Introduction to ElaSQL and ElaSQLBench

Database Systems

DataLab, CS, NTHU

Spring, 2021

Recap

- Our final project is to improve Hermes.
- Hermes is in ElaSQL.

Outline

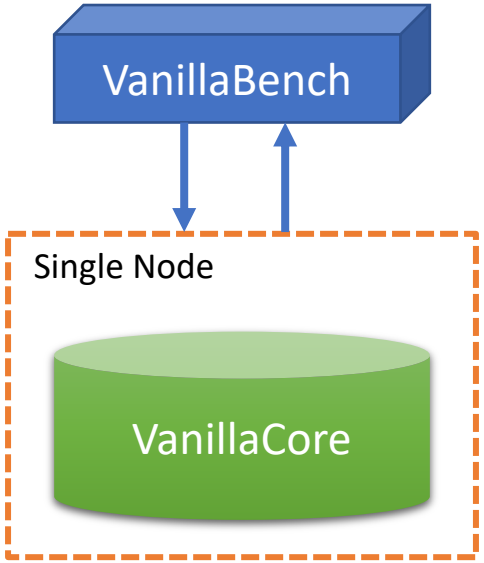
- What is ElaSQLBench & ElaSQL
- Let’s Run a Benchmark on ElaSQL
  - Package code
  - Load Data
  - Workload configuration
  - Start Benchmarking
- Schedule: Next?

Outline

- What is ElaSQLBench & ElaSQL
- Let’s Run a Benchmark on ElaSQL
  - Package code
  - Load Data
  - Workload configuration
  - Start Benchmarking
- Schedule: Next?

Recap

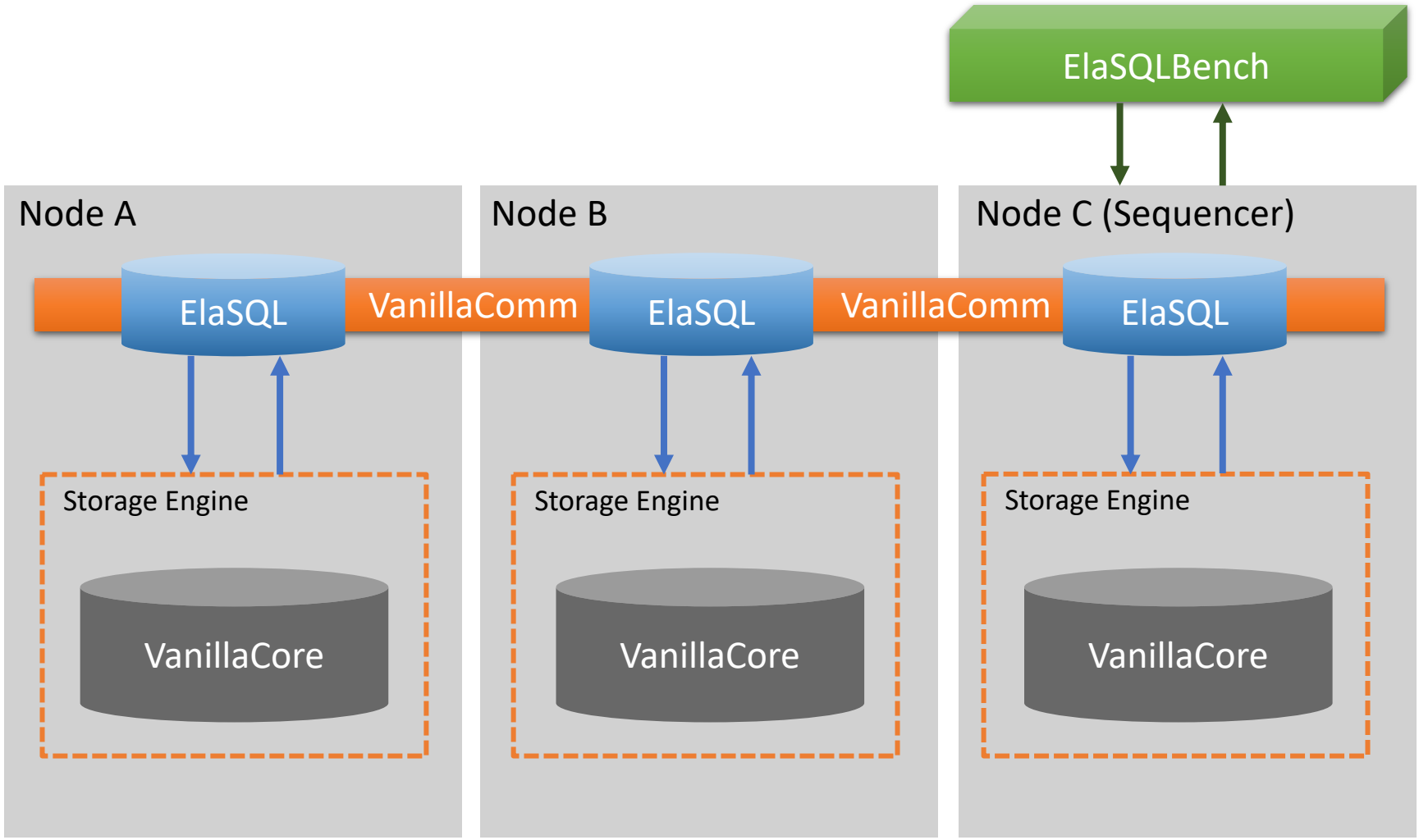
- VanillaCore is a single-node DBMS.
- VanillaBench is a benchmark framework aiming to test VanillaCore.



What is ElaSQL & ElaSQLBench

- **ElaSQL** is a cross-node DBMS that build on top of VanillaCore and VanillaComm.
- **ElaSQLBench** is a benchmark framework aiming to test ElaSQL.
- In the cross-node architecture
  - VanillaCore: as a storage engine (handle create, insert, delete, update from ElaSQL)
  - **VanillaComm**: for cross-node communication
  - VanillaBench: as a basis of ElaSQLBench (code reuse)

Cross-Node Architecture



Roles

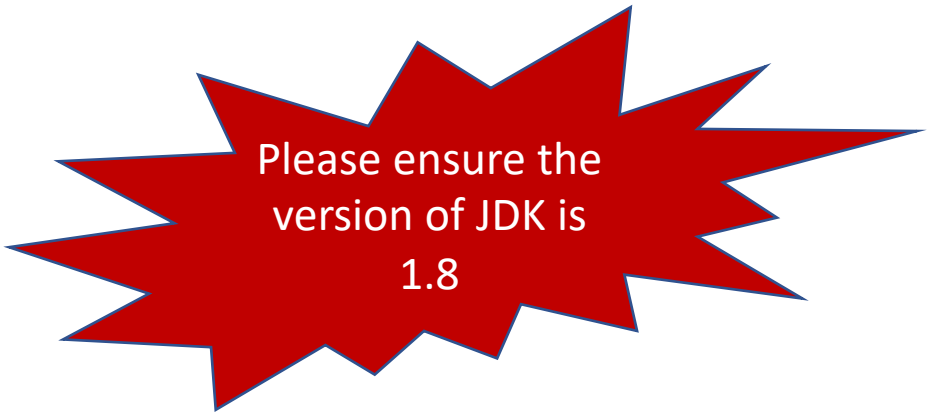
- There are three roles in a cluster
  - Client
  - Server
  - Sequencer
- In a cluster, we have several clients and servers.
- Among the servers, there must be a **SPECIAL** server called sequencer (kind of leader), which coordinates the total-order messages.

Outline

- What is ElaSQLBench & ElaSQL
- Recap: Target Workloads
- Let’s Run a Benchmark on ElaSQL
  - Package code
  - Load Data
  - Workload configuration
  - Start Benchmarking
- Schedule: Next?

