Relační databáze v Pythonu

Vladimír Župka, 2024

Oblíbené relační databáze

SQLite
MySQL
PostgreSQL

Příklad ze zásobování

Tabulka STAVY

ZAVOD	číslo závodu (textově)
SKLAD	číslo skladu (textově)
MATER	číslo materiálu (textově)
MNOZ	množství materiálu na skladě (zlomkové číslo)

Tabulka CENY

MATER	číslo materiálu (textově)
CENAJ	cena za jednotku materiálu (zlomkové číslo)
NAZEV	označení materiálu (textově)

Databáze SQLite – nemá server

je integrovaná v Pythonu pod názvem ***sqlite3***

Typy dat

SQLite:	INTEGER	REAL	TEXT	BLOB	NULL
Python:	**int**	**float**	**str**	**bytes**	**None**

```
import sqlite3
```

Vytvoření a připojení databáze (schematu)

```
connection = sqlite3.connect('sqlite3_zasoby.db')
                            # objekt třídy sqlite3.Connection
```

Vytvoření kurzoru

```
cursor = connection.cursor()  # objekt třídy sqlite3.Cursor
```

Vytvoření tabulek

```
cursor.execute("""
CREATE TABLE stavy
             (zavod TEXT, sklad TEXT, mater TEXT, mnoz REAL)""")
```

Zápis dat do tabulky STAVY jednotlivě

```
cursor.execute("INSERT INTO stavy VALUES ('01','01','40001',0)")
cursor.execute("INSERT INTO stavy VALUES ('01','02','40002',100.05)")
cursor.execute("INSERT INTO stavy VALUES ('02','01','40003',100.05)")
cursor.execute("INSERT INTO stavy VALUES ('02','02','40004',100.05)")

cursor.execute("COMMIT")  # potvrzení zápisu do tabulky
```

Zápis dat do tabulky CENY hromadně

*Hodnoty dat jsou v **seznamu n-tic** (trojic sloupců).*

```
cursor.execute("CREATE TABLE ceny (mater, cenaj, nazev)")
               # Nejsou určeny typy dat. Určí se při zápisu dat.

# ? je značka místa, kam patří příslušná hodnota sloupce
cursor.executemany("""INSERT INTO ceny (mater, cenaj, nazev)
VALUES (?, ?, ?)""",
                   [("40001",1.00,"Materiál 1"),
                    ("40002",2.01,"Materiál 2"),
                    ('40003',3.01,'Materiál 3'),
                    ('40004',4.01,'Mater')])

connection.commit()  # alternativa k SQL příkazu COMMIT
```

Dotaz a jeho vyhodnocení

Vyhodnocení příkazem *fetchall*

```
dotaz = '''SELECT zavod, sklad, S.mater, nazev, cenaj, mnoz,
                  round(cenaj*mnoz, 2) as celkem
    FROM stavy as S
    JOIN ceny as C on S.mater = C.mater
    ORDER BY mnoz'''

cursor.execute(dotaz)  # provede dotaz

result_set = cursor.fetchall()  # vytvoří seznam n-tic

for zaznam in result_set:
    print(zaznam)  # otisk záznamů jako n-tic
```

```
('01', '01', '40001', 'Materiál 1', 1.0, 0.0, 0.0)
('01', '02', '40002', 'Materiál 2', 2.01, 100.05, 201.1)
('02', '01', '40003', 'Materiál 3', 3.01, 100.05, 301.15)
('02', '02', '40004', 'Mater', 4.01, 100.05, 401.2)
```

Vyhodnocení příkazem *fetchone*

```
cursor.execute(dotaz)  # provede dotaz (viz výše)

zaznam = cursor.fetchone()  # přečte první záznam výsledné tabulky
while zaznam:
    radek = ''
    for col in zaznam:
        if type(col) == str:
            radek += f'"{col}"'  # přidá dvojité uvozovky
        else:
            # výplň *, šířka 10 míst od konce vlevo, 2 d.m., float
            radek += f'{col:*>10.2f}'
        radek += ', '
    print(radek)
    zaznam = cursor.fetchone()  # přečte další záznam výsledné tabulky
```

```
"01", "01", "40001", "Materiál 1", ******1.00, ******0.00, ******0.00,
"01", "02", "40002", "Materiál 2", ******2.01, ****100.05, ****201.10,
"02", "01", "40003", "Materiál 3", ******3.01, ****100.05, ****301.15,
"02", "02", "40004", "Mater", ******4.01, ****100.05, ****401.20,
```

