ElaSQL

Getting Started

June 12th 2021

elasql.org

Outline

- Introduction to ElaSQL project
- How to test/benchmark the system?

Outline

- Introduction to ElaSQL project
  - What is ElaSQL?
  - Architecture
  - Design & Key Features
  - Implemented Systems & Papers
  - The Sequencer
- How to test/benchmark the system?

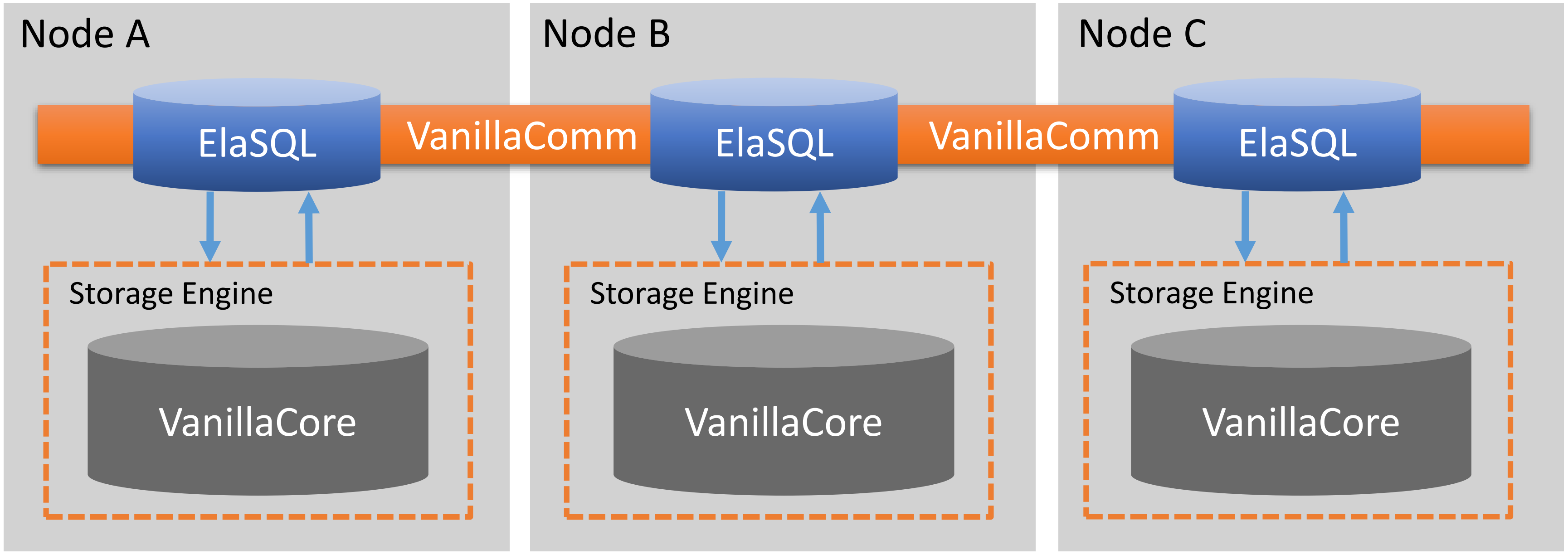
ElaSQL

- ElaSQL is a distributed relational database system that aims to provide
  - high scalability
  - high availability
  - elasticity

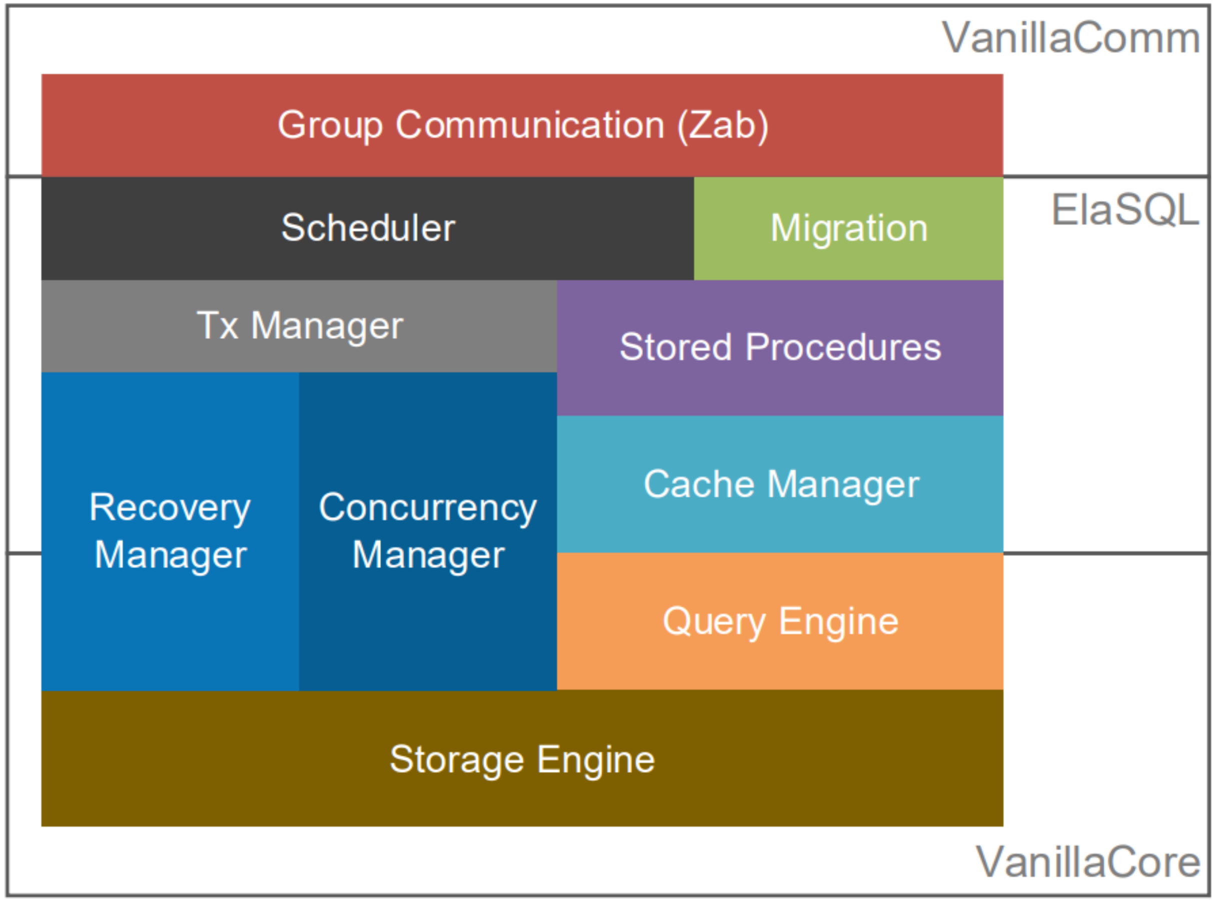
- ElaSQL can be tested with ElaSQL-Bench, which is a benchmark tool that generate pressure to the system.
  - Currently, this is the only way to test ElaSQL.

The Relationship with VanillaDB

- ElaSQL is built on top of two projects of VanillaDB
  - VanillaComm: as the communication module
  - VanillaCore: as the storage engine for each machine



Architecture inside a Machine



A Deterministic Database System

- ElaSQL is a deterministic database system, which is based on the idea of the following paper:
  - Thomson, Alexander, and Daniel J. Abadi. "The case for determinism in database systems." *Proceedings of the VLDB Endowment* 3.1-2 (2010): 70-80.

- With determinism, ElaSQL can ensure a database always reach the same state from the same initial state with the same sequence of requests.

Key Features

- Strong Consistency with high availability
  - ElaSQL uses determinism to ensure consistency without relying on two phase commit.
- High Scalability
  - ElaSQL partitions a database to distribute the loads to multiple machines.
- Elasticity
  - ElaSQL implements several data migration and re-partitioning algorithm to ensure that data partitions are always up to date.

Implemented Systems

- Since ElaSQL is a research prototype, we have implemented several algorithms and systems proposed in research papers in ElaSQL.

- Please check this list for available systems and algorithms and corresponding papers.

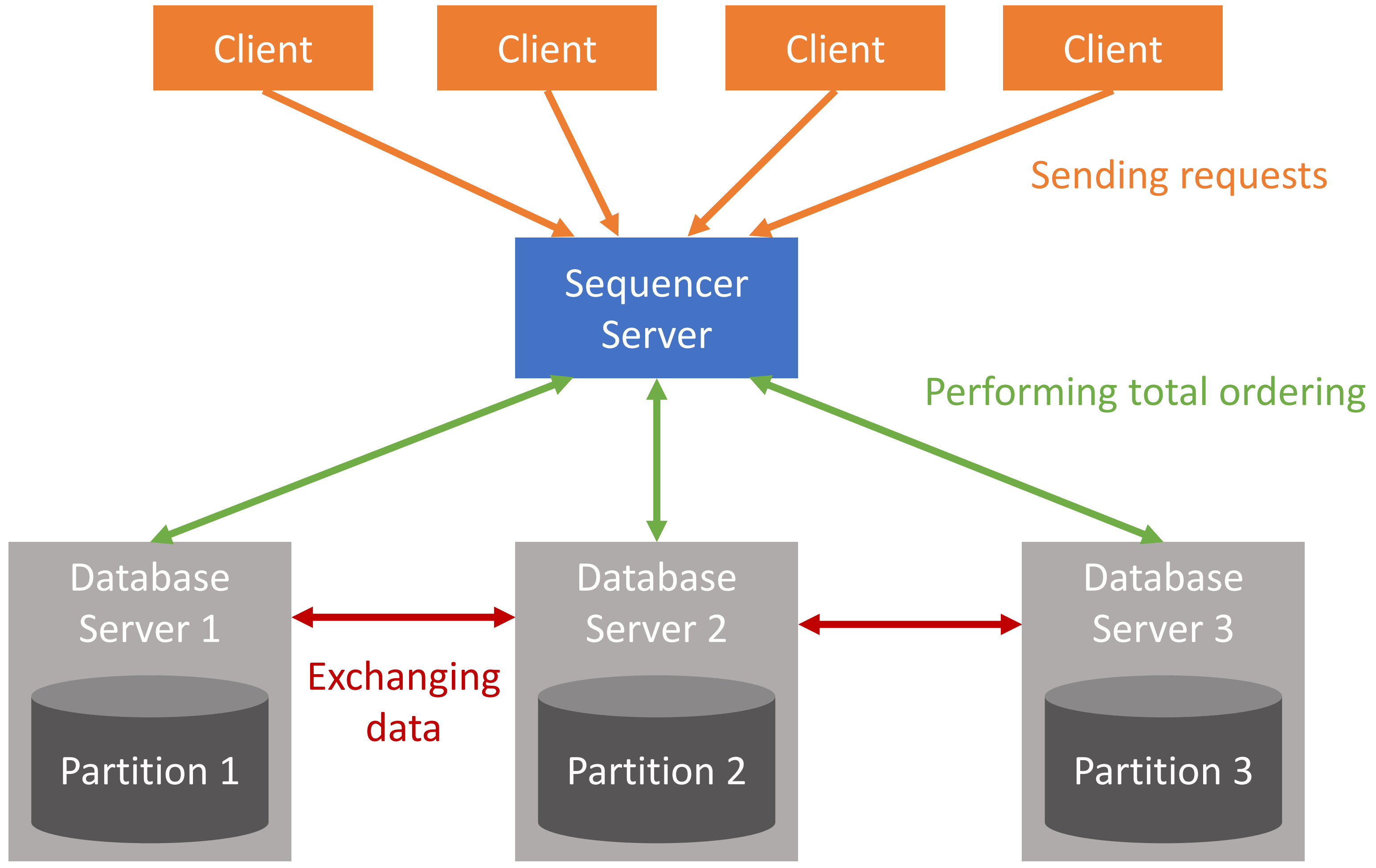
The Sequencer (ZAB Leader)

- A deterministic database system requires a total-ordering protocol to ensure the order of transactions across machines in advance.

- We implement Zookeeper Atomic Message Broadcast (ZAB) Protocol for this.
  - Which requires a machine to be the leader.
  - We call the leader as **the sequencer**.

- In our design, the sequencer is one of servers in a cluster, but it does not have database functionality.

Message Flow



Outline

- Introduction to ElaSQL project
- How to test/benchmark the system?
  - Let’s meet ElaSQL-Bench
  - Setting up development environment
  - Testing inside a Java IDE
  - Testing with runnable JARs
  - Testing in a cluster

Outline

- Introduction to ElaSQL project
- How to test/benchmark the system?
  - Let’s meet ElaSQL-Bench
  - Setting up development environment
  - Testing inside a Java IDE
  - Testing with runnable JARs
  - Testing in a cluster

ElaSQL-Bench

- In order to test how ElaSQL performs under an extreme circumstance, we implement a benchmark tool.
  - Which is based on another project, VanillaBench.

- This project includes two standard benchmarks:
  - The TPC-C Benchmark
  - The Yahoo! Cloud Serving Benchmark (YCSB)

Outline

- Introduction to ElaSQL project
- How to test/benchmark the system?
  - Let’s meet ElaSQL-Bench
  - Setting up development environment
  - Testing inside a Java IDE
  - Testing with runnable JARs
  - Testing in a cluster

Prerequisite

- We assume that you have the following programs in your environment.
  - Java Development Kit (JDK) **8**
    - We found some problems when running with JDK 10+. You may try, but there is no guarantee to work.
  - Eclipse
    - You may use another IDE, but we will demonstrate the following tasks in Eclipse.
  - Git
  - Bash

Steps to Setup Your Dev. Env.