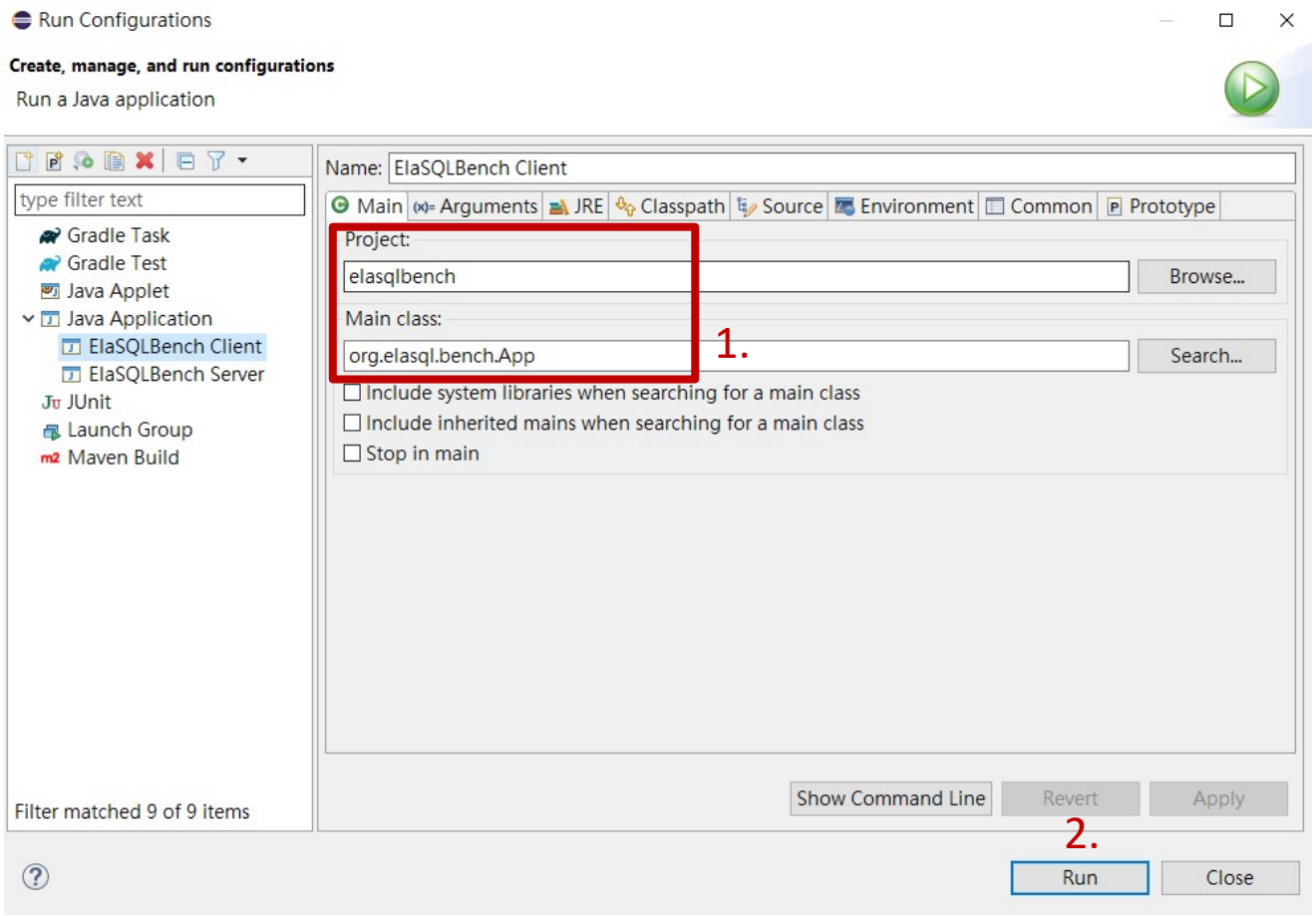
Package ElaSQLBench as JAR files

- Steps
  - 1. setup run configurations for JARs via Eclipse
  - 2. Export the project
- Example: package a client JAR
- Example: package a server JAR