Převod z tabulky do CSV souboru

```
cursor.execute('SELECT * FROM ceny')

# příkaz with-as zajišťuje automatické uzavření souboru
with open("sqlite3_ceny.csv", encoding="utf-8", mode="w") as outfile:
    vysl_tabulka = cursor.fetchall()  # provede příkaz SELECT
    for zaznam in vysl_tabulka:
        radek = ''
        for index, col in enumerate(zaznam, start=0):
            if index != 0:
                radek += ','
            if type(col) == str:
                radek += repr(col)  # přidá jednoduché uvozovky
            else:
                radek += str(col)  # převede číslo na text
        outfile.write(radek + '\n')
```

Výstupní soubor ***sqlite3_ceny.csv***

```
'40001',1.0,'Materiál 1'
'40002',2.01,'Materiál 2'
'40003',3.01,'Materiál 3'
'40004',4.01,'Mater'
```

Převod z CSV souboru do tabulky

Znakové hodnoty musí být uzavřeny v uvozovkách.

```
cursor.execute('DELETE FROM ceny')  # smažu tabulku
connection.commit()

import csv

# převedu CSV soubor do tabulky
with open("sqlite3_ceny.csv") as infile:
    rows = csv.reader(infile, quoting=csv.QUOTE_NONNUMERIC)
    for row in rows:
       cursor.execute("INSERT INTO  ceny  VALUES (?, ?, ?)", row)
connection.commit()

# zkontroluji obsah tabulky
cursor.execute("SELECT * FROM ceny")
print(cursor.fetchall())
```

```
[('40001', 1.0, 'Materiál 1'), ('40002', 2.01, 'Materiál 2'), ('40003',
3.01, 'Materiál 3'), ('40004', 4.01, 'Mater')]
```

Zpracování výsledné tabulky pomocí knihovny Pandas

```
pip install pandas
```

```
import pandas as pd

dotaz = '''SELECT zavod, sklad, S.mater, nazev, cenaj, mnoz,
                  round(cenaj*mnoz, 2) as celkem
    FROM stavy as S  JOIN ceny as C on S.mater = C.mater ORDER BY mnoz'''
cursor.execute(dotaz)  # provede se dotaz

# seznam jmen sloupců pomocí "list comprehension" z popisu dotazu
headings = [desc[0].upper() for desc in cursor.description]
# ['ZAVOD', 'SKLAD', 'MATER', 'NAZEV', 'CENAJ', 'MNOZ', 'CELKEM']

result_set = cursor.fetchall()  # výsledná tabulka dotazu

df = pd.DataFrame(result_set, columns=headings)  # objekt třídy DataFrame
```

DataFrame je tabulka dat jako v SQL, ale včetně jmen sloupců a řádků.

Výstup do formátu HTML

Převedu data frame do textového souboru typu HTML

```
with open("sqlite3_zasoby.html", encoding="utf-8", mode="w") as outfile:
    print(df.to_html(index=False), file=outfile)
```

```
<table border="1" class="dataframe">
  <thead>
    <tr style="text-align: right;">
      <th>ZAVOD</th>
    ...
```

Zobrazení HTML souboru v prohlížeči

ZAVOD	SKLAD	MATER	NAZEV	CENAJ	MNOZ	CELKEM
01	01	40001	Materiál 1	1.00	0.00	0.00
01	02	40002	Materiál 2	2.01	100.05	201.10
02	01	40003	Materiál 3	3.01	100.05	301.15
02	02	40004	Mater	4.01	100.05	401.20

Výstup do formátu EXCEL

```
pip install openpyxl
```

```
import openpyxl

df.to_excel("sqlite3_zasoby.xlsx")
```

Zobrazení v LibreOffice:

	A	B	C	D	E	F	G	H
1		ZAVOD	SKLAD	MATER	NAZEV	CENAJ	MNOZ	CELKEM
2	0	01	01	40001	Materiál	1	0	0
3	1	01	02	40002	Materiál	2,01	100,05	201,1
4	2	02	01	40003	Materiál	3,01	100,05	301,15
5	3	02	02	40004	Mater	4,01	100,05	401,2