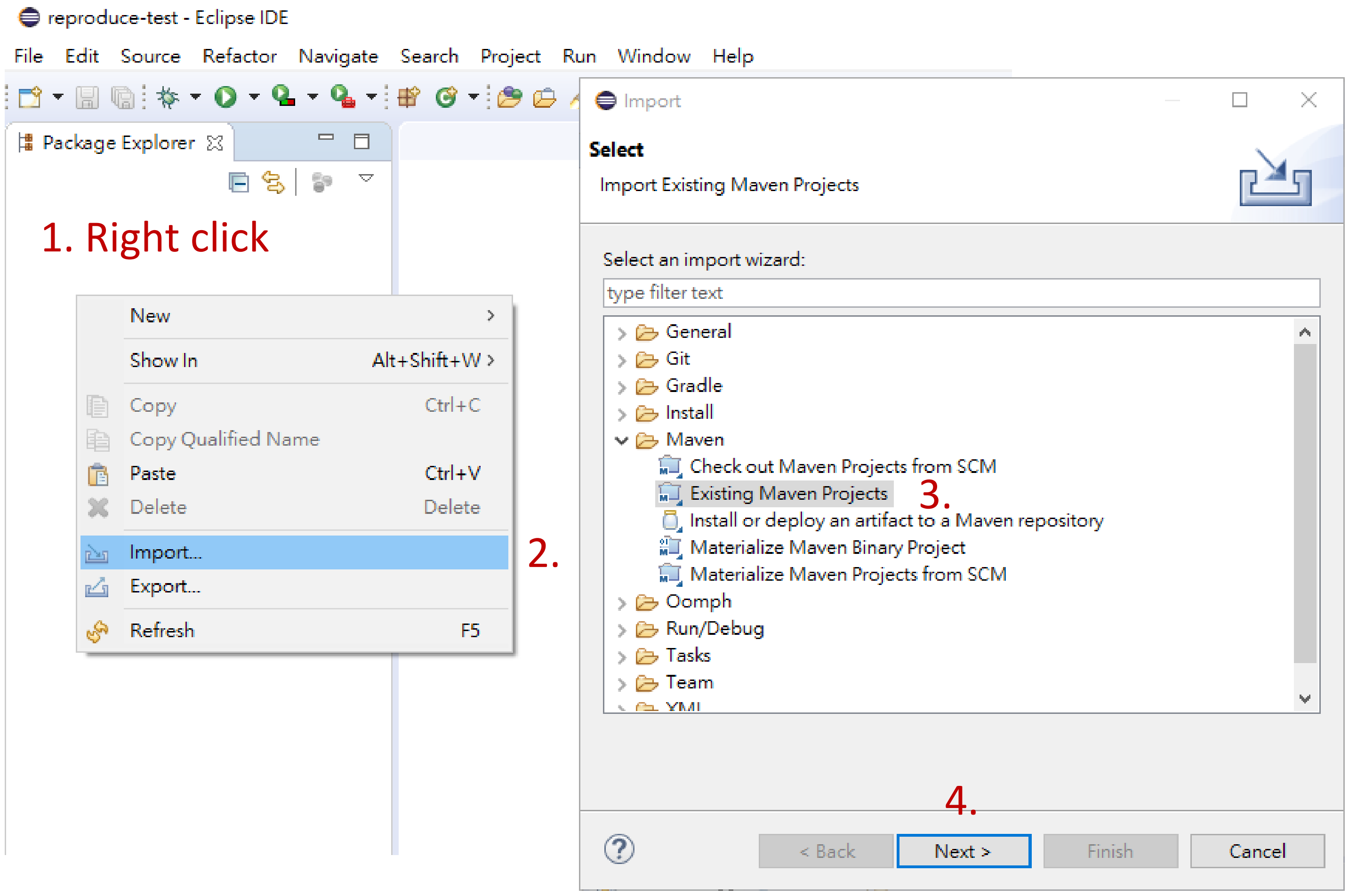
1. Clone ElaSQL and ElaSQL-Bench
2. Import the projects to Eclipse
3. Done

Cloning the Project

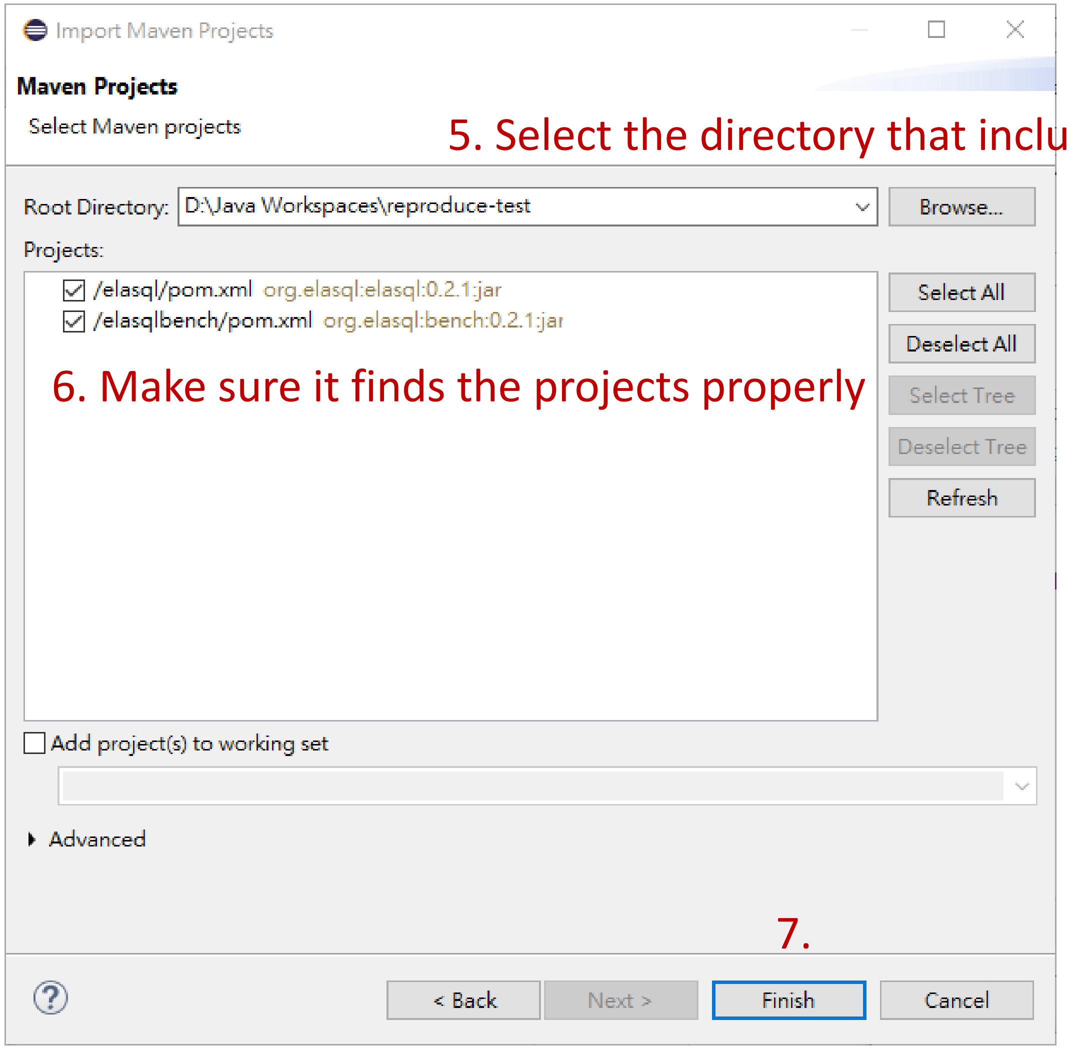
- Clone the following projects:
  - ElaSQL: https://github.com/elasql/elasql
  - ElaSQL-Bench: https://github.com/elasql/elasqlbench

- Checkout the branch you need
  - The default branch is “**master**”.
  - However, if you want to reproduce certain experiments, you may want to checkout other branches.
  - For example, to reproduce MgCrab experiments, you may need to checkout “**reproduce/mgcrab**” branch.

Importing into Eclipse



Importing into Eclipse



Import Maven Projects

Maven Projects

Select Maven projects

5. Select the directory that includes the projects

Root Directory: D:\Java Workspaces\reproduce-test

Projects:

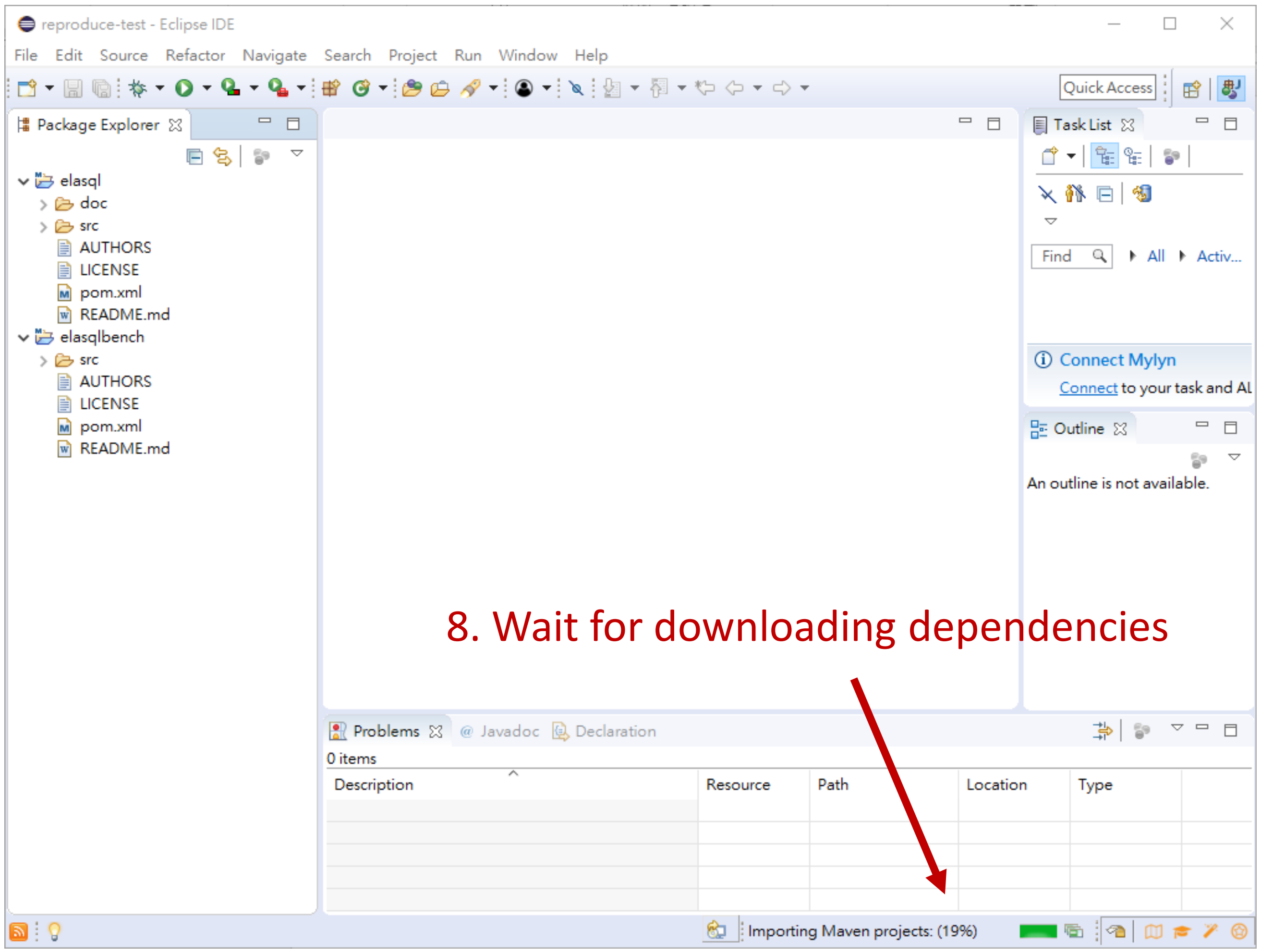
- [✓] /elasql/pom.xml org.elasql:elasql:0.2.1:jar
- [✓] /elasqlbench/pom.xml org.elasql:bench:0.2.1:jar

6. Make sure it finds the projects properly

[ ] Add project(s) to working set

7.