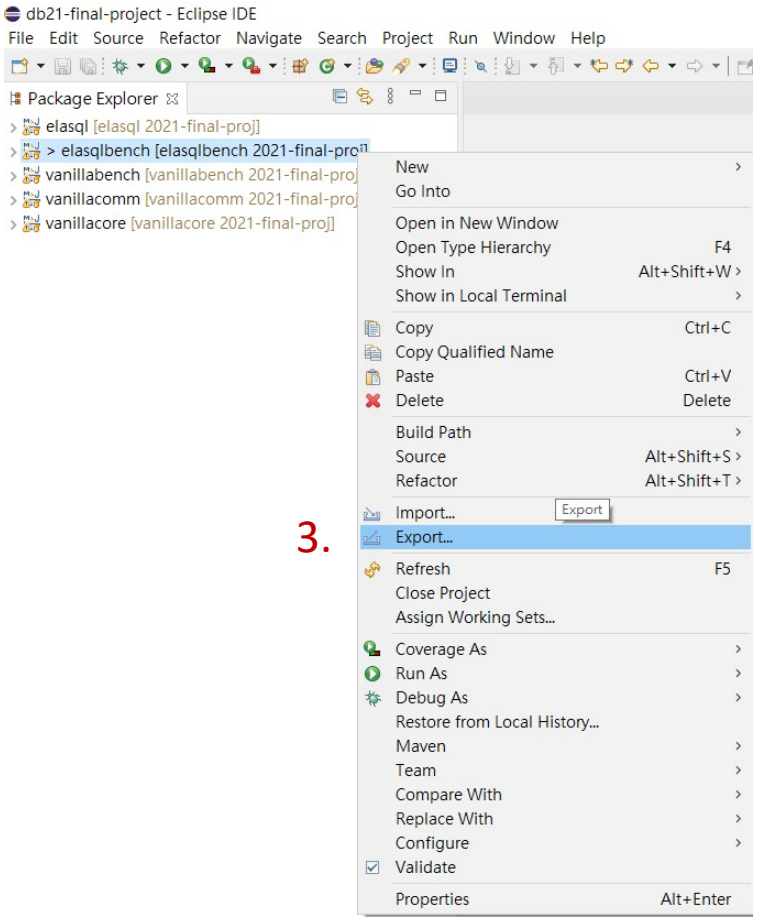


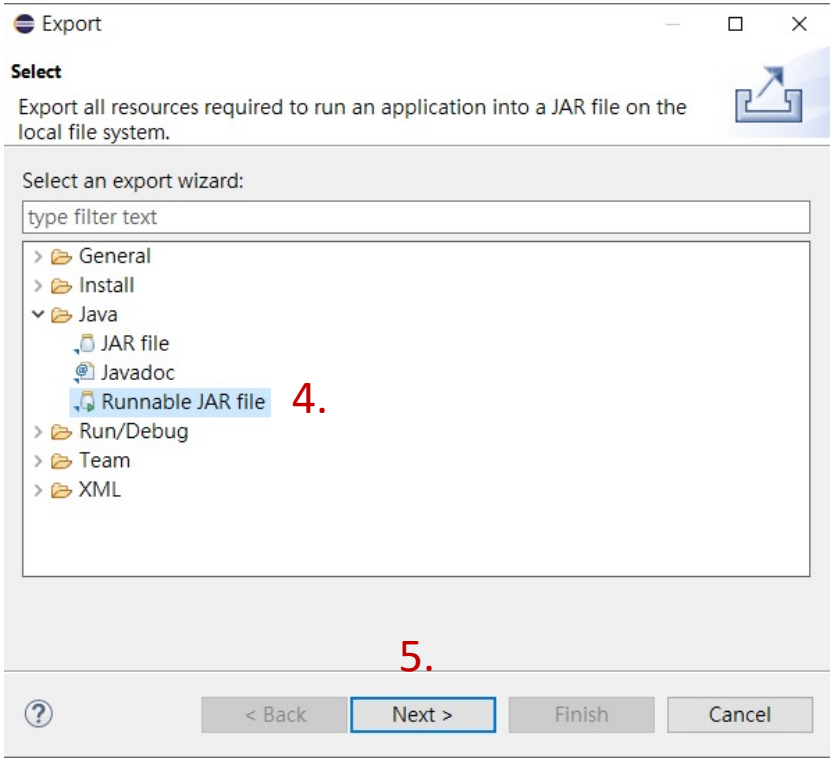
Example: Package a Client JAR

Client

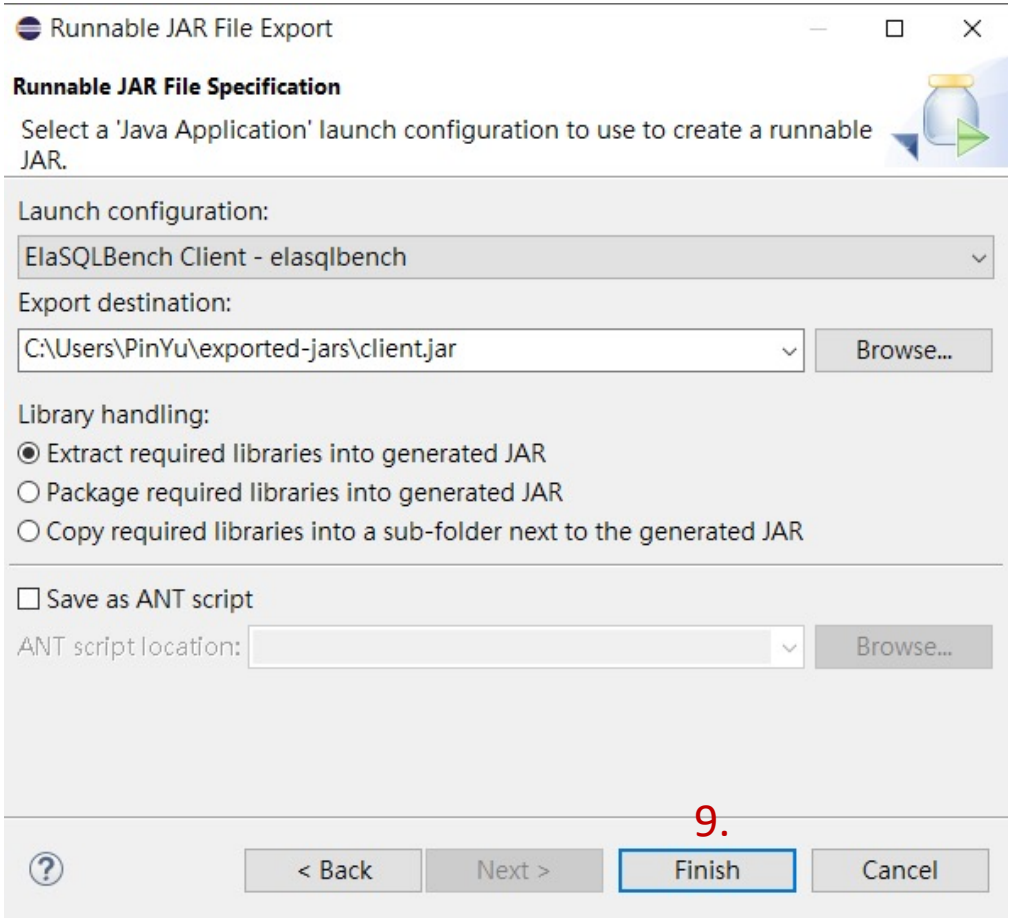

Run Configurations
Create, manage, and run configurations
Run a Java application
type filter text
Gradle Task
Gradle Test
Java Applet
Java Application
ElaSQLBench Client
ElaSQLBench Server
JUnit
Launch Group
Maven Build
Filter matched 9 of 9 items
Name: ElaSQLBench Client
Main
Arguments
JRE
Classpath
Source
Environment
Common
Prototype
Project:
elasqlbench
Browse...
Main class:
org.elasql.bench.App
Search...
1.
Include system libraries when searching for a main class
Include inherited mains when searching for a main class
Stop in main
Show Command Line
Revert
Apply
2.
Run
Close

Client


db21-final-project - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
elasql [elasql 2021-final-proj]
> elasqlbench [elasqlbench 2021-final-proj]
vanillabench [vanillabench 2021-final-proj
vanillacomm [vanillacomm 2021-final-proj
vanillacore [vanillacore 2021-final-proj]
New
Go Into
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export
Export...
Refresh F5
Close Project
Assign Working Sets...
Coverage As
Run As
Debug As
Restore from Local History...
Maven
Team
Compare With
Replace With
Configure
Validate
Properties Alt+Enter
3.


Export
Select
Export all resources required to run an application into a JAR file on the local file system.
Select an export wizard:
type filter text
General
Install
Java
JAR file
Javadoc
Runnable JAR file
4.
Run/Debug
Team
XML
5.
< Back
Next >
Finish
Cancel