Databáze MySQL

https://mysql.com/

https://dev.mysql.com/downloads/connector/python/

```
pip install mysql-connector-python
```

Zapnutí a vypnutí serveru

Různé v macOS, Windows a Linux

Připojení k databázi

```
import mysql.connector

connection = None
try:
    connection = mysql.connector.connect(
        host="localhost",
        user="root",
        password="012345678"
    )
    print("Database connection created.")
except Exception as err:
    print(f"Error: {err}. Connection failed.")
    exit(1)
```

Vytvoření kurzoru

```
cursor = connection.cursor()
```

Vytvoření schematu

```
schema_name = "zasoby"  # jméno schematu = jméno databáze

drop_schema = f"DROP DATABASE {schema_name}"
try:
    cursor.execute(drop_schema)
except Exception as err:
    print("Chyba odstraňování databáze:", err)


cursor.execute(f"CREATE DATABASE {schema_name}")
```

Zařazení schematu do databáze

```
connection.database = schema_name
```

Vytvoření tabulek

Jméno databáze (schematu) není v příkazech povinné.

```
try:
    cursor.execute('DROP TABLE stavy')
    cursor.execute('DROP TABLE ceny')
except mysql.connector.Error as err:
    pass

cursor.execute('''
CREATE TABLE stavy
            (zavod char(2),
             sklad char(2),
             mater char(5),
             mnoz decimal(10,2))''')
cursor.execute('''
CREATE TABLE ceny
            (mater char(5),
            cenaj decimal(10,2),
            nazev varchar(50))''')
```

Zápis dat do tabulky STAVY hromadně

```
# %s je značka místa, kam patří příslušná hodnota sloupce
sql = "INSERT INTO stavy VALUES (%s,%s,%s,%s)"
val = [('01','01','40001',0),
       ('01','02','40002',100.05),
       ('02','01','40003',100.05),
       ('02','02','40004',100.05)]

cursor.executemany(sql, val)

# potvrdím data metodou commit()
connection.commit()
```

Zápis dat do tabulky CENY převodem z CSV souboru

Máme-li například soubor ***mysql_ceny.csv***

```
40001,1.0,Materiál 1
40002,2.01,Materiál 2
40003,3.01,Materiál 3
40004,4.01,Mater
```

třeba bez uvozovek, který odpovídá struktuře SQL tabulky CENY, můžeme jeho data vložit do tabulky:

```
with open("mysql_ceny.csv", mode="r") as infile:
    rows = csv.reader(infile)
    for row in rows:
        cursor.execute("INSERT INTO ceny VALUES (%s, %s, %s)", row)

    cursor.execute("SELECT * FROM ceny")  # kontrola
    print(cursor.fetchall())  # výsledek dotazu je seznam n-tic
```

```
[('40001', Decimal('1.01'), 'Materiál 1'), ('40002', Decimal('2.01'),
'Materiál 2'), ('40003', Decimal('3.01'), 'Materiál 3'), ('40004',
Decimal('4.01'), 'Mater')]
```

Zpětný převod z tabulky CENY do CSV souboru

```
result_set = cursor.execute("SELECT * FROM ceny")

with open("mysql_ceny2.csv", mode="w") as outfile:
    for row in result_set:
        line = ''
        for n, col in enumerate(row):
            if n != 0:
                line += ','
            if type(col) == str:
                line += f'"{col}"'  # doplní dvojité uvozovky
            else:
                line += str(col)
        print(line, file=outfile)
```

Výstupní soubor ***mysql_ceny2.csv***

```
"40001",1.00,"Materiál 1"
"40002",2.01,"Materiál 2"
"40003",3.01,"Materiál 3"
"40004",4.01,"Mater"
```

Databáze PostgreSQL

https://www.postgresql.org/download/

```
pip install psycopg2-binary
```

Zapnutí a vypnutí serveru

Různé v macOS, Windows a Linux

Připojení k databázi

```
import psycopg2


connection = None
try:
    connection = psycopg2.connect(
        host="localhost",
        user="vzupka",
        database="postgres",
        password="012345678")
    print("PostgreSQL database connection created.")
except Exception as err:
    print(f"Error: {err}. Program ended. ")
    exit(1)
```

