

# Jazyk SQL – dotazy

Jiří Zaccpal



KATEDRA INFORMATIKY  
UNIVERZITA PALACKÉHO V OLOMOUCI

KMI/DATAB Databáze

# Studenti



Proveďte příkazy ze souboru [datab\\_02\\_dotazy\\_vstup.sql](#)

id_student integer	jmeno text	prijmeni text	rocnik integer	email text
1	Jan	Novák	1	jan.novak@upol.cz
2	Pavla	Dočkalová	1	pavla.dockalova@upol.cz
3	Karel	Bártek	2	karel.bartek@upol.cz
4	Václav	Snášel	3	vaclav.snasel@upol.cz
5	Iva	Nová	2	iva.nova@seznam.cz

# Příklad



- Chtěli bychom získat seznam studentů z tabulky studenti.

# Příkaz SELECT

- slouží k výběru záznamů
- základní syntaxe:

```
SELECT sloupce  
FROM jmeno_tabulky  
[WHERE where_fráze ];
```

# Příklad

- Chtěli bychom získat seznam studentů z tabulky studenti:

```
SELECT *
```

```
FROM studenti;
```

- Stačí nám jen jméno, příjmení a email:

```
SELECT jmeno,prijmeni,email
```

```
FROM studenti;
```

- Nešlo by jméno a příjmení spojit do jednoho sloupce?

# Vypočítané sloupce a aliasy

- obecný zápis pro přejmenování sloupce:

```
SELECT jmeno_sloupce AS alias_name  
FROM table_name;
```

- obecný zápis pro vytvoření nového sloupce:

```
SELECT výraz AS alias_name  
FROM table_name;
```

# Příklad

- Vytvoříme sloupec cele\_jmeno:

```
SELECT jmeno || prijmeni AS cele_jmeno, email  
FROM studenti;
```

- Doplníme mezeru mezi jménem a příjmením:

```
SELECT jmeno || ' ' || prijmeni AS cele_jmeno, email  
FROM studenti;
```

- Vše spojíme do jednoho sloupce:

```
SELECT 'Student ' || jmeno || ' ' || prijmeni || ' má e-mail:  
' || email AS kontakt  
FROM studenti;
```

# Příklad



- Chtěli bychom zjistit počáteční rok studia:

```
SELECT jmeno || ' ' || prijmeni AS cele_jmeno, 2024 -  
rocnik AS pocatecni_rok  
FROM studenti;
```

- Šlo by výpočet zobecnit pro aktuální rok?

```
SELECT jmeno || ' ' || prijmeni AS cele_jmeno,  
date_part('year',now()) + 1 - rocnik AS pocatecni_rok  
FROM studenti;
```



# Příklad

- Vraťme se k poslednímu úkolu:

```
SELECT semestr  
FROM predmety;
```

- Chtěli bychom zobrazit každou hodnotu pouze jednou.
- Použijeme příkaz **SELECT DISTINCT**

```
SELECT DISTINCT semestr  
FROM predmety;
```

semestr text
ZS
LS
ZS
LS

semestr text
ZS
LS

# Klauzule WHERE

- slouží k výběru řádků,
- základní syntaxe:

SELECT sloupce

FROM jmeno\_tabulky

WHERE podmínka;

- vrátí ty řádky, které splňují podmínku.

# Příklad

- Z tabulky studenti vybereme řádky, které mají ve sloupci jmeno hodnotu „Karel“:

```
SELECT *  
FROM studenti  
WHERE jmeno='Karel';
```

- Vybereme studenty 1 ročníku:

```
SELECT *  
FROM studenti  
WHERE rocnik=1;
```

- Vybereme studenty prvního ročníku a zobrazíme jen jejich jmena a prijmeni:

```
SELECT jmeno, prijmeni  
FROM studenti  
WHERE rocnik=1;
```