Importing into Eclipse



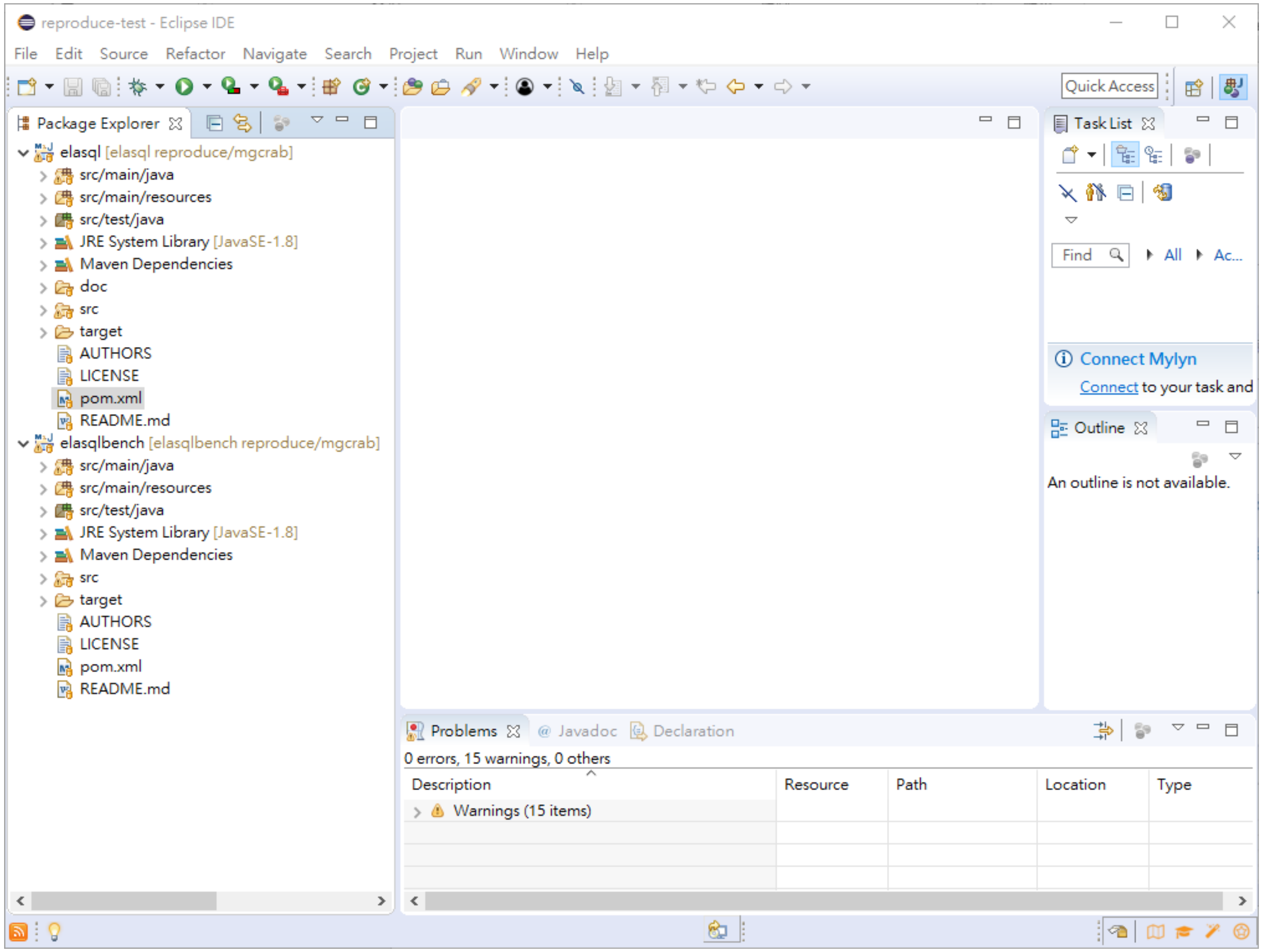
An outline is not available.

0 items

Importing Maven projects: (19%)

8. Wait for downloading dependencies

Done



reproduce-test - Eclipse IDE

elasql [elasql reproduce/mgcrab]
- src/main/java
- src/main/resources
- src/test/java
- JRE System Library [JavaSE-1.8]
- Maven Dependencies
- doc
- src
- target
- AUTHORS
- LICENSE
- pom.xml
- README.md

elasqlbench [elasqlbench reproduce/mgcrab]
- src/main/java
- src/main/resources
- src/test/java
- JRE System Library [JavaSE-1.8]
- Maven Dependencies
- src
- target
- AUTHORS
- LICENSE
- pom.xml
- README.md

Connect Mylyn

Connect to your task and

An outline is not available.

0 errors, 15 warnings, 0 others

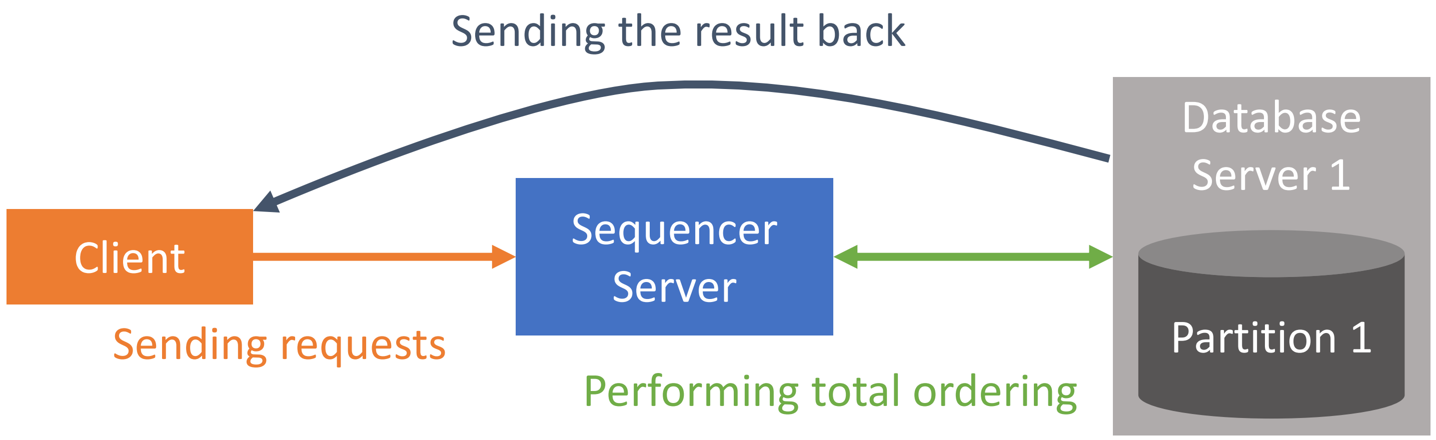
Description	Resource	Path	Location	Type
Warnings (15 items)				

Outline

- Introduction to ElaSQL project
- How to test/benchmark the system?
  - Let’s meet ElaSQL-Bench
  - Setting up development environment
  - Testing inside a Java IDE
  - Testing with runnable JARs
  - Testing in a cluster

Testing Environments

- To launch a benchmarking test, at least three processes must be launched.
  - 1 Sequencer Server (the ZAB leader)
  - 1 Database Server
    - Adding more database servers can increase throughput
  - 1 Benchmark Client
    - Adding more clients can generate higher pressure to the system



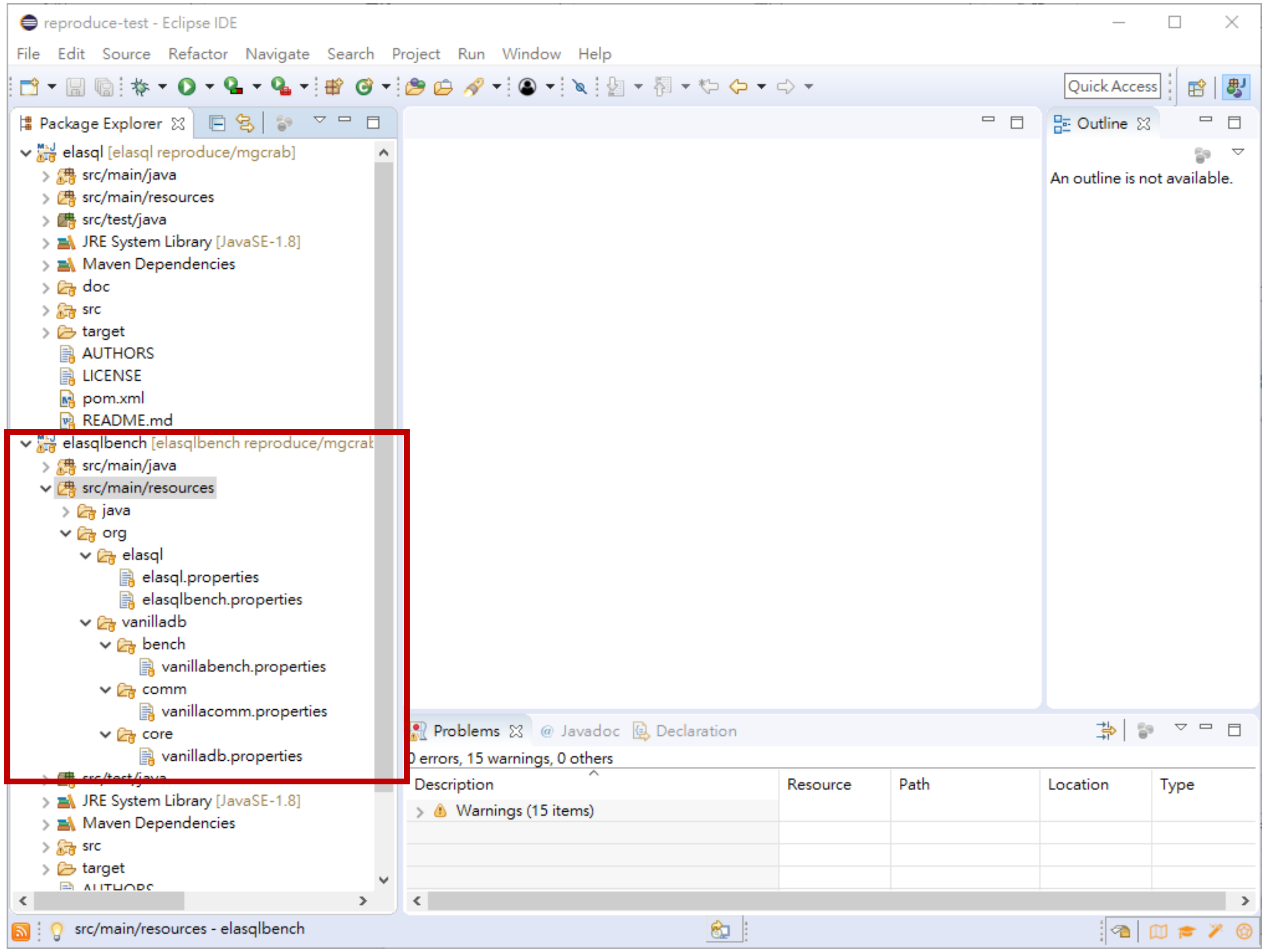
Testing inside Eclipse

1. Setup the properties files
   - Which includes the configurations for ElaSQL and ElaSQL-Bench
2. Setup run configurations
3. Loading a testbed
   1. Launch servers
   2. Launch clients
4. Benchmarking
   1. Launch servers
   2. Launch clients

Testing inside Eclipse

1. Setup the properties files
   - Which includes the configurations for ElaSQL and ElaSQL-Bench
2. Setup run configurations
3. Loading a testbed
   1. Launch servers
   2. Launch clients
4. Benchmarking
   1. Launch servers
   2. Launch clients

The Properties Files


reproduce-test - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Outline
Problems
Javadoc
Declaration

An outline is not available.

```
elasql [elasql reproduce/mgcrab]
  src/main/java
  src/main/resources
  src/test/java
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  doc
  src
  target
  AUTHORS
  LICENSE
  pom.xml
  README.md
elasqlbench [elasqlbench reproduce/mgcrab
  src/main/java
  src/main/resources
    java
    org
      elasql
        elasql.properties
        elasqlbench.properties
      vanilladb
        bench
          vanillabench.properties
        comm
          vanillacomm.properties
        core
          vanilladb.properties
  src/test/java
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  src
  target
  AUTHORS
```

0 errors, 15 warnings, 0 others

Description	Resource	Path	Location	Type
Warnings (15 items)				

src/main/resources - elasqlbench

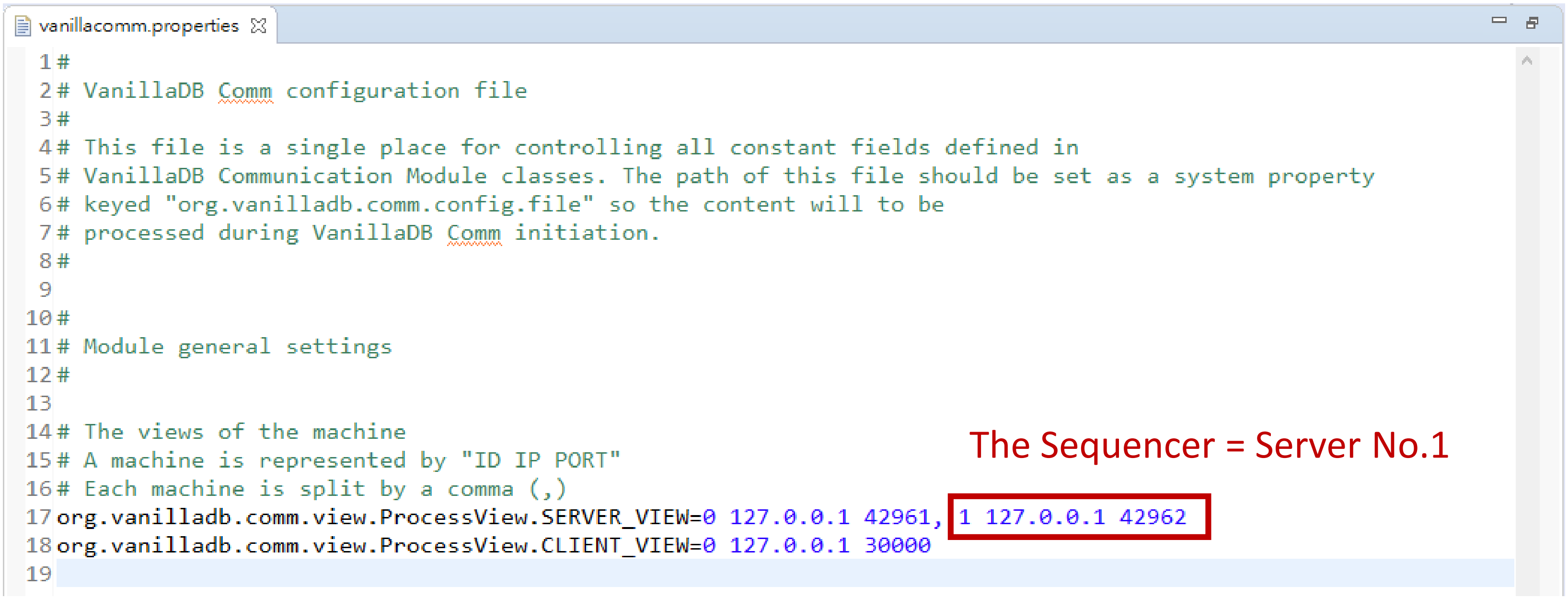
Setting Up Network Addresses

- ElaSQL uses VanillaComm to communicate through networks.