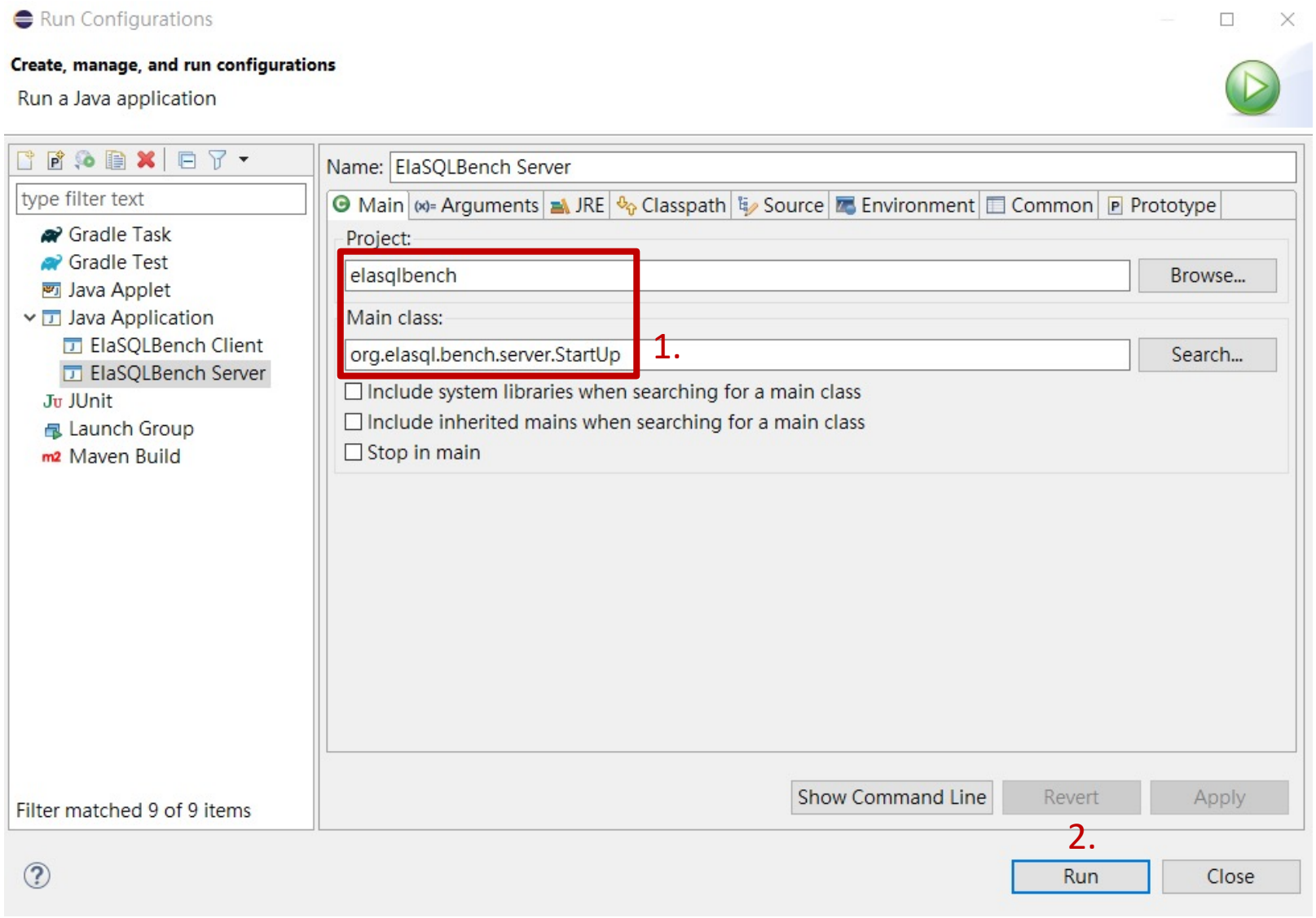
Client



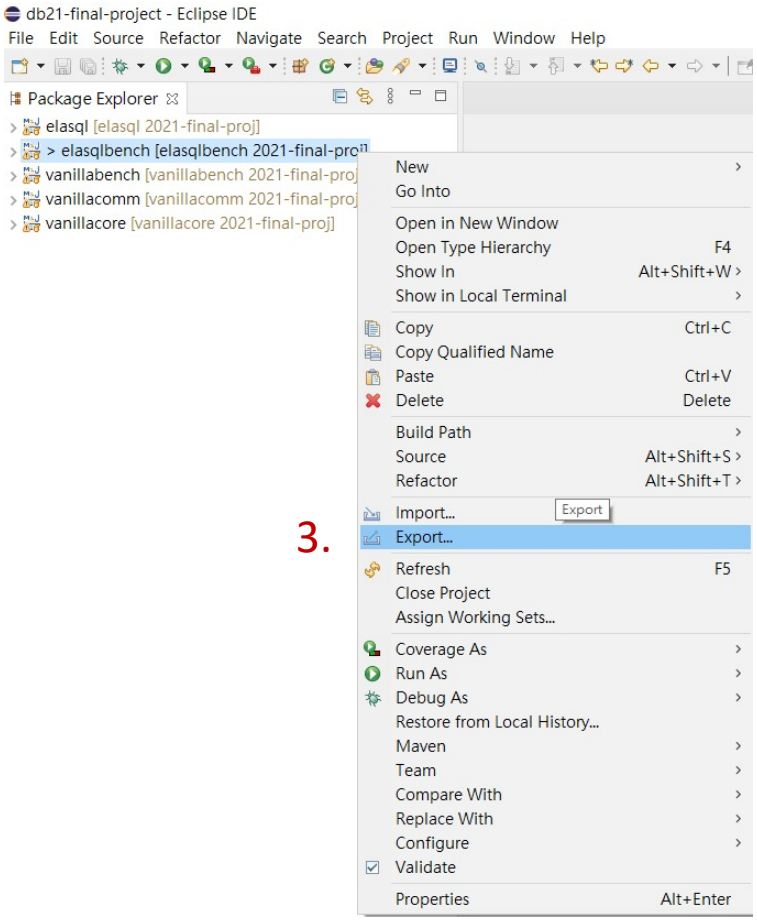
It’s ok to ignore the warning message when exporting

