Performance testing and why even the imperfect one is important

Ondřej Babec
Why?

Why to take the time and do the performance testing

1. Why not?
2. Interesting work
3. Performance gain for product
4. Deeper understanding of product core
5. Obtained numbers can be used as some base information about product expected throughputs
Testing vs Measurements

Numbers are just a side product

No need for an absolutely isolated environment

The goal is to find the bottlenecks of the product

Monitoring and Tracing are necessary

Exact numbers that you can rely on

Completely isolated environment

The goal is to find the numbers not optimize the product

Monitoring of the whole behaviour is not necessary
Goals

- Find bottlenecks
- **Optimize** implementation
- **Compare** different runtimes
- Find **baseline** performance figures
- **Stress** system with various load types
Obstacles

Pre-test steps

- Know your environment!
  - Latency, memory, simply everything
- Prepare **metrics** scraping
  - Speed problems
- **Tune up** all systems that you depend on
  - Databases, ...
- The generated load must be **reproducible**
Knowledge is power.

Francis Bacon
Monitoring and tracing

Key to success

- **Slow scraping** can be a big problem
- As many metrics as possible
  - Hardware
  - Software metrics
- Raw data to graph = **no problem**
- A graph without data = **impossible**
- Use as **few tools** as possible
Patterns and trends!
All you need to look for
SuT - Debezium
Real life example
Real life example
Tips & Tricks

- **DMT** - Database Manipulation Tool
- **Load** generator
- Ansible **automation**
  - MySQL auto tune
- **Netdata** to **Prometheus** scraper
Summary

Let’s find the **bottlenecks** together!

- Don’t be **scared**
- Gather as many metrics as you can
- Look for trends, not concrete numbers
- Tune everything around your app!
- **See our blog and repo for new tooling!**
Thank you!

github.com/skodjob

github.com/debezium/debezium

skodjob.io

linkedin.com/in/ondrej-babec