- We need to tell VanillaComm where to find all the machines (including servers and clients).
  - The addresses should be put in vanillacomm.properties

Setting Up Network Addresses

- Here is an example to setup the addresses for 2 servers and 1 client.
  - The last server will become the sequencer.



vanillacomm.properties

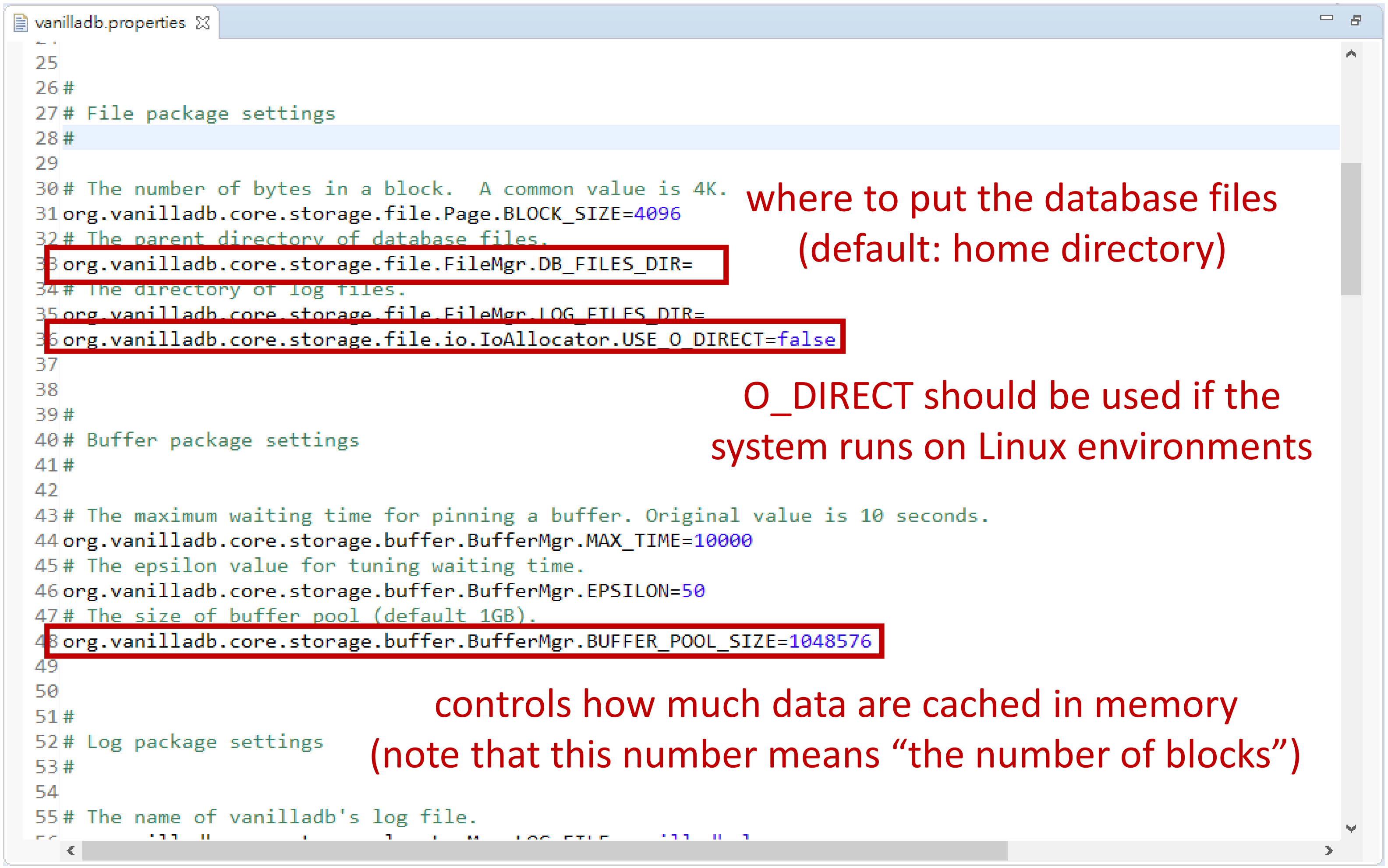
```
#
# VanillaDB Comm configuration file
#
# This file is a single place for controlling all constant fields defined in
# VanillaDB Communication Module classes. The path of this file should be set as a system property
# keyed "org.vanilladb.comm.config.file" so the content will to be
# processed during VanillaDB Comm initiation.
#


#
# Module general settings
#

# 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
org.vanilladb.comm.view.ProcessView.CLIENT_VIEW=0 127.0.0.1 30000

```

The Sequencer = Server No.1

Setting Up The Storage Engine

- ElaSQL uses VanillaCore as a storage engine to store data on each machine.
  - vanillacore.properties includes the configurations for the storage engine.
- Most configurations have been tuned for benchmarking.
  - Only some of them should be checked carefully.

Setting Up The Storage Engine



```
#
# File package settings
#

# The number of bytes in a block.  A common value is 4K.
org.vanilladb.core.storage.file.Page.BLOCK_SIZE=4096
# The parent directory of database files.
org.vanilladb.core.storage.file.FileMgr.DB_FILES_DIR=
# The directory of log files.
org.vanilladb.core.storage.file.FileMgr.LOG_FILES_DIR=
org.vanilladb.core.storage.file.io.IoAllocator.USE_O_DIRECT=false


#
# Buffer package settings
#

# The maximum waiting time for pinning a buffer. Original value is 10 seconds.
org.vanilladb.core.storage.buffer.BufferMgr.MAX_TIME=10000
# The epsilon value for tuning waiting time.
org.vanilladb.core.storage.buffer.BufferMgr.EPSILON=50
# The size of buffer pool (default 1GB).
org.vanilladb.core.storage.buffer.BufferMgr.BUFFER_POOL_SIZE=1048576


#
# Log package settings
#

# The name of vanilladb's log file.
```

where to put the database files
(default: home directory)

O_DIRECT should be used if the
system runs on Linux environments

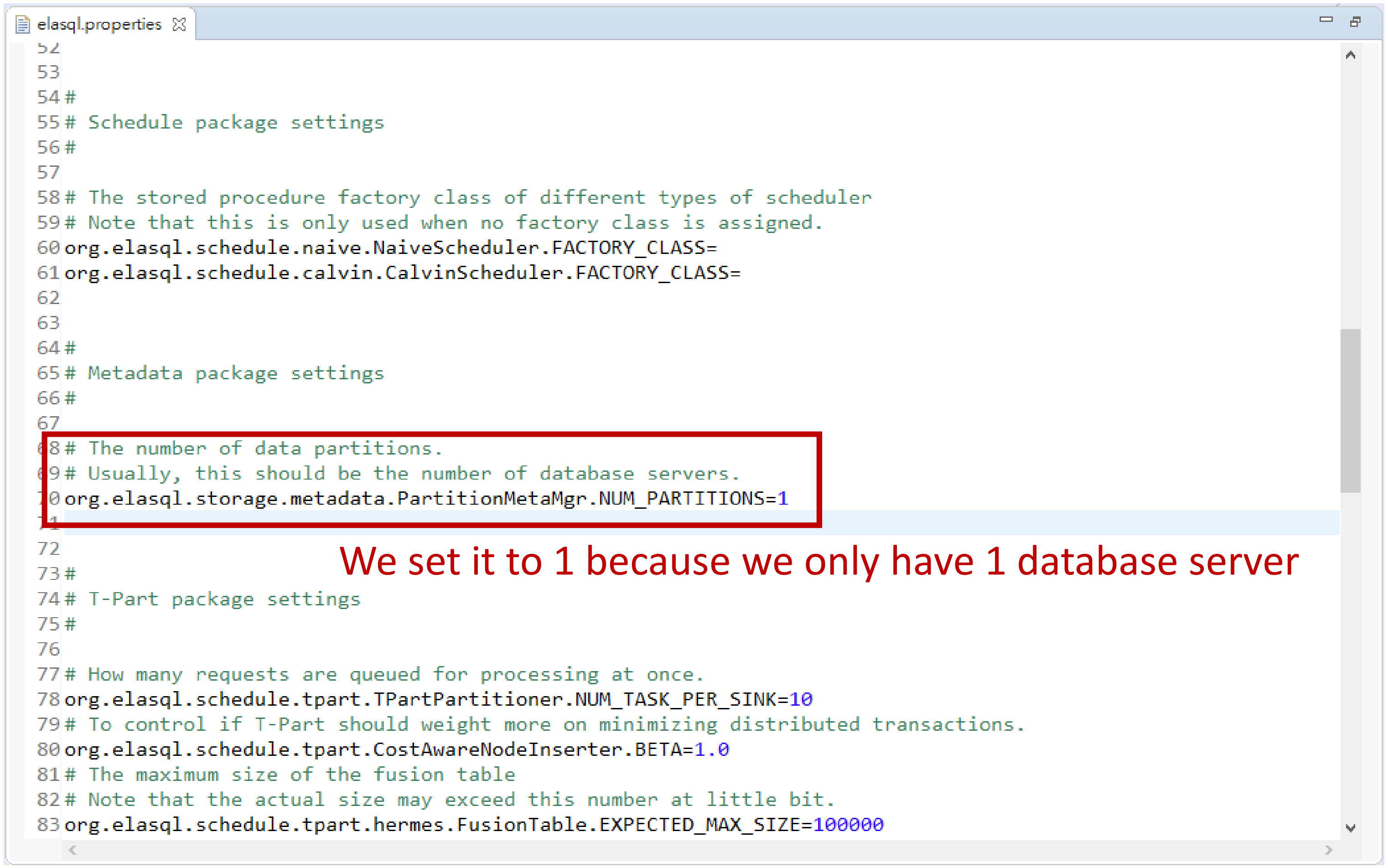
controls how much data are cached in memory
(note that this number means “the number of blocks”)

Setting Up ElaSQL (The Distributed Modules)

- ElaSQL also has many configurations:
  - How many data partitions are there?
  - Which system to run? Calvin? Hermes?
  - Which data migration algorithm to use?

- All these are put in elasql.properties
  - The file contains a comprehensive explanations for each parameter, so we will not go through all the parameters here.

Setting Up ElaSQL (The Distributed Modules)



```
#
# Schedule package settings
#

# The stored procedure factory class of different types of scheduler
# Note that this is only used when no factory class is assigned.
org.elasql.schedule.naive.NaiveScheduler.FACTORY_CLASS=
org.elasql.schedule.calvin.CalvinScheduler.FACTORY_CLASS=


#
# Metadata package settings
#

# The number of data partitions.
# Usually, this should be the number of database servers.
org.elasql.storage.metadata.PartitionMetaMgr.NUM_PARTITIONS=1


#
# T-Part package settings
#

# How many requests are queued for processing at once.
org.elasql.schedule.tpart.TPartPartitioner.NUM_TASK_PER_SINK=10
# To control if T-Part should weight more on minimizing distributed transactions.
org.elasql.schedule.tpart.CostAwareNodeInserter.BETA=1.0
# The maximum size of the fusion table
# Note that the actual size may exceed this number at little bit.
org.elasql.schedule.tpart.hermes.FusionTable.EXPECTED_MAX_SIZE=100000
```

