

Tvorba aplikace pro simulaci problémů v databázovém systému

Lukáš Hanusek

27. května 2019

Aplikace pro hromadné vytěžování několika databázových serverů

- Možnost připravit seznam databázových serverů
- Možnost otestovat dostupnost zadaných databázových serverů
- Možnost předpřipravit posloupnost SQL příkazů
- Validace SQL
- Podpora Oracle Databáze a MS SQL Serveru

Použité technologie pro implementaci aplikace

- Jazyk Java
- JDBC
- XML pro datové soubory a JAXB
- JavaFX

Struktura aplikace

Spuštěná úloha

- Vykonává úlohu
- Připravuje spojení
- Monitoruje výsledky
- Zaznamenává chyby do logu

Spustitelná úloha

- Interval
- Počet opakování
- Počet spojení
- Seznam databází
- Seznam SQL

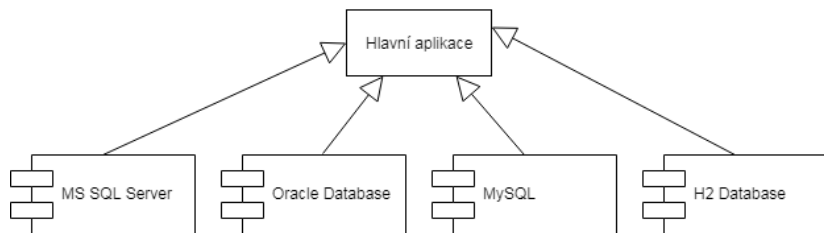
Údaje k DB

- Adresa, port
- Typ databáze
- Přihlašovací údaje

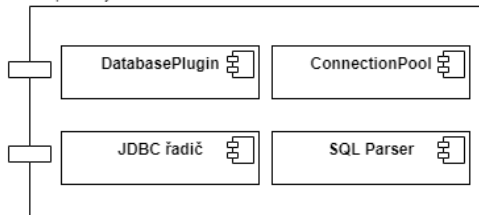
SQL s parametry

- SQL + typ
- Typ databáze
- Parametry + funkce

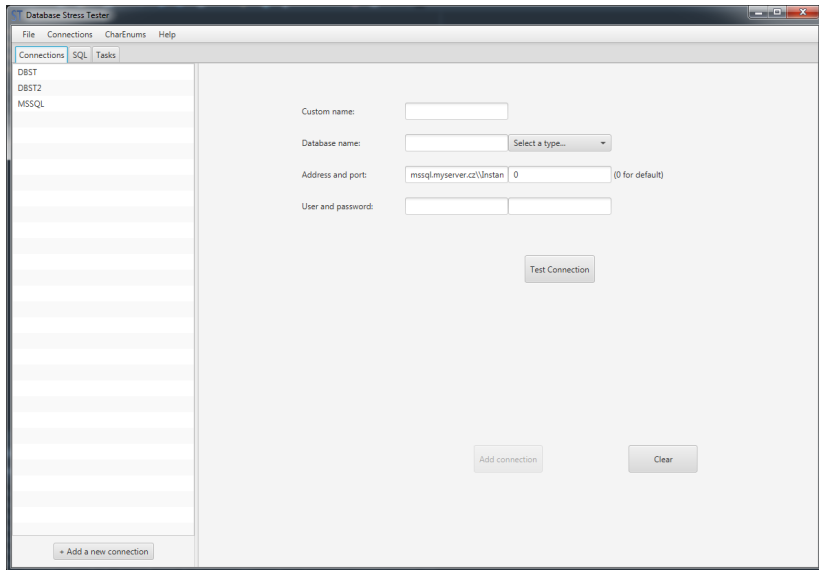
Modulární podpora databázových systémů



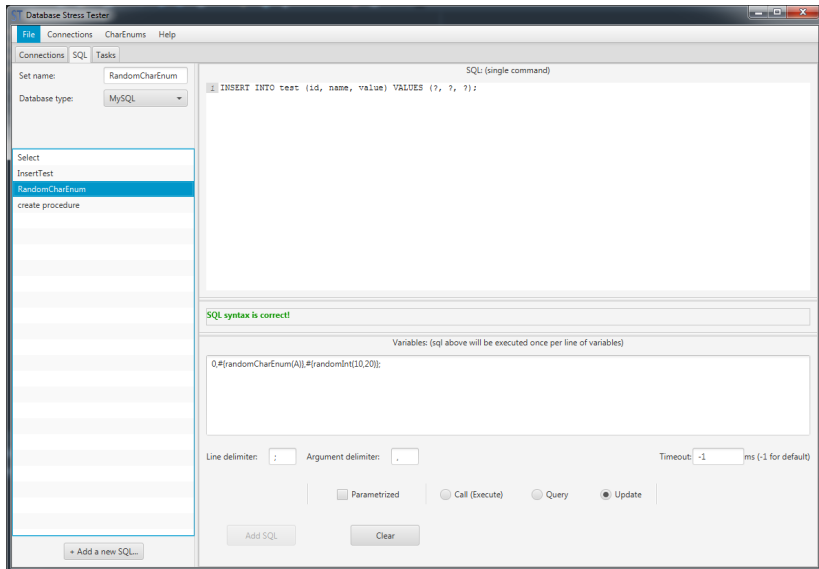
Komponenty rozšíření



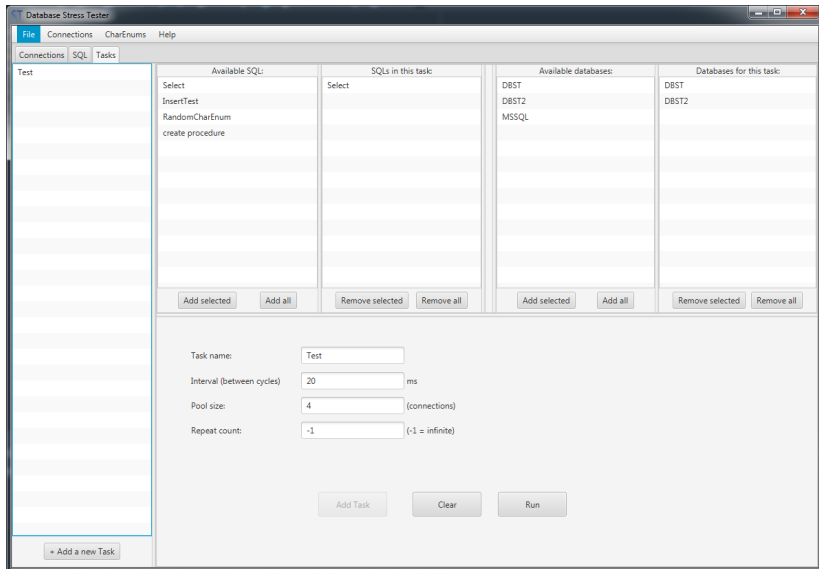
Uživatelské rozhraní - okno editoru



Uživatelské rozhraní - okno editoru



Uživatelské rozhraní - okno editoru



Uživatelské rozhraní - okno monitoru

The screenshot displays the SQL Task Monitor interface with three main sections:

- Databases in this task:** A list containing DBST and DBST22.
- Selected DB:** DBST
- Console:** A log of query execution times, such as "[12:48:23]: Query select * from test; took 47ms".

Summary statistics are shown on the right:

- Loops: 785 / -1
- Queries: 785 / -1
- Total time: 0d 00:00:15
- Total query time: 38124 ms