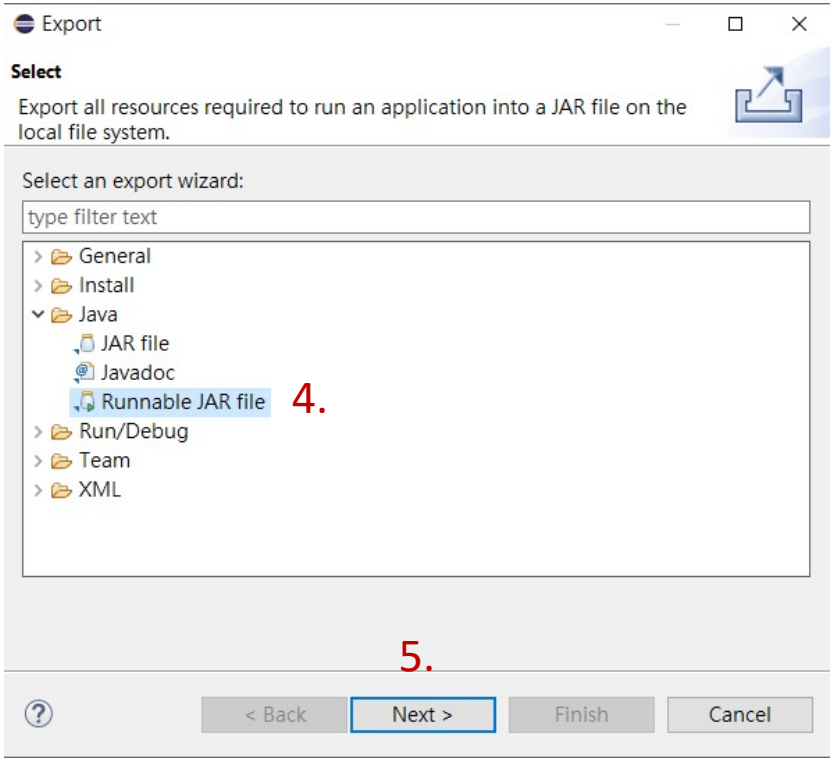
Example: Package a Server JAR

Server

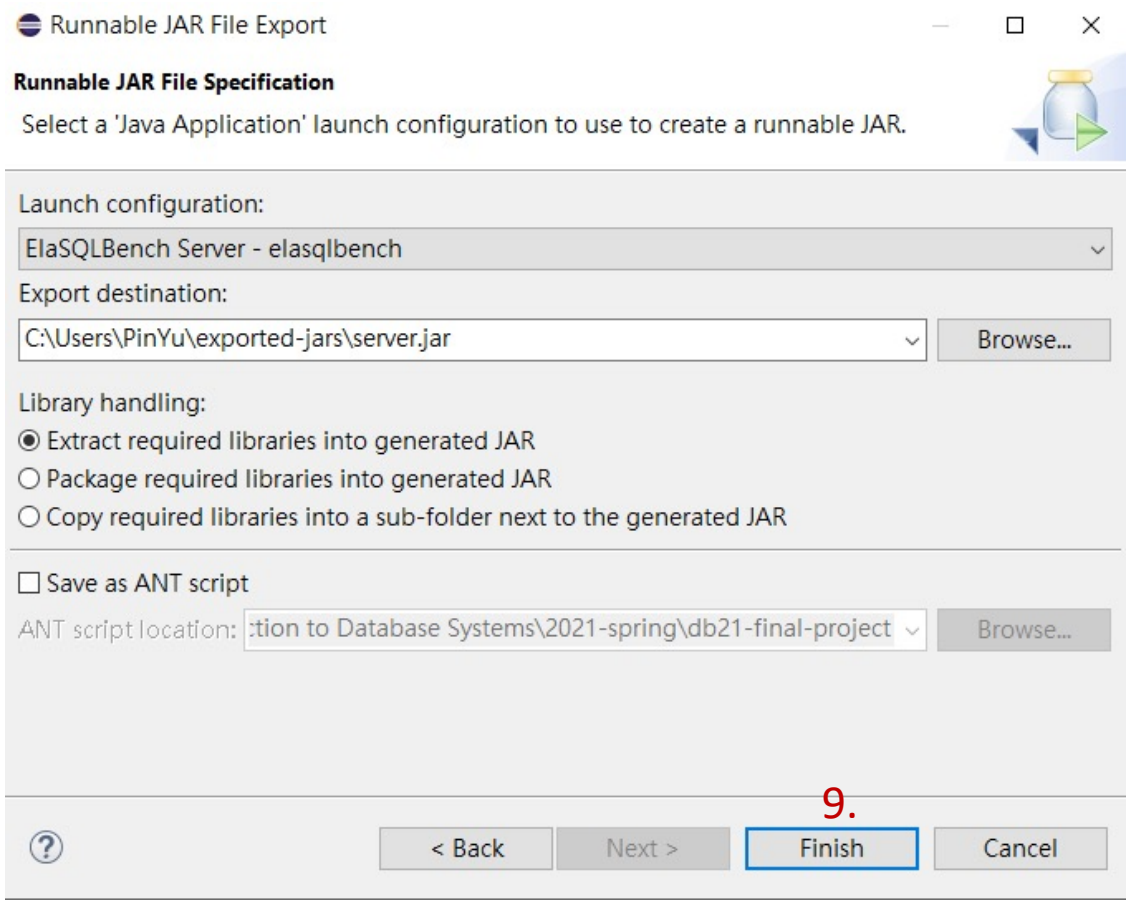

Run Configurations
Create, manage, and run configurations
Run a Java application
type filter text
Gradle Task
Gradle Test
Java Applet
Java Application
ElaSQLBench Client
ElaSQLBench Server
JUnit
Launch Group
Maven Build
Filter matched 9 of 9 items
Name: ElaSQLBench Server
Main
Arguments
JRE
Classpath
Source
Environment
Common
Prototype
Project:
elasqlbench
Browse...
Main class:
org.elasql.bench.server.StartUp
1.
Search...
Include system libraries when searching for a main class
Include inherited mains when searching for a main class
Stop in main
Show Command Line
Revert
Apply
2.
Run
Close

Server


db21-final-project - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
elasql [elasql 2021-final-proj]
> elasqlbench [elasqlbench 2021-final-proj]
vanillabench [vanillabench 2021-final-proj]
vanillacomm [vanillacomm 2021-final-proj]
vanillacore [vanillacore 2021-final-proj]
New
Go Into
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export
Export...
Refresh F5
Close Project
Assign Working Sets...
Coverage As
Run As
Debug As
Restore from Local History...
Maven
Team
Compare With
Replace With
Configure
Validate
Properties Alt+Enter
3.


Export
Select
Export all resources required to run an application into a JAR file on the local file system.
Select an export wizard:
type filter text
General
Install
Java
JAR file
Javadoc
Runnable JAR file
4.
Run/Debug
Team
XML
5.
< Back
Next >
Finish
Cancel

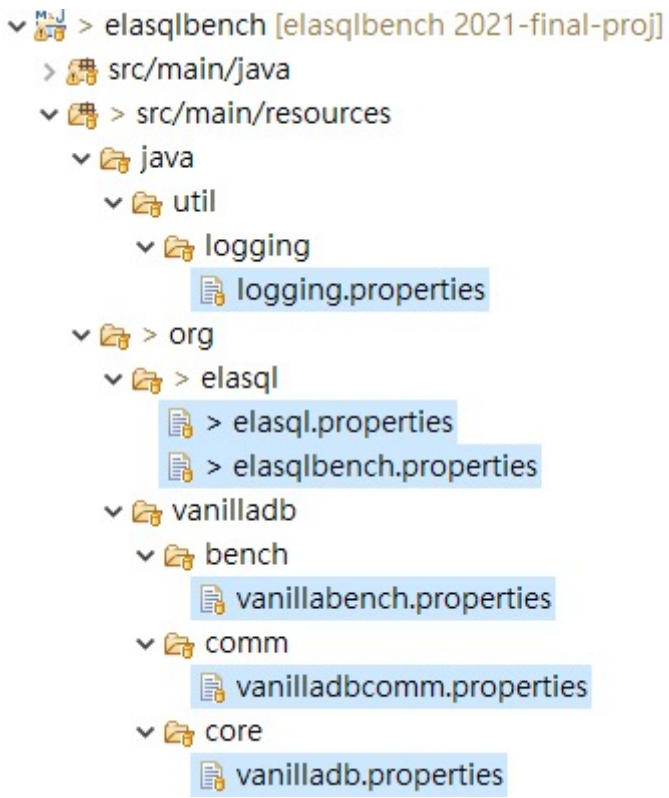
Server



It’s ok to ignore the warning message when exporting

Copy Properties Files

- Copy 6 properties files from ElasqlBench to the export destination



Copy Google-workloads Csv

- Copy google-workloads-2min-3days.csv to the export destination

Result

- There will be 2 jars, 6 properties files and 1 csv file in the export destination

Outline

- What is ElaSQLBench & ElaSQL
- Recap: Target Workloads
- Let’s Run a Benchmark on ElaSQL
  - Package code
  - Load Data
  - Workload configuration
  - Start Benchmarking
- Schedule: Next?

Load Data

- For simplicity, we demonstrate it on a single machine
- 3 server processes (2 normal server + 1 sequencer)
- 1 client process

Load Data

- Modify the properties

Files	Properties

vanilladb.properties	org.vanilladb.core.storage.buffer.BufferMgr.BUFFER_POOL_SIZE=128000
	# set it true if your OS is Linux
	org.vanilladb.core.storage.file.io.IoAllocator.USE_O_DIRECT=true

vanillabench.properties	org.vanilladb.bench.BenchmarkerParameters.BENCH_TYPE=4
elasql.properties	org.elasql.server.Elasql.SERVICE_TYPE=1 org.elasql.remote.groupcomm.client.BatchSpcSender.BATCH_SIZE=1
elasqlbench.properties	org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.DATABASE_MODE=1 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.INIT_RECORD_PER_PART=1000000
vanillacomm.properties	org.vanilladb.comm.view.ProcessView.SERVER_VIEW=0 127.0.0.1 42961, 1 127.0.0.1 42962, 2 127.0.0.1 42963 org.vanilladb.comm.view.ProcessView.CLIENT_VIEW=0 127.0.0.1 30000

3 servers including sequencers is a minimum requirement

1 client is enough when loading the data

You could increase YCSB table size by increasing INIT_RECORD_PER_PART

Run Commands to Start Servers/Clients

- A machine is represented by “ID IP PORT”
- Because we run these processes on a single machine, IP is 127.0.0.1(localhost)

```
# The views of the machine
# A machine is represented by "ID IP PORT"
# Each machine is split by a comma (,)
org.vanilladb.comm.view.ProcessView.SERVER_VIEW=0 127.0.0.1 42961, 1 127.0.0.1 42962, 2 127.0.0.1 42963
org.vanilladb.comm.view.ProcessView.CLIENT_VIEW=0 127.0.0.1 30000
```

Run Commands to Start Servers

- Copy the script below and save it as server.sh
  - Argument $1: Database Name
  - Argument $2: ID
  - Argument $3: isSequencer (0 -> not sequencer, 1 -> is sequencer)

```
java \
-Dorg.elasql.config.file=elasql.properties \
-Dorg.elasql.bench.config.file=elasqlbench.properties \
-Dorg.vanilladb.comm.config.file=vanilladbcomm.properties \
-Dorg.vanilladb.bench.config.file=vanillabench.properties \
-Dorg.vanilladb.core.config.file=vanilladb.properties \
-Djava.util.logging.config.file=logging.properties \
-jar server.jar \
$1 \
$2 \
$3 \
```