Vytvoření kurzoru

```
cursor = connection.cursor()
```

Schema není nutné vytvářet (jeho standardní jméno je ***public*** *)*

Vytvoření tabulek

Datové typy podle standardu SQL

```
cursor.execute('''
CREATE TABLE stavy
            (zavod char(2),
             sklad char(2),
             mater char(5),
             mnoz decimal(10,2))''')
cursor.execute('''
CREATE TABLE ceny
            (mater char(5),
            cenaj decimal(10,2),
            nazev varchar(50))''')
```

CSV —> tabulka

Máme-li například soubor ***pgsql_ceny.csv***

```
40001,1.0,Materiál 1
40002,2.01,Materiál 2
40003,3.01,Materiál 3
40004,4.01,Mater
```

třeba bez uvozovek u textových dat, který odpovídá struktuře SQL tabulky ceny, vložíme data do tabulky s použitím specifické metody ***copy_from()***

```
with open("pgsql_ceny.csv", mode="r") as infile:
    cursor.copy_from(infile, table="ceny", sep=",")

    cursor.execute("SELECT * FROM ceny")  # kontrola
    print(cursor.fetchall())
```

s výsledkem

```
[('40001', Decimal('1.01'), 'Materiál 1'), ('40002', Decimal('2.01'),
'Materiál 2'), ('40003', Decimal('3.01'), 'Materiál 3'), ('40004',
Decimal('4.01'), 'Mater')]
```

Grafické prostředí PySimpleGui

https://www.pysimplegui.org/en/latest/

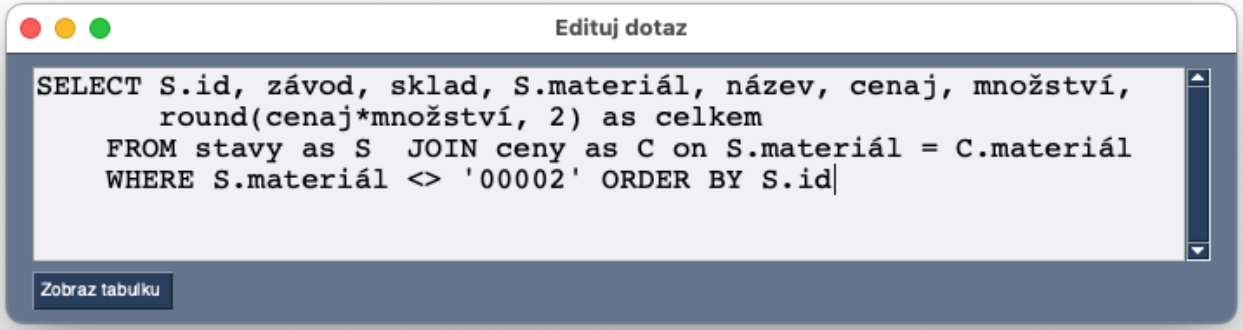
```
pip install pysimplegui
```

Vytvoření databáze, tabulek a dat

```
import sqlite3

connection = sqlite3.connect('zasoby.db')
cursor = connection.cursor()

cursor.execute('DROP TABLE IF EXISTS ceny')
cursor.execute('DROP TABLE IF EXISTS stavy')

cursor.execute("""CREATE TABLE ceny (id INTEGER PRIMARY KEY AUTOINCREMENT,
materiál TEXT, cenaj REAL, název TEXT)""")

cursor.execute("""CREATE TABLE stavy (id INTEGER PRIMARY KEY AUTOINCREMENT,
závod TEXT, sklad TEXT, materiál TEXT, množství REAL)""")
```

```
cursor.executemany("INSERT INTO ceny (materiál, cenaj, název)
                                    VALUES (?, ?, ?)",
                   [
                       ('00001', 8.99, 'PIŠKOTY OPAVIA'),
                       ('00002', 459.00, 'Zubní pasta Kalodont'),
                       ...
                   ])

cursor.executemany("""INSERT INTO stavy (závod, sklad, materiál, množství)
                                     VALUES (?, ?, ?, ?)""",
                   [
                       ('01','01','00009',1.05),
                       ('01','02','00002',2.05),
                       ...
                   ])

connection.commit()
cursor.close()
connection.close()
```

