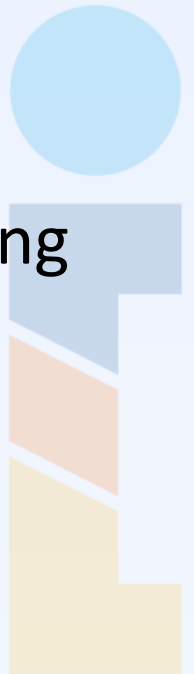
- **Boolean formulas**

We use traditional connectives **AND**, **OR**, **NOT** for conjunction, disjunction and negation, respectively.

- **Comparison operators**

Equality test **=** (do not use **==**). The common meaning have the symbols **<**, **>**, **<>** or **!=**.

We have a special test checking a NULL value – expressions **IS NULL** or **IS NOT NULL**.



- **Arithmetics**

We use common operators $+$, $-$, $*$, $/$, $\%$ (modulo). Dividing two integer numbers gives also an integer number!

$+$ can be used also for concatenating two strings.

- **Strings**

The **'strings'** have to be written in apostrophes.

There is a special operator **LIKE** to compare a string with a regular expression, where $\%$ stands for an arbitrary number of arbitrary characters.

*eg., **name LIKE 'P%'** – all names starting with 'P'.*



- **SELECT Clause**

Immediately after the SELECT clause, we can write the following modifiers:

DISTINCT – removes the duplicate results.

```
SELECT DISTINCT f_name  
FROM Student
```

Select all first names of the students and do not output the same name twice.

TOP n – we are interested only in the first n results. We only want to preview the data, or we can combine this modifier with the ORDER BY clause.

```
SELECT TOP 1 login  
FROM Student  
ORDER BY date_of_birth
```

Return the login of the oldest student.



```
select TOP 10 f_name FROM STUDent,  
product  
    where l_NAME = 'White' ORDER  
BY  
date_of_birth
```

vs.

```
SELECT TOP 10 f_name  
FROM Student, Product  
WHERE l_name = 'Lukáš'  
ORDER BY date_of_birth
```

- Each **clause** begins on a separate line.
- For complicated queries, it is appropriate to write the clause keywords on separate lines.
- **Keywords** are written in capitals.
- **Names of tables and attributes** are always written as they have been created. E.g., we have a table Student, we will not write STUDENT.
- If we are not sure about a priority of an operator, we use **brackets**.



- **IN, EXISTS, ALL, ANY**
- **Subqueries**



Product

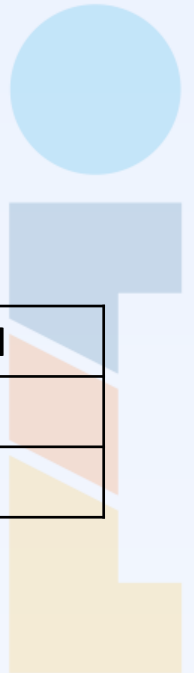
p_id	name
1	computer
2	display
3	keyboard
4	mouse

Select IDs of products with the name 'display' or 'mouse'.

```
SELECT p_id
FROM Product
WHERE name = 'display' OR name = 'mouse'
```



p_id
2
4



Product

p_id	name
1	computer
2	display
3	keyboard
4	mouse

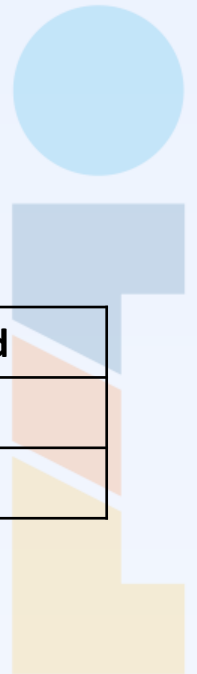
Select IDs of products with the name 'display' or 'mouse'.

```
SELECT p_id  
FROM Product  
WHERE name = 'display' OR name = 'mouse'
```

```
SELECT p_id  
FROM Product  
WHERE name IN ('display', 'mouse')
```



p_id
2
4



Product

p_id	name
1	computer
2	display
3	keyboard
4	mouse

Order_Items

order	p_id	amount
2017-001	1	2
2017-001	3	3
2017-002	1	5
2017-002	2	4
2017-003	4	5

Select names of products of the order '2017-001'.

```
SELECT name
FROM Product
WHERE p_id IN (
  SELECT p_id
  FROM Order_Items
  WHERE order = '2017-001'
)
```



name
computer
keyboard

Product

p_id	name
1	computer
2	display
3	keyboard
4	mouse

Order_Items

order	p_id	amount
2017-001	1	2
2017-001	3	3
2017-002	1	5
2017-002	2	4
2017-003	4	5

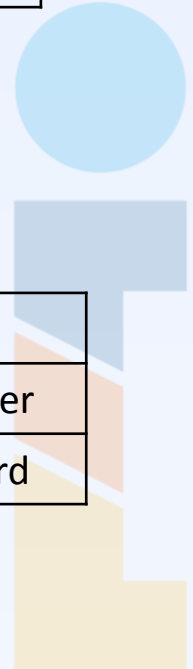
Select names of products of the order '2017-001'.

```

SELECT name
FROM Product
WHERE EXISTS (
  SELECT *
  FROM Order_Items
  WHERE order = '2017-001' AND
    Order_Items.p_id = Product.p_id
)
    
```



name
computer
keyboard



Product

p_id	name	store_amount
1	computer	3
2	display	4
3	keyboard	3
4	mouse	3

Order_Items

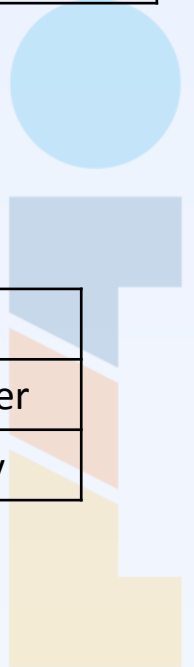
order	p_id	amount
2017-001	1	2
2017-001	3	3
2017-002	1	5
2017-002	2	4
2017-003	4	5

Select product names where the store amount is less than any amount of the product in order items.

```
SELECT name
FROM Product
WHERE store_amount < ANY (
  SELECT amount
  FROM Order_Items
  WHERE Order_Items.p_id = Product.p_id
)
```



name
computer
display



Product

p_id	name	store_amount
1	computer	3
2	display	4
3	keyboard	3
4	mouse	3

Order_Items

order	p_id	amount
2017-001	1	2
2017-001	3	3
2017-002	1	5
2017-002	2	4
2017-003	4	5

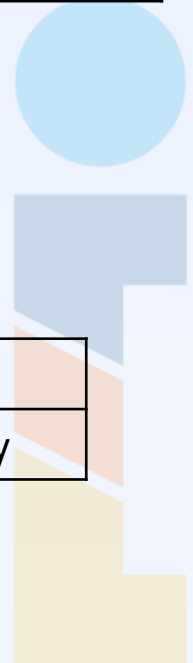
Select product names where the store amount is less than all amounts of the product in order items.

```

SELECT name
FROM Product
WHERE store_amount < ALL (
  SELECT amount
  FROM Order_Items
  WHERE Order_Items.p_id = Product.p_id
)
    
```



name
display



- **IN** We ask if a value is in some set.
- **EXISTS** We test the presence of a record
- **ANY** We compare a value with a set of values, the comparison must be satisfied at least for one value of the set.
- **ALL** We compare a value with a set of values, the comparison must be satisfied for all values of the set.



www.dbedu.cs.vsb.cz

- LDAP **login** and **password**
- ***English Courses -> IDS***