Run Commands to Start Servers

- Open a Gitbash and run
  - Start server0

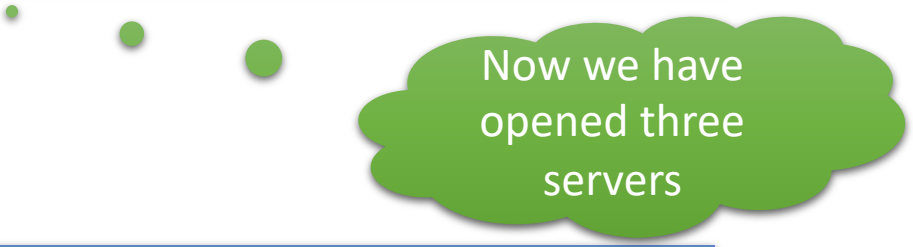
```
bash server.sh db0 0 0
```

  - Start server1

```
bash server.sh db1 1 0
```

  - Start sequencer (Actually, sequencer is for communication only and it won't create any database file)

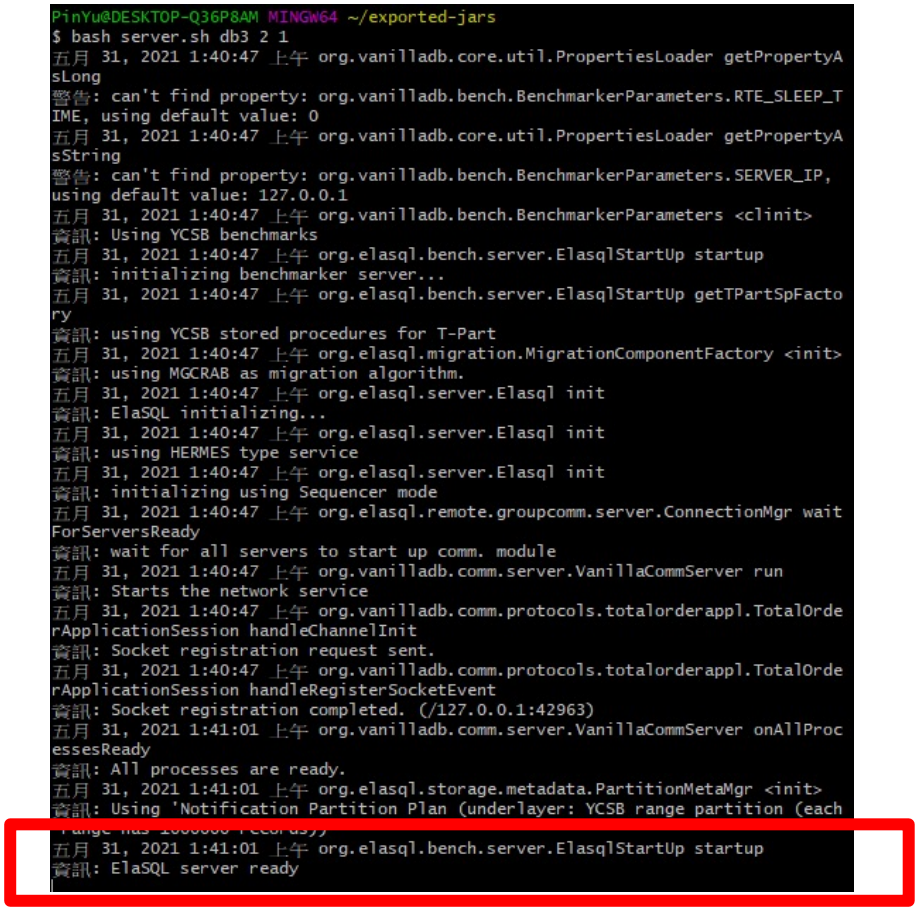
```
bash server.sh dbseq 2 1
```




```
bash server.sh [db-name] [ID] [isSequencer]
```

Run Commands to Start Servers

- Check if “ElaSQL server ready” is printed in the console of sequencer



Run Commands to Start Clients

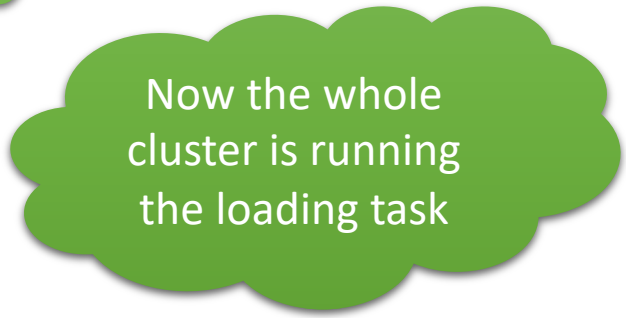
- Copy the script below and save it as client.sh
  - Argument $1: ID
  - Argument $2: BenchType (1: load, 2: benchmark)

```
java \
-Dorg.elasql.config.file=elasql.properties \
-Dorg.elasql.bench.config.file=elasqlbench.properties \
-Dorg.vanilladb.comm.config.file=vanilladbcomm.properties \
-Dorg.vanilladb.bench.config.file=vanillabench.properties \
-Dorg.vanilladb.core.config.file=vanilladb.properties \
-Djava.util.logging.config.file=logging.properties \
-jar server.jar \
$1 \
$2 \
```

Run Commands to Start Servers

- After ElaSQL server ready, ppen a Gitbash and run
  - Start client0

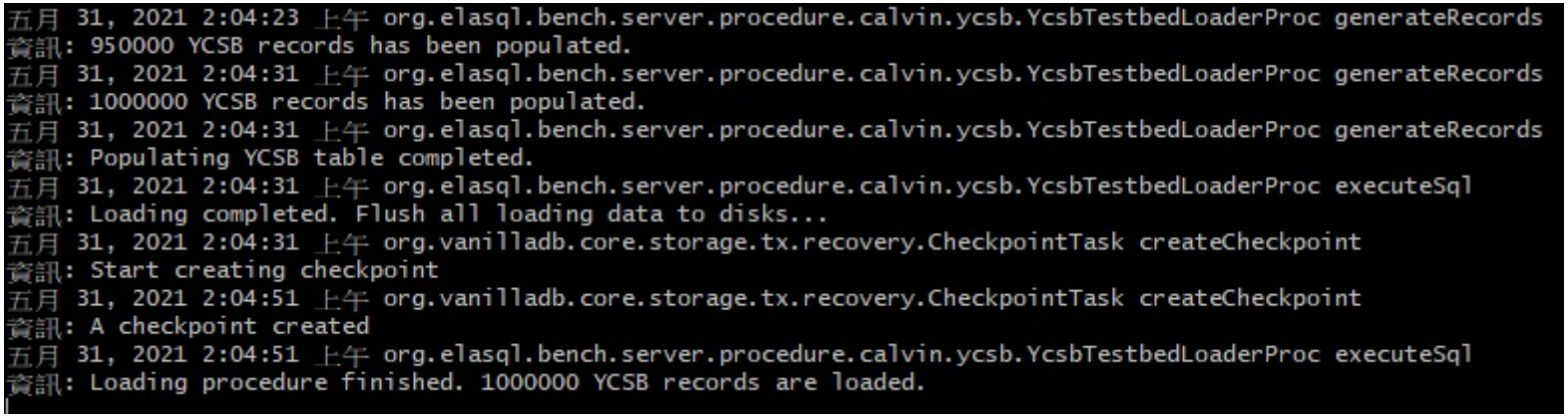
```
bash cleint.sh 0 1
```




```
bash client.sh [ID] [benchType]
```