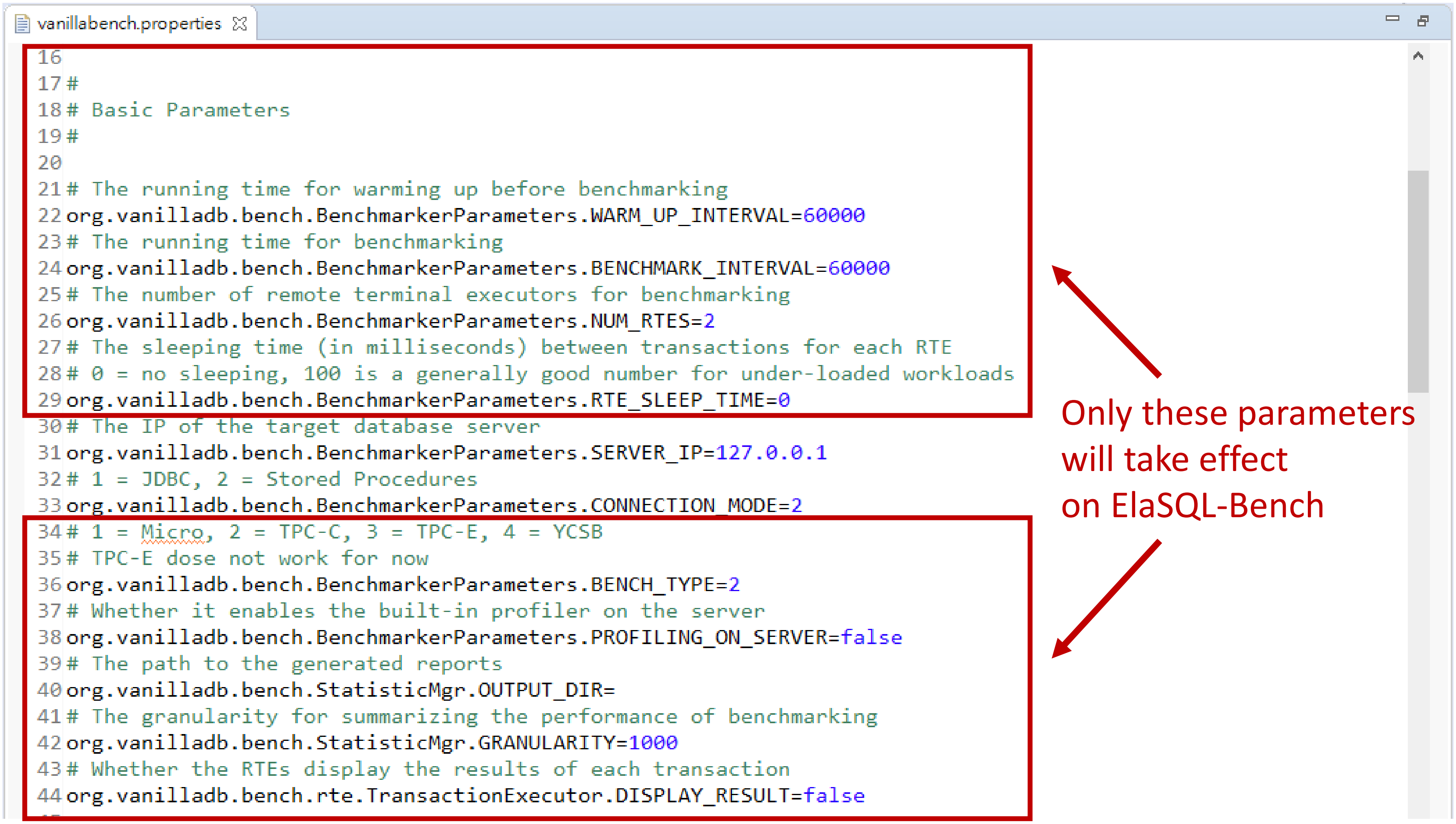
We set it to 1 because we only have 1 database server

Setting Up a Benchmarking Test

- ElaSQL-Bench reuses the codebase of VanillaBench, which is a benchmarking tool for single-node DBMS.

- So, the configurations are separated in two files:
  - vanillabench.properties (only the basic configrations)
  - elasqlbench.properties

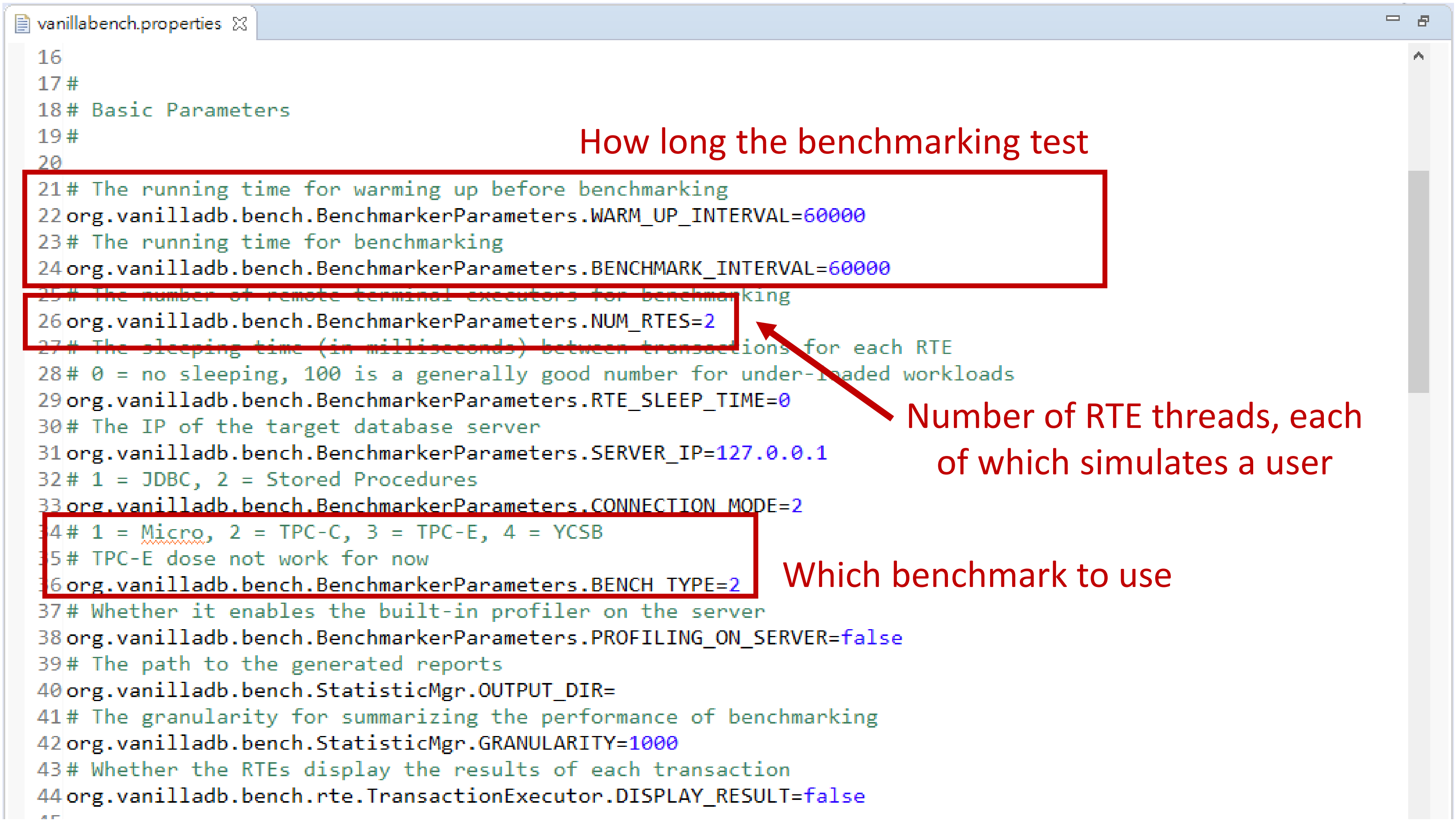
vanillabench.properties



```
#
# Basic Parameters
#

# The running time for warming up before benchmarking
org.vanilladb.bench.BenchmarkerParameters.WARM_UP_INTERVAL=60000
# The running time for benchmarking
org.vanilladb.bench.BenchmarkerParameters.BENCHMARK_INTERVAL=60000
# The number of remote terminal executors for benchmarking
org.vanilladb.bench.BenchmarkerParameters.NUM_RTES=2
# The sleeping time (in milliseconds) between transactions for each RTE
# 0 = no sleeping, 100 is a generally good number for under-loaded workloads
org.vanilladb.bench.BenchmarkerParameters.RTE_SLEEP_TIME=0
# The IP of the target database server
org.vanilladb.bench.BenchmarkerParameters.SERVER_IP=127.0.0.1
# 1 = JDBC, 2 = Stored Procedures
org.vanilladb.bench.BenchmarkerParameters.CONNECTION_MODE=2
# 1 = Micro, 2 = TPC-C, 3 = TPC-E, 4 = YCSB
# TPC-E dose not work for now
org.vanilladb.bench.BenchmarkerParameters.BENCH_TYPE=2
# Whether it enables the built-in profiler on the server
org.vanilladb.bench.BenchmarkerParameters.PROFILING_ON_SERVER=false
# The path to the generated reports
org.vanilladb.bench.StatisticMgr.OUTPUT_DIR=
# The granularity for summarizing the performance of benchmarking
org.vanilladb.bench.StatisticMgr.GRANULARITY=1000
# Whether the RTEs display the results of each transaction
org.vanilladb.bench.rte.TransactionExecutor.DISPLAY_RESULT=false
```

Only these parameters will take effect on ElaSQL-Bench

vanillabench.properties



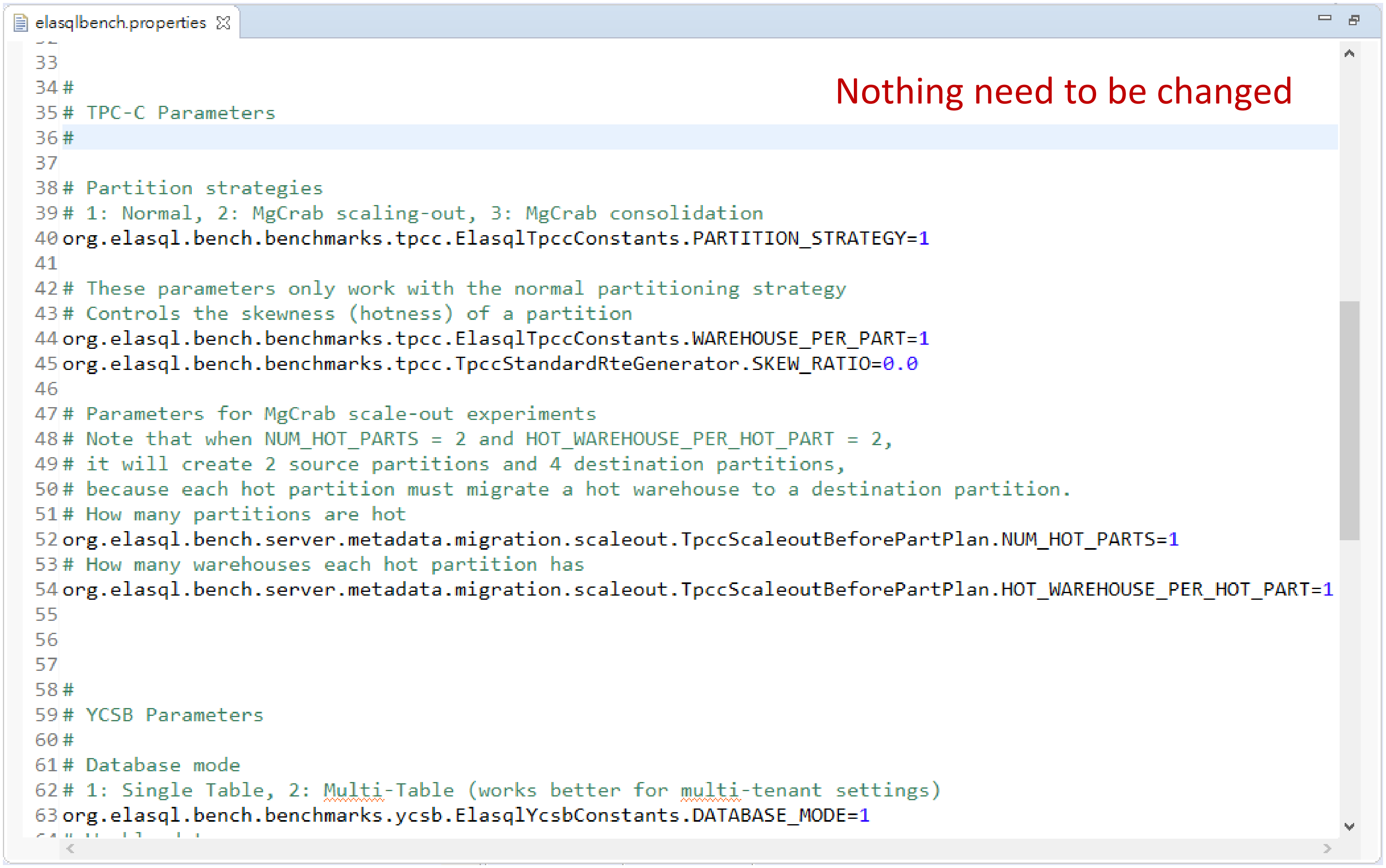
```

#
# Basic Parameters
#

# The running time for warming up before benchmarking
org.vanilladb.bench.BenchmarkerParameters.WARM_UP_INTERVAL=60000
# The running time for benchmarking
org.vanilladb.bench.BenchmarkerParameters.BENCHMARK_INTERVAL=60000
# The number of remote terminal executors for benchmarking
org.vanilladb.bench.BenchmarkerParameters.NUM_RTES=2
# The sleeping time (in milliseconds) between transactions for each RTE
# 0 = no sleeping, 100 is a generally good number for under-loaded workloads
org.vanilladb.bench.BenchmarkerParameters.RTE_SLEEP_TIME=0
# The IP of the target database server
org.vanilladb.bench.BenchmarkerParameters.SERVER_IP=127.0.0.1
# 1 = JDBC, 2 = Stored Procedures
org.vanilladb.bench.BenchmarkerParameters.CONNECTION_MODE=2
# 1 = Micro, 2 = TPC-C, 3 = TPC-E, 4 = YCSB
# TPC-E dose not work for now
org.vanilladb.bench.BenchmarkerParameters.BENCH_TYPE=2
# Whether it enables the built-in profiler on the server
org.vanilladb.bench.BenchmarkerParameters.PROFILING_ON_SERVER=false
# The path to the generated reports
org.vanilladb.bench.StatisticMgr.OUTPUT_DIR=
# The granularity for summarizing the performance of benchmarking
org.vanilladb.bench.StatisticMgr.GRANULARITY=1000
# Whether the RTEs display the results of each transaction
org.vanilladb.bench.rte.TransactionExecutor.DISPLAY_RESULT=false
```