Dotaz a zobrazení výsledku v oknech



id	závod	sklad	materiál	název	cenaj	množství	celkem
1	01	01	00009	Whisky Balantine	250.0	1.05	262.5
3	02	01	00003	Prádelní šňůra	1.25	3.05	3.81
4	02	02	00001	PIŠKOTY OPAVIA	8.99	400.05	3596.45
5	03	01	00005	Tričko bílé	120.0	5.55	666.0
7	03	03	00005	Tričko bílé	120.0	5.55	666.0
9	02	01	00007	Kalhoty manšestrové	1510.0	7.05	10645.5
10	02	02	00008	Kalhoty džínové	1700.0	800.05	1360085.0
11	01	01	00009	Whisky Balantine	250.0	9.0	2250.0
12	01	02	00010	Koňak Gruzínský	6500.0	40.05	260325.0
13	02	01	00001	PIŠKOTY OPAVIA	8.99	50.06	450.04

```
import PySimpleGUI as sg

# Funkce – získání dat z výsledné tabulky

def data_tabulky():  # funkce získání hlaviček a dat z dotazu
    hlavicky = []  # seznam hlaviček
    data = []  # řádky jako seznam seznamů (řádků - záznamů)
    row_obj = cursor.fetchone()  # přečtu záznam - nyní je to Row objekt
    while row_obj:  # v cyklu přidám řádek do dat a čtu další záznam
        hlavicky = row_obj.keys()  # dostanu seznam jmen sloupců (hlaviček)
        data.append(list(row_obj))  # Row objekt musím přeměnit na seznam
        row_obj = cursor.fetchone()
    return hlavicky, data

# HLAVNÍ PROGRAM

   # Připojení k databázi a první dotaz ke zjištění hlaviček a dat
connection = sqlite3.connect('zasoby.db')
connection.row_factory = sqlite3.Row  # objekt s údaji o řádku tabulky
cursor = connection.cursor()  # kurzor pro řádky s hlavičkami

   # První, vzorový SQL dotaz
text_dotazu = '''SELECT S.id, závod, sklad, S.materiál, název, cenaj, množství,
       round(cenaj*množství, 2) as celkem
    FROM stavy as S  JOIN ceny as C on S.materiál = C.materiál
    WHERE S.materiál <> '00002' ORDER BY S.id'''
cursor.execute(text_dotazu)  # provedu vzorový dotaz

hlavicky, data = data_tabulky()  # hlavičky a data pro tabulku z prvního dotazu

print = sg.easy_print  # přesměrování tisku do ladicího okna (debug window)
```

```
# Řídicí cyklus

while True:
    textova_oblast = sg.Multiline(auto_size_text=False, default_text=text_dotazu,
                                  font='Courier 17', size=(60, 6), expand_x=True)
    rozvrh1 = [[textova_oblast], [sg.Button("Zobraz tabulku")]]
    okno1 = sg.Window('Edituj dotaz', rozvrh1, resizable=True)  # zobrazím okno
    event, values = okno1.read()  # čekám a přečtu události a hodnoty z okna

    if event == "Zobraz tabulku":
        try:
            text_dotazu = textova_oblast.get()  # přečtu nový text dotazu
            cursor.execute(text_dotazu)  # provedu nový dotaz
            hlavicky, data = data_tabulky()  # nové hlavičky a data z dotazu
        except sqlite3.OperationalError as err:  # testuji chybu sqlite3
            print(err)  # chyba se objeví v okně pro přesměrovaný tisk
        tabulka = sg.Table(values=data, headings=hlavicky,
                           font='Times 15 roman', auto_size_columns=True,
                           expand_x=True, justification='right')
        rozvrh = [[tabulka]]  # jenom tabulka s uzávěrem
        okno = sg.Window('Výsledek dotazu', rozvrh, resizable=True)  # zobrazím okno
        okno1.close()  # zavřu první okno s textem dotazu
        okno.read()  # čekám a přečtu události a hodnoty z okna
        okno.close()  # po klepnutí na uzávěr okna zavřu okno a opakuji cyklus

    if event == sg.WIN_CLOSED:  # po klepnutí na uzávěr okna končím cyklus
        okno1.close()
        break
```