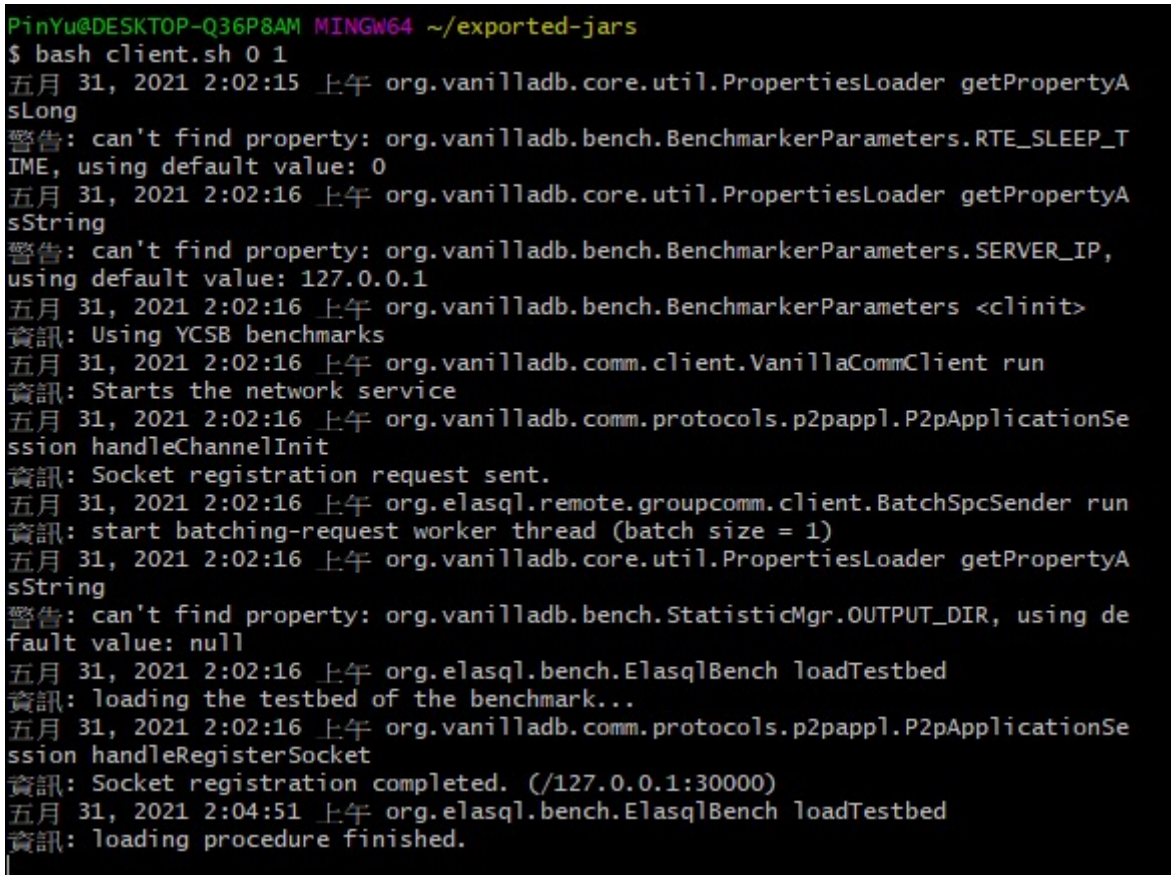
The Scree Shots When Loading Successfully

- On the server console



The Scree Shots When Loading Successfully

- On the client console



The Scree Shots When Loading Successfully

- In db1

The Scree Shots When Loading Successfully

- In db2

Outline

- What is ElaSQLBench & ElaSQL
- Recap: Target Workloads
- Let’s Run a Benchmark on ElaSQL
  - Package code
  - Load Data
  - Workload configuration
  - Start Benchmarking
- Schedule: Next?

Recap: Target Workloads

- We prepare three workloads for testing:
  - The Hotspot Workload
  - The Google Workload
  - The Hot Counter Workload
- For more details, please refer to the introduction slides of final project.

Workload Configuration

- 3 configurations we need to handle.
  - Hotspot Workload
  - Google Workload
  - Hot Counter Workload

Workload Configuration – Hotspot Workload

Files	Properties
vanillabench.properties	org.vanilladb.bench.BenchmarkerParameters.BENCH_TYPE=4 org.vanilladb.bench.BenchmarkerParameters.BENCHMARK_INTERVAL=900000 org.vanilladb.bench.BenchmarkerParameters.RTE_SLEEP_TIME= 0 org.vanilladb.bench.BenchmarkerParameters.NUM_RTES=200 org.vanilladb.bench.StatisticMgr.GRANULARITY=10000

elasql.properties	org.elasql.server.Elasql.SERVICE_TYPE=3
	org.elasql.remote.groupcomm.client.BatchSpcSender.BATCH_SIZE=20
	# T-Part & Hermes
	org.elasql.schedule.tpart.TPartPartitioner.ROUTING_BATCH=200
	# Hermes
	org.elasql.schedule.tpart.hermes.FusionTable.EXPECTED_MAX_SIZE=1000000
	org.elasql.schedule.tpart.hermes.HermesNodeInserter.IMBALANCED_TOLERANCE=0.1

elasqlbench.properties	org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.DATABASE_MODE=1 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.WORKLOAD_TYPE=3 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.RW_TX_RATE=0.5 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.TX_RECORD_COUNT=2 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.ADD_INSERT_IN_WRITE_TX=0 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.ZIPFIAN_PARAMETER=0.99 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.TENANTS_PER_PART=4 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.ENABLE_HOTSPOT=true org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.HOTSPOT_HOTNESS=0.9 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.HOTSPOT_CHANGE_PERIOD= 600

Workload Configuration – Google Workload

After finishing the previous workload (hotspot), please modify these properties to fit the google workload

Files	Properties
elasqlbench.properties	org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.WORKLOAD_TYPE=2 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.DIST_TX_RATE=0.5 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.USE_DYNAMIC_RECORD_COUNT=false org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.GOOGLE_TRACE_FILE=/path/google-workloads-2min-3days.csv org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.GOOGLE_TRACE_LENGTH=2160

Workload Configuration – Hot Counter Workload

After finishing the previous workloads, please modify these properties to fit the hot counter workload

Files	Properties
elasqlbench.properties	org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.WORKLOAD_TYPE=4 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.HOT_COUNT_PER_PART=1 org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.HOT_UPDATE_RATE_IN_RW_TX=0.1

Common Properties

- For the 3 workloads, they are all the same
  - org.vanilladb.bench.BenchmarkerParameters.BENCH_TYPE=4
  - org.elasql.server.Elasql.SERVICE_TYPE=3

Particular Properties

- This properties determines which workloads we are going to run
  - org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.WORKLOAD_TYPE
- In Google Workload, we need to include a special file called google-workloads-2min-3days.csv
  - This file has kept the real cpu usage from Google's datacenter.
  - ElaSQL will reproduce the similar behaviors based on this file.

Some Properties You Can Modify

- It might work well to run 3 clients in one machine (vanillacomm.properties)
- The more powerful your computer is, the larger RTE numbers you can set (vanillabench.properties)
  - 200 might be a good number, please find a better number for your clusters
- BATCH_SIZE (elasql.properties)
  - 20 might be a good number
  - There are some restrictions on this properties, please see the next slide
- ROUTING_BATCH (elasql.properties)
  - 100~500 might be a good range, please find a better number for your clusters
  - There are some restrictions on this properties, please see the next slide

Restriction!