How long the benchmarking test

Number of RTE threads, each of which simulates a user

Which benchmark to use

elasqlbench.properties



Nothing need to be changed

```
#
# TPC-C Parameters
#

# Partition strategies
# 1: Normal, 2: MgCrab scaling-out, 3: MgCrab consolidation
org.elasql.bench.benchmarks.tpcc.ElasqlTpccConstants.PARTITION_STRATEGY=1

# These parameters only work with the normal partitioning strategy
# Controls the skewness (hotness) of a partition
org.elasql.bench.benchmarks.tpcc.ElasqlTpccConstants.WAREHOUSE_PER_PART=1
org.elasql.bench.benchmarks.tpcc.TpccStandardRteGenerator.SKEW_RATIO=0.0

# Parameters for MgCrab scale-out experiments
# Note that when NUM_HOT_PARTS = 2 and HOT_WAREHOUSE_PER_HOT_PART = 2,
# it will create 2 source partitions and 4 destination partitions,
# because each hot partition must migrate a hot warehouse to a destination partition.
# How many partitions are hot
org.elasql.bench.server.metadata.migration.scaleout.TpccScaleoutBeforePartPlan.NUM_HOT_PARTS=1
# How many warehouses each hot partition has
org.elasql.bench.server.metadata.migration.scaleout.TpccScaleoutBeforePartPlan.HOT_WAREHOUSE_PER_HOT_PART=1



#
# YCSB Parameters
#
# Database mode
# 1: Single Table, 2: Multi-Table (works better for multi-tenant settings)
org.elasql.bench.benchmarks.ycsb.ElasqlYcsbConstants.DATABASE_MODE=1
```

Testing inside Eclipse

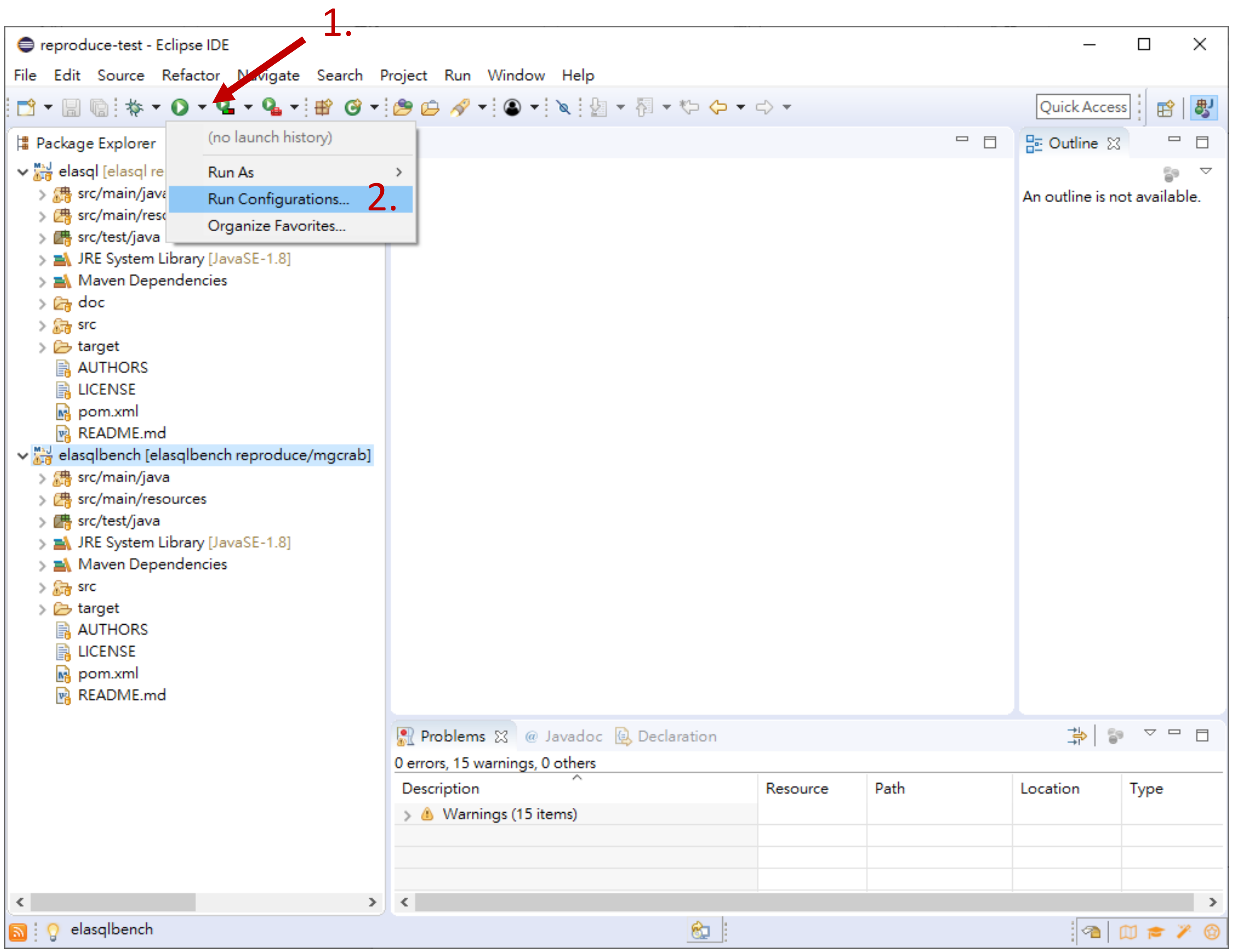
1. Setup the properties files
   - Which includes the configurations for ElaSQL and ElaSQL-Bench
2. Setup run configurations
3. Loading a testbed
   1. Launch servers
   2. Launch clients
4. Benchmarking
   1. Launch servers
   2. Launch clients

Run Configurations

- A run configuration configures how eclipse launch a Java process.

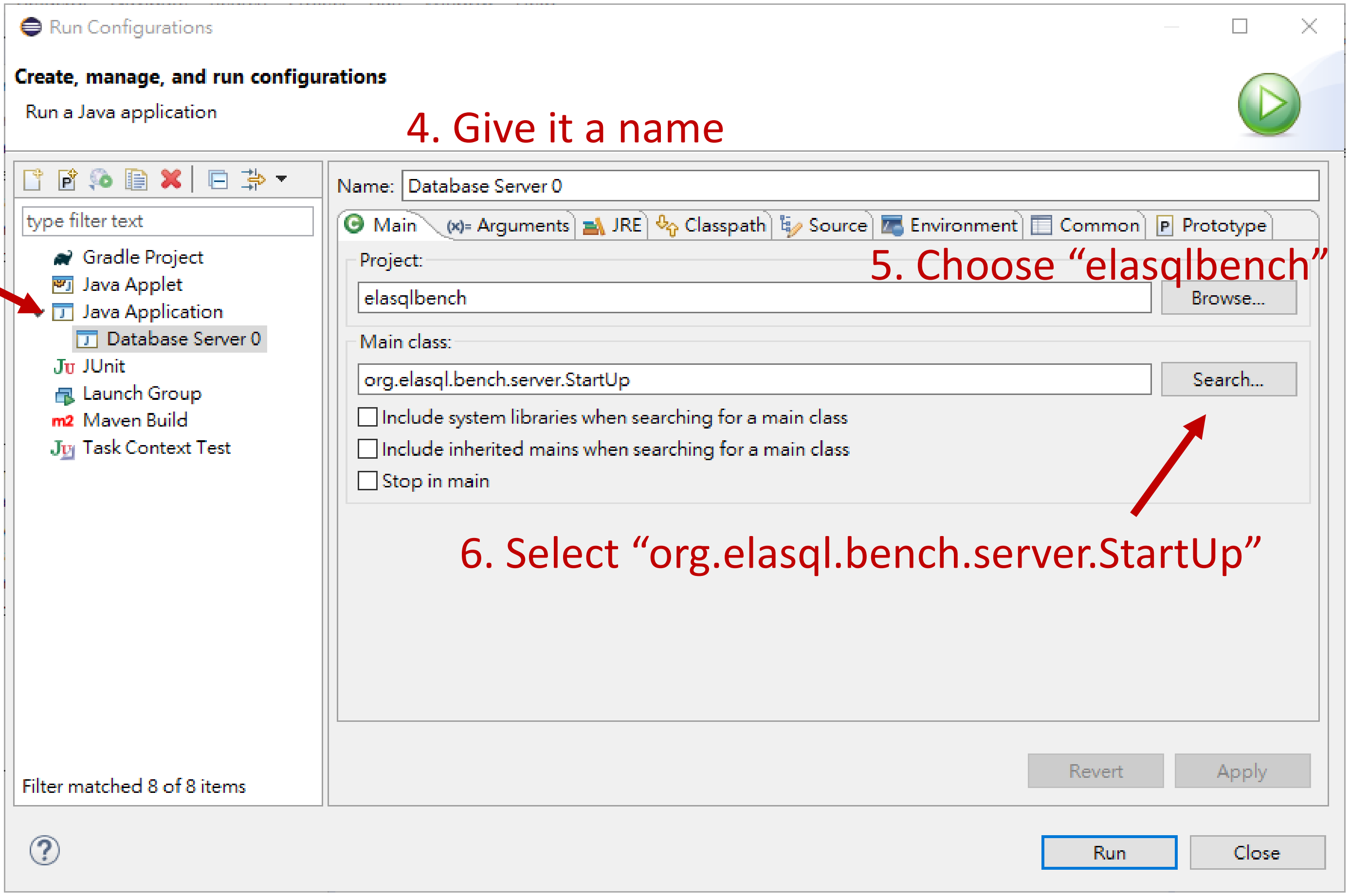
- Each process must have its own run configuration.
  - 3 configurations for a sequencer server, a database server, and a client.

Setting Up Run Configurations



Setting Up Run Configurations (A Database Server)

3. Double Click



Run Configurations

Create, manage, and run configurations

Run a Java application

4. Give it a name

type filter text

- Gradle Project
- Java Applet
- Java Application
  - Database Server 0
- JUnit
- Launch Group
- Maven Build
- Task Context Test

Filter matched 8 of 8 items

Name: Database Server 0

Main | Arguments | JRE | Classpath | Source | Environment | Common | Prototype

5. Choose "elasqlbench"

Project: elasqlbench [Browse...]

Main class: org.elasql.bench.server.StartUp [Search...]

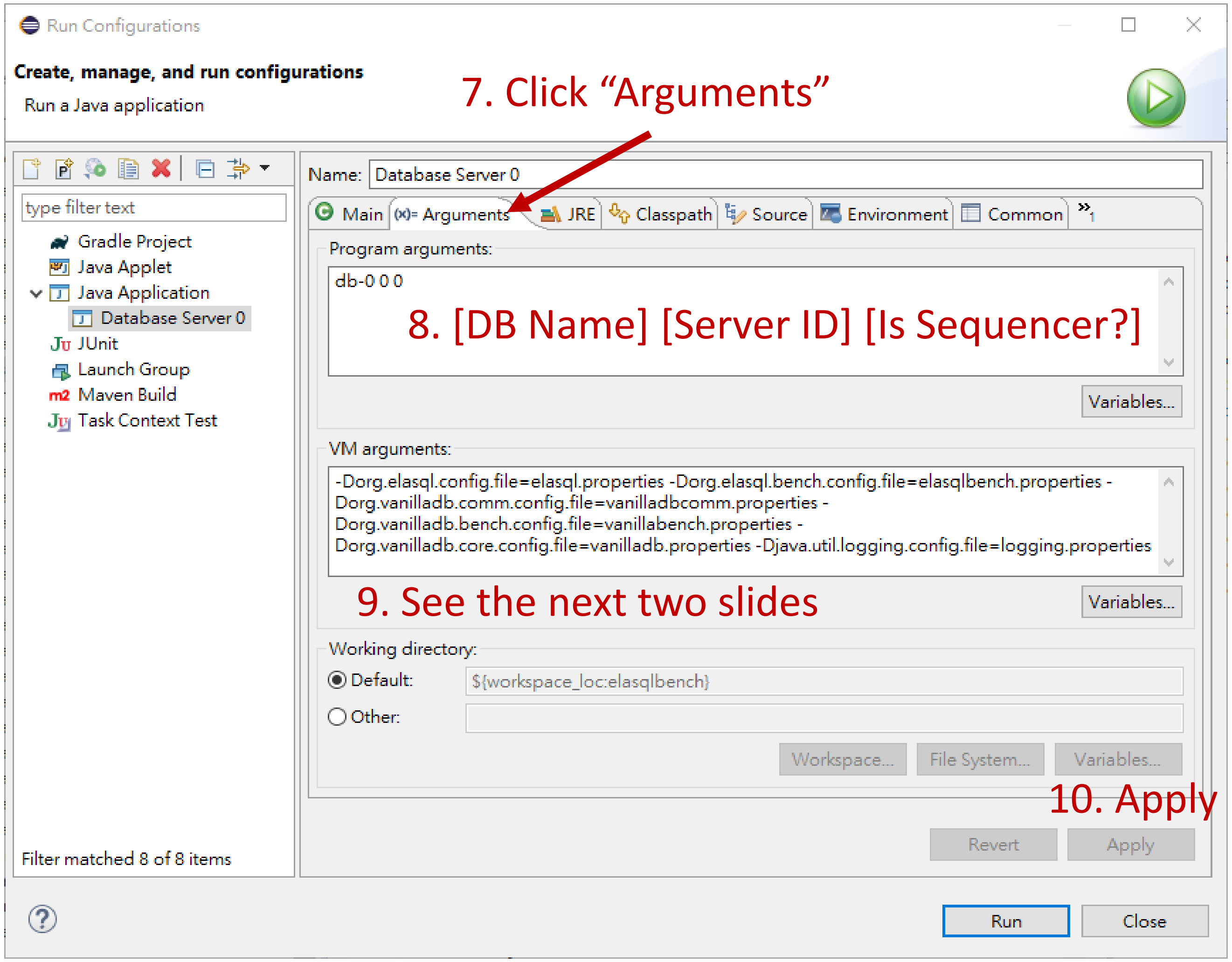
- [ ] Include system libraries when searching for a main class
- [ ] Include inherited mains when searching for a main class
- [ ] Stop in main

6. Select "org.elasql.bench.server.StartUp"

Revert | Apply

Run | Close

Setting Up Run Configurations (A Database Server)



Run Configurations

Create, manage, and run configurations

Run a Java application

7. Click “Arguments”

Name: Database Server 0

Program arguments:

```
db-0 0 0
```

8. [DB Name] [Server ID] [Is Sequencer?]