Poznámky

Zjištění typu sloupců

```
text_prikazu = f'PRAGMA table_info("stavy")'
sloupce_info = cursor.execute(text_prikazu).fetchall()
jmena_sl = [col[1] for col in sloupce_info]  # seznam jmen sloupců

print(jmena_sl)
```

```
['id', 'závod', 'sklad', 'materiál', 'množství']
```

```
import sqlite_utils

db = sqlite_utils.Database(connection)  # přístup k údajům o databázi
typy_sloupcu = db["stavy"].columns_dict  # získám slovník jmen a typů sloupců

print(typy_sloupcu)
```

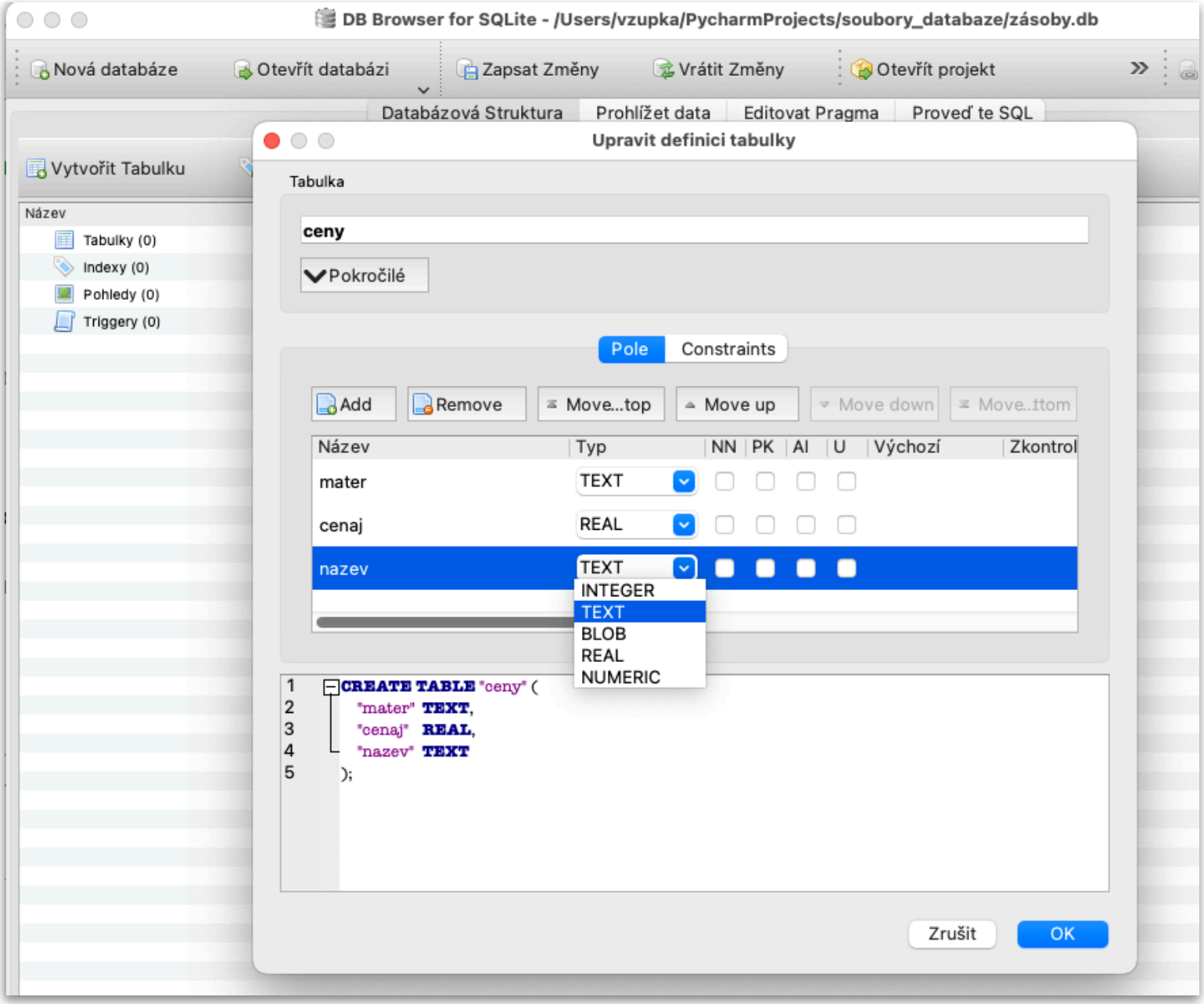
```
{'id': <class 'int'>, 'závod': <class 'str'>, 'sklad': <class 'str'>, 'materiál':
<class 'str'>, 'množství': <class 'float'>}
```