- *org.elasql.remote.groupcomm.client.BatchSpcSender.BATCH_SIZE* must be smaller than *org.vanilladb.bench.BenchmarkerParameters.NUM_RTES*
- *org.elasql.schedule.tpart.TPartPartitioner.ROUTING_BATCH* must be smaller than (org.vanilladb.bench.BenchmarkerParameters.NUM_RTES * Client Nums)
- Please do not modify these properties
  - USE_DYNAMIC_RECORD_COUNT
  - ADD_INSERT_IN_WRITE_TX
  - ZIPFIAN_PARAMETER
  - GOOGLE_TRACE_LENGTH
- You could modify the other properties excluding the above properties. But default value might be good enough.

Outline

- What is ElaSQLBench & ElaSQL
- Recap: Target Workloads
- Let’s Run a Benchmark on ElaSQL
  - Package code
  - Load Data
  - Workload configuration
  - Start Benchmarking
- Schedule: Next?

Start Benchmarking – Google Workload

- Open a Gitbash and run
  - Start server0

```
bash server.sh db0 0 0
```

  - Start server1

```
bash server.sh db1 1 0
```

  - Start sequencer

```
bash server.sh dbseq 2 1
```

  - After ElaSQL server readyt, Start client0

```
bash cleint.sh 0 2
```

Start Benchmarking – Google Workload

Client console

```
PinYu@DESKTOP-Q36P8AM MINGW64 ~/exported-jars
$ bash client.sh 0 2
五月 31, 2021 1:41:08 上午 org.vanilladb.core.util.PropertiesLoader getPropertyA
sLong
警告: can't find property: org.vanilladb.bench.BenchmarkerParameters.RTE_SLEEP_T
IME, using default value: 0
五月 31, 2021 1:41:08 上午 org.vanilladb.core.util.PropertiesLoader getPropertyA
sString
警告: can't find property: org.vanilladb.bench.BenchmarkerParameters.SERVER_IP,
using default value: 127.0.0.1
五月 31, 2021 1:41:08 上午 org.vanilladb.bench.BenchmarkerParameters <clinit>
資訊: Using YCSB benchmarks
五月 31, 2021 1:41:08 上午 org.vanilladb.comm.client.VanillaCommClient run
資訊: Starts the network service
五月 31, 2021 1:41:08 上午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSe
ssion handleChannelInit
資訊: Socket registration request sent.
五月 31, 2021 1:41:08 上午 org.elasql.remote.groupcomm.client.BatchSpcSender run
資訊: start batching-request worker thread (batch size = 20)
五月 31, 2021 1:41:08 上午 org.vanilladb.core.util.PropertiesLoader getPropertyA
sString
警告: can't find property: org.vanilladb.bench.StatisticMgr.OUTPUT_DIR, using de
fault value: null
五月 31, 2021 1:41:08 上午 org.elasql.bench.ElasqlBench benchmark
資訊: checking the database on the server...
五月 31, 2021 1:41:08 上午 org.elasql.bench.ElasqlBench benchmark
資訊: database check passed.
五月 31, 2021 1:41:08 上午 org.elasql.bench.ElasqlBench benchmark
資訊: creating 200 emulators...
五月 31, 2021 1:41:08 上午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSe
ssion handleRegisterSocket
資訊: Socket registration completed. (/127.0.0.1:30000)
五月 31, 2021 1:41:09 上午 org.elasql.bench.benchmarks.ycsb.rte.SingleTableGoogl
eParamGen <clinit>
資訊: Use single-table Google YCSB generators (Read-write tx ratio: 0.500000, di
stributed tx ratio: 0.500000, 2 records/tx, 1 remote records/dist. tx, data size
: 2000000, google trace file: C:\Users\PinYu\exported-jars\google-workloads-2min
-3days.csv, google trace length: 2160)
五月 31, 2021 1:41:09 上午 org.elasql.bench.ElasqlBench benchmark
資訊: waiting for connections...
五月 31, 2021 1:41:10 上午 org.elasql.bench.ElasqlBench benchmark
資訊: start benchmarking.
五月 31, 2021 1:41:10 上午 org.elasql.bench.ElasqlBench benchmark
資訊: warm up period finished.
五月 31, 2021 1:41:10 上午 org.elasql.bench.ElasqlBench benchmark
資訊: start recording results...
```

Client console

```
資訊: start benchmarking.
五月 31, 2021 1:41:10 上午 org.elasql.bench.ElasqlBench benchmark
資訊: warm up period finished.
五月 31, 2021 1:41:10 上午 org.elasql.bench.ElasqlBench benchmark
資訊: start recording results...
Not replaying. Current replay point: -86
Not replaying. Current replay point: -85
Not replaying. Current replay point: -80
Not replaying. Current replay point: -75
Not replaying. Current replay point: -70
Not replaying. Current replay point: -65
Not replaying. Current replay point: -60
Not replaying. Current replay point: -55
Not replaying. Current replay point: -50
Not replaying. Current replay point: -45
Not replaying. Current replay point: -40
Not replaying. Current replay point: -35
Not replaying. Current replay point: -30
Not replaying. Current replay point: -25
Not replaying. Current replay point: -20
Not replaying. Current replay point: -15
Not replaying. Current replay point: -10
Not replaying. Current replay point: -5
Replaying. Current replay point: 0
Replaying. Current replay point: 4
Replaying. Current replay point: 9
Replaying. Current replay point: 14
Replaying. Current replay point: 19
Replaying. Current replay point: 24
Replaying. Current replay point: 29
Replaying. Current replay point: 34
Replaying. Current replay point: 39
Replaying. Current replay point: 44
Replaying. Current replay point: 49
Replaying. Current replay point: 54
Replaying. Current replay point: 59
Replaying. Current replay point: 64
Replaying. Current replay point: 69
Replaying. Current replay point: 74
Replaying. Current replay point: 79
Replaying. Current replay point: 84
Replaying. Current replay point: 89
Replaying. Current replay point: 94
Replaying. Current replay point: 99
Replaying. Current replay point: 104
Replaying. Current replay point: 109
Replaying. Current replay point: 114
Replaying. Current replay point: 119
Replaying. Current replay point: 124
Replaying. Current replay point: 129
Replaying. Current replay point: 134
Replaying. Current replay point: 139
Replaying. Current replay point: 144
Replaying. Current replay point: 149
Replaying. Current replay point: 154
Replaying. Current replay point: 159
Replaying. Current replay point: 164
Replaying. Current replay point: 169
Replaying. Current replay point: 174
Replaying. Current replay point: 179
Replaying. Current replay point: 184
Replaying. Current replay point: 189
Replaying. Current replay point: 194
Replaying. Current replay point: 199
Replaying. Current replay point: 204
五月 31, 2021 1:46:10 上午 org.elasql.bench.ElasqlBench benchmark
資訊: benchmark preiod finished. Stoping RTEs...
五月 31, 2021 1:46:11 上午 org.vanilladb.bench.StatisticMgr outputReport
資訊: Finnish creating tpcc benchmark report
五月 31, 2021 1:46:11 上午 org.elasql.bench.ElasqlBench benchmark
資訊: benchmark process finished.
```

Outline

- What is ElaSQLBench & ElaSQL
- Recap: Target Workloads
- Let’s Run a Benchmark on ElaSQL
  - Package code
  - Load Data
  - Workload configuration
  - Start Benchmarking
- Schedule: Next?

Timeline

5/20 Announce the assigned reading

5/27 Announce the details of the Final Project

5/31 Introduction to ElaSQL and ElaSQLBench

Next: 6/7 Walkthrough the Codebase of ElaSQL & Hermes

6/21、6/24 Final Project Presentation


Good Luck !