- Pool status: 0 / 4(0 idle)

Query	Avg. Time	Executed	Errors	Last success	Last error
Select	48	785	2	7.4.2019 12:48:24	7.4.2019 12...

At the bottom, there are controls for "Start all", "Stop all", and "CSV Export all".

Uživatелеm definované funkce

```
1 INSERT INTO test (id, name, value) VALUES (?, ?, ?);
```

SQL syntax is correct!

Variables: (sql above will be executed once per line of variables)

```
0,#{randomCharEnum(A)},#{randomInt(10,20)};
```

Line delimiter:

Argument delimiter:

Timeout:

ms (-1 for default)

Uživatелеm definované funkce

- *Funkce* **randomInt(MIN,MAX)**:

Argumenty:

MIN - definuje minimální velikost generovaného čísla

MAX - definuje maximální velikost generovaného čísla

Příklad použití funkce:

`#{randomInt(1,10)}` - náhodné číslo v rozsahu 1 až 10

- *Funkce* **randomCharEnum(enum)**:

Argumenty:

enum - jméno posloupnosti znaků, která bude použita pro generování nové náhodné posloupnosti

Příklad použití funkce:

`#{randomCharEnum(A)}` - vygeneruje náhodnou posloupnost znaků z posloupnosti se jménem 'A'

ANTLR: SELECT * FROM TEST

Parser	První spuštění	Opakované spuštění
PL-SQL	1831 ms	2 ms
T-SQL	525 ms	1 ms
MySQL	428 ms	1 ms

JSql: SELECT * FROM TEST

Parser	První spuštění	Opakované spuštění
Jsql	30 ms	0.26 ms

ANTLR: SELECT COUNT(*) FROM TEST WHERE CID = 5 AND
PRICE <100 GROUP BY NAME ORDER BY PRICE

Parser	První spuštění	Opakované spuštění
PL-SQL	2522 ms	6 ms
T-SQL	479 ms	6 ms
MySQL	364 ms	5 ms

JSql: SELECT COUNT(*) FROM TEST WHERE CID = 5 AND PRICE
<100 GROUP BY NAME ORDER BY PRICE

Parser	První spuštění	Opakované spuštění
Jsql	33 ms	0.75 ms

Děkuji za pozornost.

Lukáš Hanusek