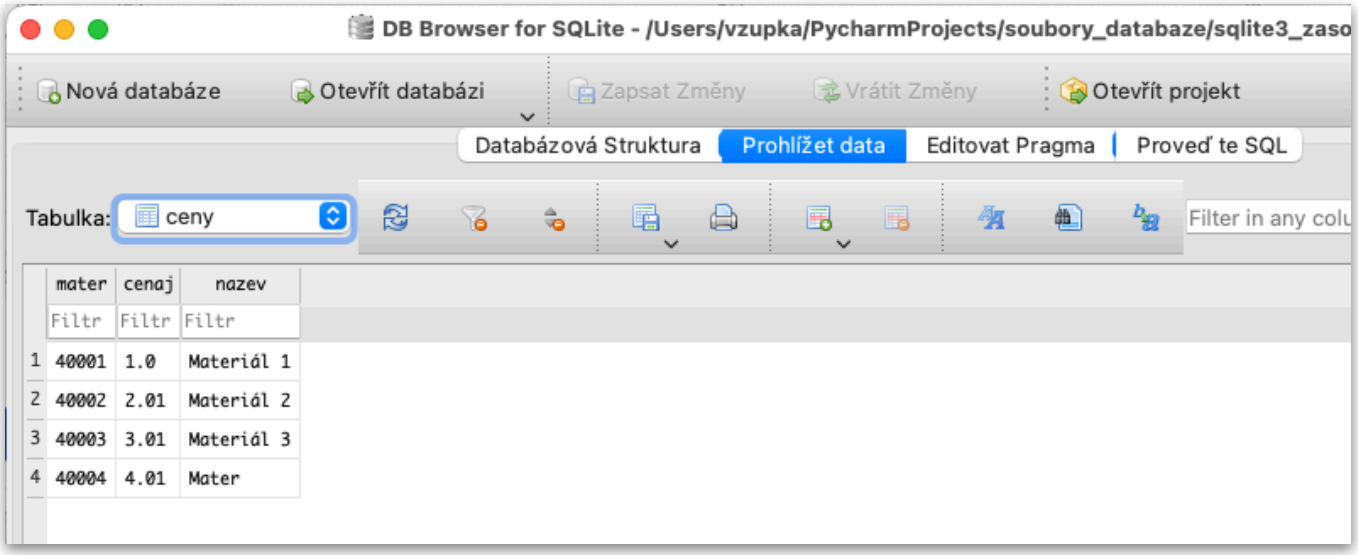
Vytvoření nové tabulky z dotazu

```
dotaz = '''SELECT zavod, sklad, S.mater, nazev, cenaj, mnoz,
                  round(cenaj*mnoz, 2) as celkem
    FROM stavy as S
    JOIN ceny as C on S.mater = C.mater
    ORDER BY mnoz'''
try:
    cursor.execute('DROP TABLE nova_tabulka')  # smazání tabulky
except:
    pass

cursor.execute(f'''CREATE TABLE nova_tabulka AS {dotaz}''')  # vytvoření nové tabulky

cursor.execute('SELECT * FROM nova_tabulka')  # kontrolní dotaz

print(f"nova_tabulka:\n{cursor.fetchall()}")
```