VM arguments:

```
-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
```

9. See the next two slides

Working directory:

[✓ Default:] ${workspace_loc:elasqlbench}

Other:

10. Apply

Program Arguments (For Servers)

- Program Arguments
  - DB Name: the database name
    - Note that if you run servers on the same machine, each server should have an unique name for its database.
  - Server ID: the ID of the server process
  - Is Sequencer: to set if it is running in sequencer mode.
    - The server with the greatest ID should turn this ON.

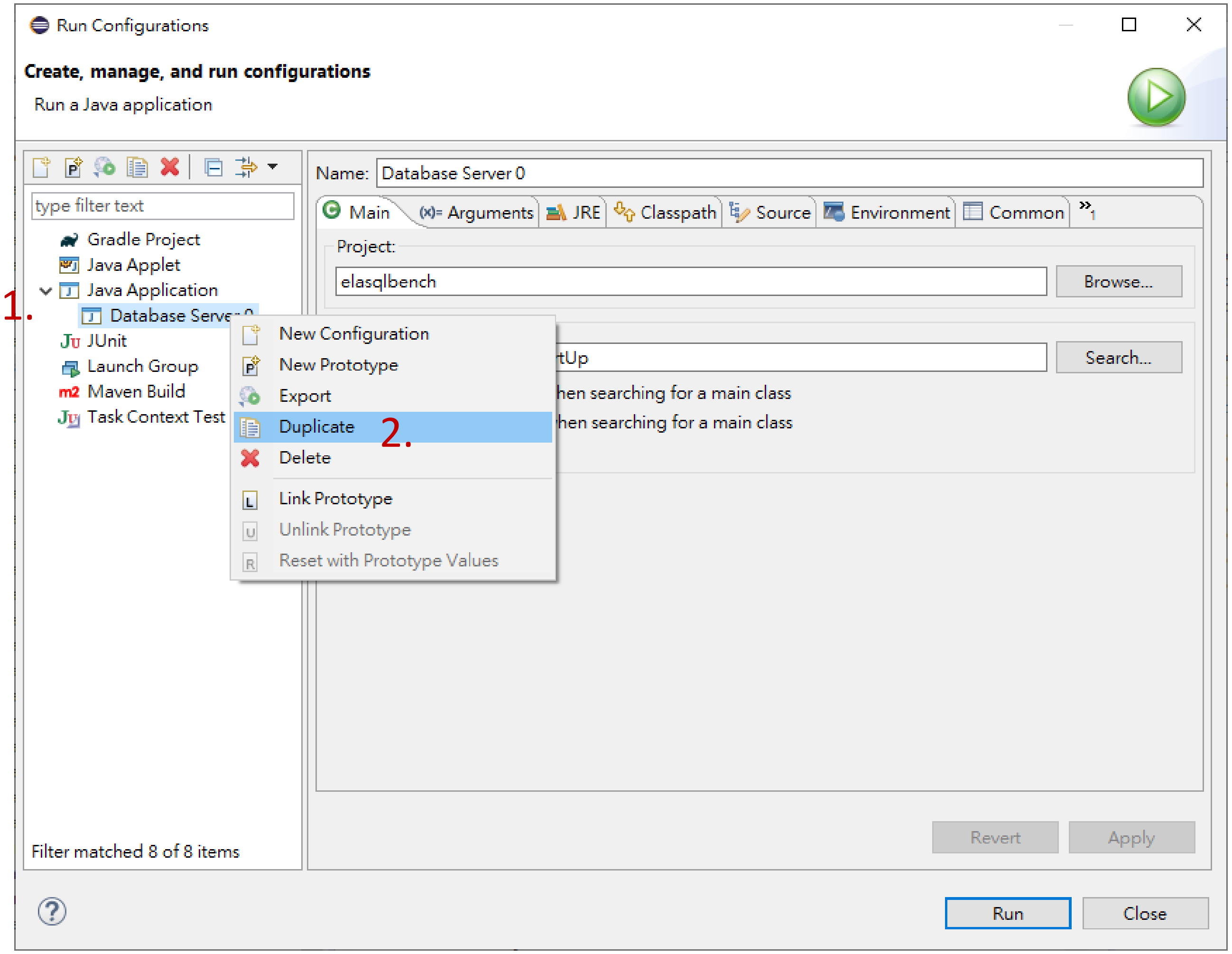
VM Arguments

- VM Arguments:
  - To tell ElaSQL where to find those properties files
  - Copy and paste this:

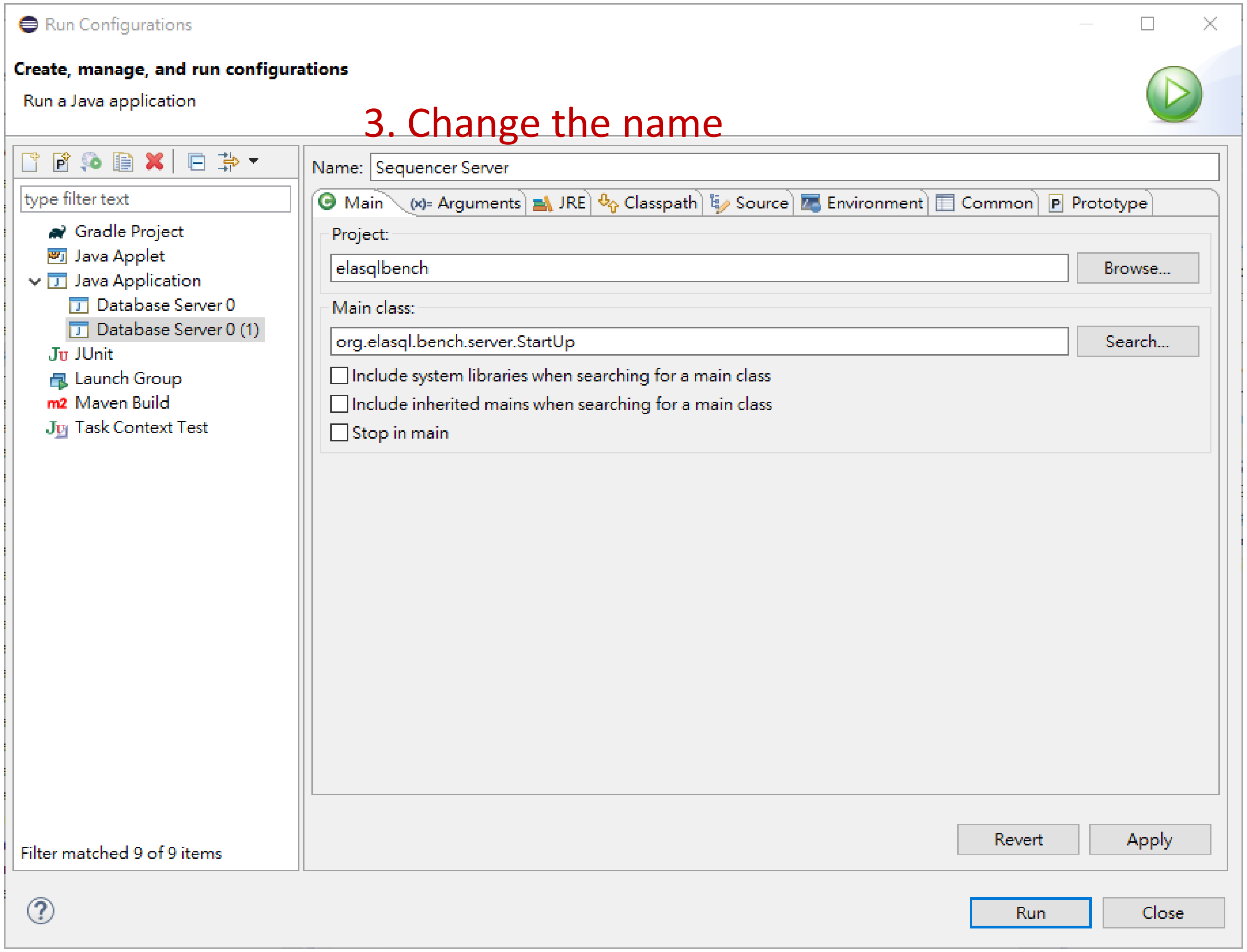
```
-Dorg.elasql.config.file=target/classes/org/elasql/elasql.properties
-Dorg.elasql.bench.config.file=target/classes/org/elasql/elasqlbench.properties
-Dorg.vanilladb.comm.config.file=target/classes/org/vanilladb/comm/vanillacomm.properties
-Dorg.vanilladb.bench.config.file=target/classes/org/vanilladb/bench/vanillabench.properties
-Dorg.vanilladb.core.config.file=target/classes/org/vanilladb/core/vanilladb.properties
-Djava.util.logging.config.file=target/classes/java/util/logging/logging.properties
```

- If you encounter any problem when copying the arguments from this slide, you can copy from here.

Setting Up Run Configurations (The Sequencer Server)



Setting Up Run Configurations (The Sequencer Server)



Run Configurations

Create, manage, and run configurations

Run a Java application

3. Change the name

type filter text

- Gradle Project
- Java Applet
- Java Application
  - Database Server 0
  - Database Server 0 (1)
- JUnit
- Launch Group
- Maven Build
- Task Context Test

Filter matched 9 of 9 items

Name: Sequencer Server

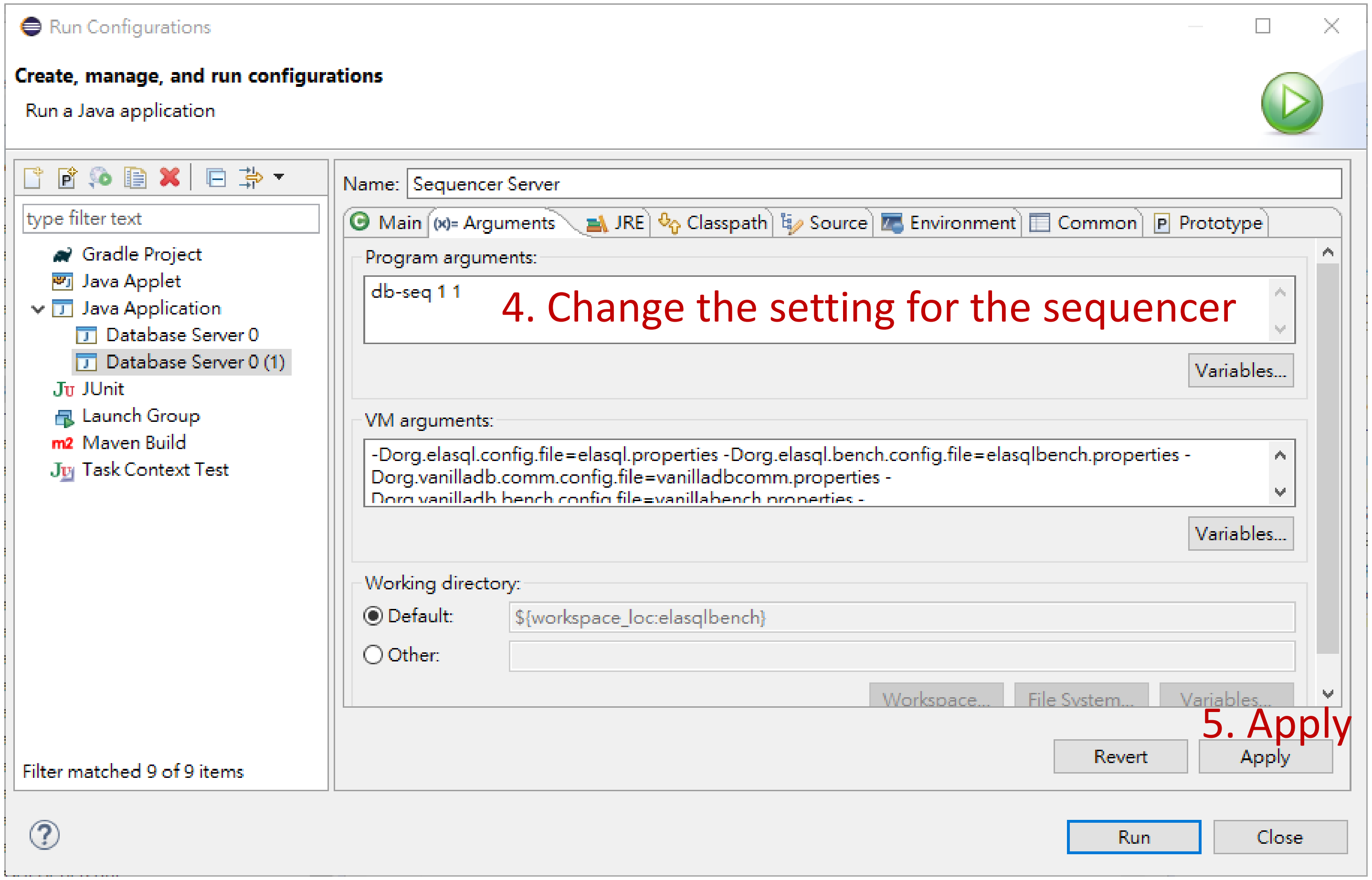
Main | Arguments | JRE | Classpath | Source | Environment | Common | Prototype