# Operátory v klauzuli WHERE

- porovnání

<, <=, >, >=, =, <> nebo !=

- aritmetické

+, -, \*, /, ^

- logické

AND, OR, NOT

# Příklad

- Chceme vybrat studenty, kteří nejsou ze 3. ročníku:

```
SELECT *  
FROM studenti  
WHERE rocnik!=3;
```

- Nebo:

```
SELECT *  
FROM studenti  
WHERE rocnik<3;
```

# Příklad

- Jinou možností je toto (když víme, že ročníky jsou jen 1, 2 s 3):

```
SELECT *  
FROM studenti  
WHERE rocnik=1 OR rocnik=2;
```

- Co se ale stane, když někdo nebude mít ročník vyplněn?

```
INSERT INTO studenti(id_student,jmeno,prijmeni)  
VALUES(6, 'Alois', 'Musil');
```

- Hodnotu NULL nelze porovnávat pomocí operátorů porovnání:

```
SELECT *  
FROM studenti  
WHERE rocnik=NULL;
```

# Operátor IS NULL

IS NULL

- hodnota atributu je NULL

IS NOT NULL

- hodnota atributu není NULL

# Příklad

- Vypíšeme studenty, kteří nemají nastavený email:

```
SELECT *  
FROM studenti  
WHERE email IS NULL;
```

- Vypíšeme studenty, kteří mají nastavený email:

```
SELECT *  
FROM studenti  
WHERE email IS NOT NULL;
```



# Příklad

- Chceme zobrazit studenty, kteří mají zapsaný předmět číslo 2 a současně ho úspěšně ukončili:

```
SELECT student  
FROM zapsani  
WHERE predmet=2 AND uspel;
```

- Dotaz upravíme tak, aby zobrazil studenty, kteří mají zapsaný předmět číslo 2 nebo úspěšně ukončili jakýkoliv předmět:

```
SELECT student  
FROM zapsani  
WHERE predmet=2 OR uspel;
```

- Jak je to s hodnotou NULL u logických operátorů?

# Logické operátory a hodnota NULL

- V tomto případě mluvíme o tříhodnotové logice.
- Výsledek operace se řídí těmito pravdivostními tabulkami:

AND	t	f	null	OR	t	f	null	NOT	
t	t	f	null	t	t	t	t	t	f
f	f	f	f	f	t	f	null	f	t
null	null	f	null	null	t	null	null	null	null

# Příklad



- Chtěli bychom zobrazit studenty, kteří mají e-mail z domény upol.

# Operátor LIKE

- LIKE vzor
  - hodnota atributu je rovna řetězcovému vzoru
  - vzor se dá do apostrofů
  - lze v něm použít speciální znaky:
    - % - nahrazuje libovolný počet znaků
    - \_ - nahrazuje libovolný jeden znak

# Příklad

- Chtěli bychom zobrazit studenty, kteří mají e-mail z domény upol:

```
SELECT jmeno, prijmeni, email  
FROM studenti  
WHERE email LIKE '%upol____';
```

- Chtěli bychom zobrazit studenty, jejichž jméno obsahuje znak ,v':

```
SELECT jmeno  
FROM studenti  
WHERE jmeno LIKE '%v%'
```

- Chtěli bychom zobrazit studenty, jejichž jméno končí na znak ,v':

```
SELECT jmeno  
FROM studenti  
WHERE jmeno LIKE '%v';
```

# Smazání záznamů

- obecná syntaxe příkazu DELETE:

```
DELETE FROM jmeno  
[WHERE podmínka]
```

- Příklad:

```
DELETE FROM Osoby
```

```
DELETE FROM Osoby WHERE stav='nestuduje'
```

# Příklad

- Chceme smazat z tabulky zapsani všechny studenty, kteří předmět již úspěšně zvládli:

```
DELETE FROM zapsani  
WHERE uspel=TRUE;
```

- Chceme smazat z tabulky zapsani všechny studenty:

```
DELETE FROM zapsani;
```