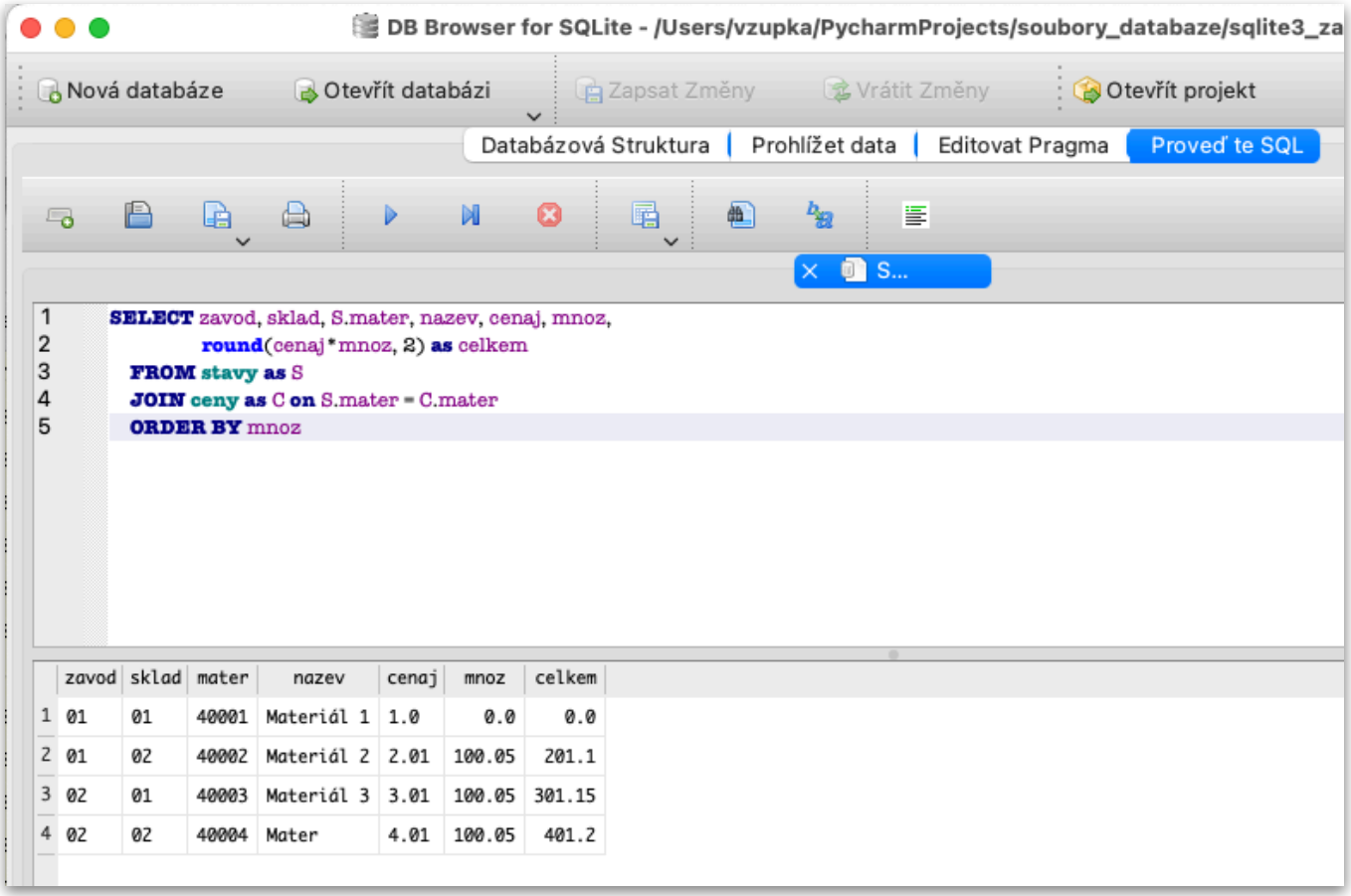
```
nova_tabulka:
[('01', '01', '40001', 'Materiál 1', 1.0, 0.0, 0.0), ('01', '02', '40002', 'Materiál 2', 2.01,
100.05, 201.1), ('02', '01', '40003', 'Materiál 3', 3.01, 100.05, 301.15), ('02', '02', '40004',
'Mater', 4.01, 100.05, 401.2)]
```

DB Browser for SQLite





	mater	cenaj	nazev
	Filtr	Filtr	Filtr
1	40001	1.0	Materiál 1
2	40002	2.01	Materiál 2
3	40003	3.01	Materiál 3
4	40004	4.01	Mater




```
1  SELECT zavod, sklad, S.mater, nazev, cenaj, mnoz,
2         round(cenaj*mnoz, 2) as celkem
3    FROM stavy as S
4    JOIN ceny as C on S.mater = C.mater
5    ORDER BY mnoz
```

	zavod	sklad	mater	nazev	cenaj	mnoz	celkem
1	01	01	40001	Materiál 1	1.0	0.0	0.0
2	01	02	40002	Materiál 2	2.01	100.05	201.1
3	02	01	40003	Materiál 3	3.01	100.05	301.15
4	02	02	40004	Mater	4.01	100.05	401.2