Project:
elasqlbench

Main class:
org.elasql.bench.server.StartUp

- [ ] Include system libraries when searching for a main class
- [ ] Include inherited mains when searching for a main class
- [ ] Stop in main

Setting Up Run Configurations (The Sequencer Server)



Run Configurations

Create, manage, and run configurations

Run a Java application

type filter text

- Gradle Project
- Java Applet
- Java Application
  - Database Server 0
  - Database Server 0 (1)
- JUnit
- Launch Group
- Maven Build
- Task Context Test

Filter matched 9 of 9 items

Name: Sequencer Server

Main | Arguments | JRE | Classpath | Source | Environment | Common | Prototype

Program arguments:

```
db-seq 1 1
```

4. Change the setting for the sequencer

VM arguments:

```
-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 -
```

Working directory:

Default: ${workspace_loc:elasqlbench}

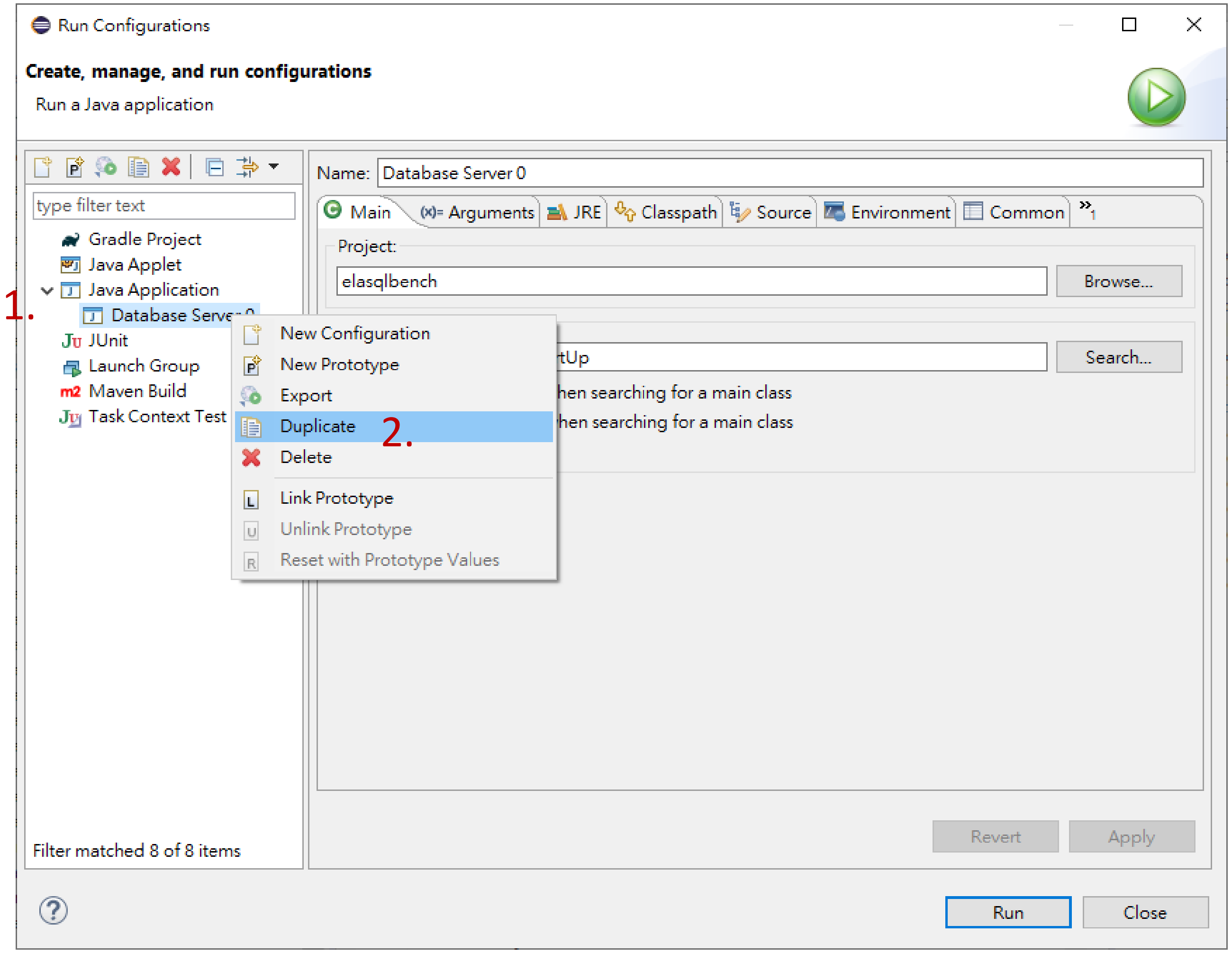
Other:

5. Apply

Revert | Apply

Run | Close

Setting Up Run Configurations (A Benchmarking Client)



Name: Database Server 0

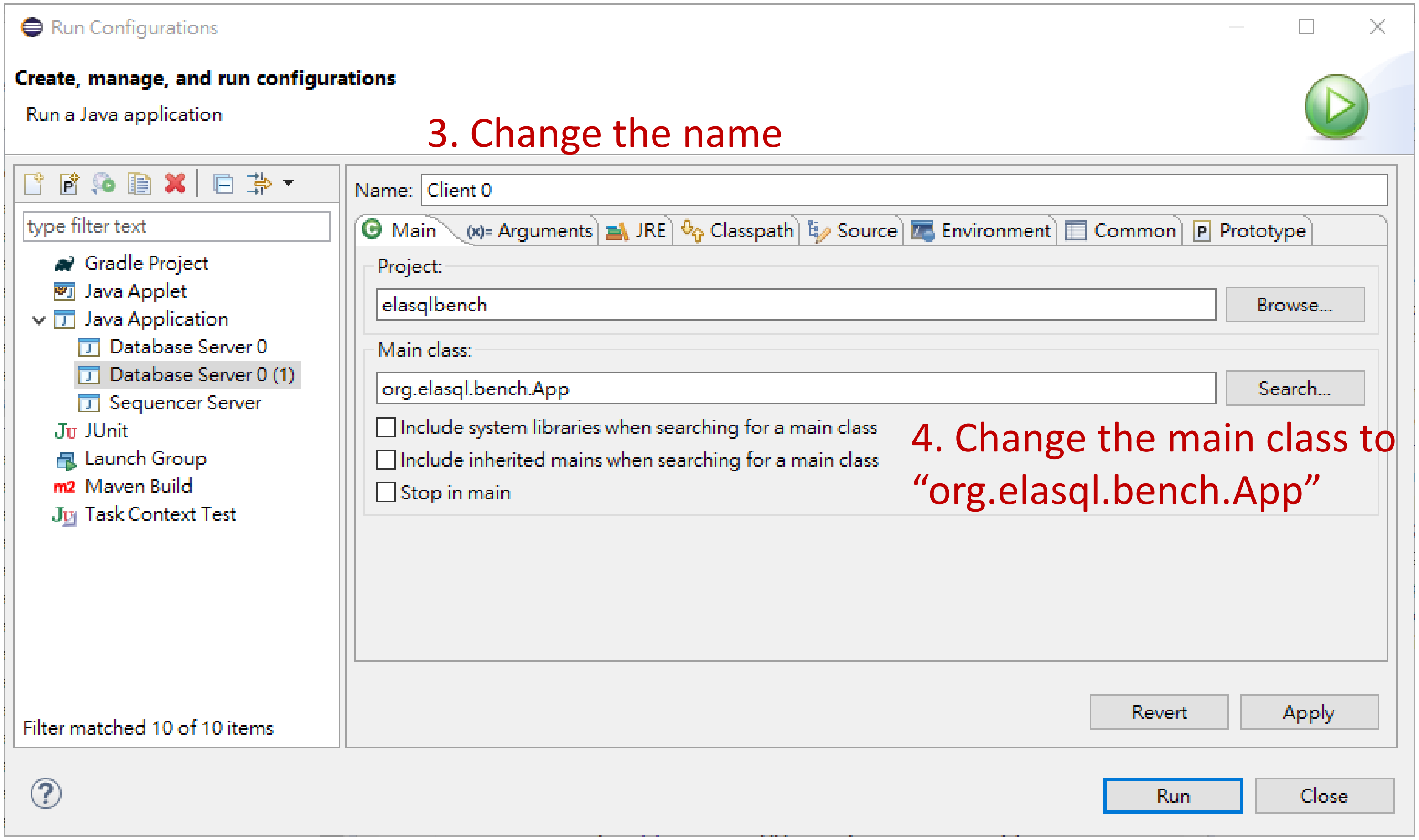
Project: elasqlbench

tUp

hen searching for a main class

hen searching for a main class

Setting Up Run Configurations (A Benchmarking Client)



Run Configurations

Create, manage, and run configurations

Run a Java application

3. Change the name

Name: Client 0

Main | Arguments | JRE | Classpath | Source | Environment | Common | Prototype

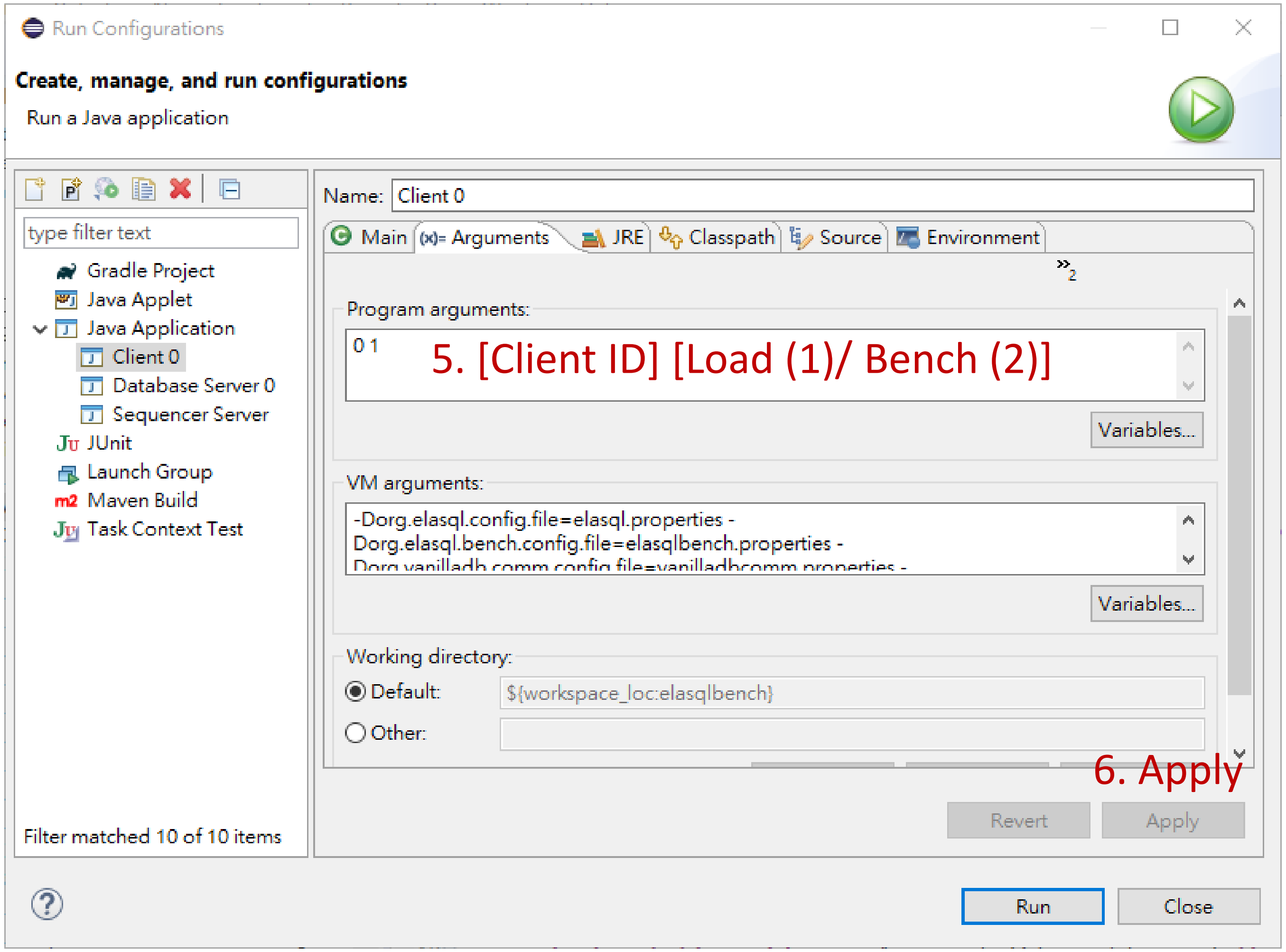
Project: elasqlbench

Main class: org.elasql.bench.App

- [ ] Include system libraries when searching for a main class
- [ ] Include inherited mains when searching for a main class
- [ ] Stop in main

4. Change the main class to "org.elasql.bench.App"

Setting Up Run Configurations (A Benchmarking Client)



Run Configurations

Create, manage, and run configurations

Run a Java application

Name: Client 0

Program arguments:

```
0 1
```

5. [Client ID] [Load (1)/ Bench (2)]

VM arguments:

```
-Dorg.elasql.config.file=elasql.properties -
Dorg.elasql.bench.config.file=elasqlbench.properties -
Dorg.vanilladb.comm.config.file=vanilladbcomm.properties -
```

Working directory:

Default: [✓] ${workspace_loc:elasqlbench}

Other:

6. Apply

Program & VM Arguments (For Clients)

- Program Arguments
  - Client ID: the ID of the client process
  - Load/Bench: to controls the action of this client
    - 1: Loading a new testbed on a clean database.
    - 2: Benchmarking on an existing testbed.

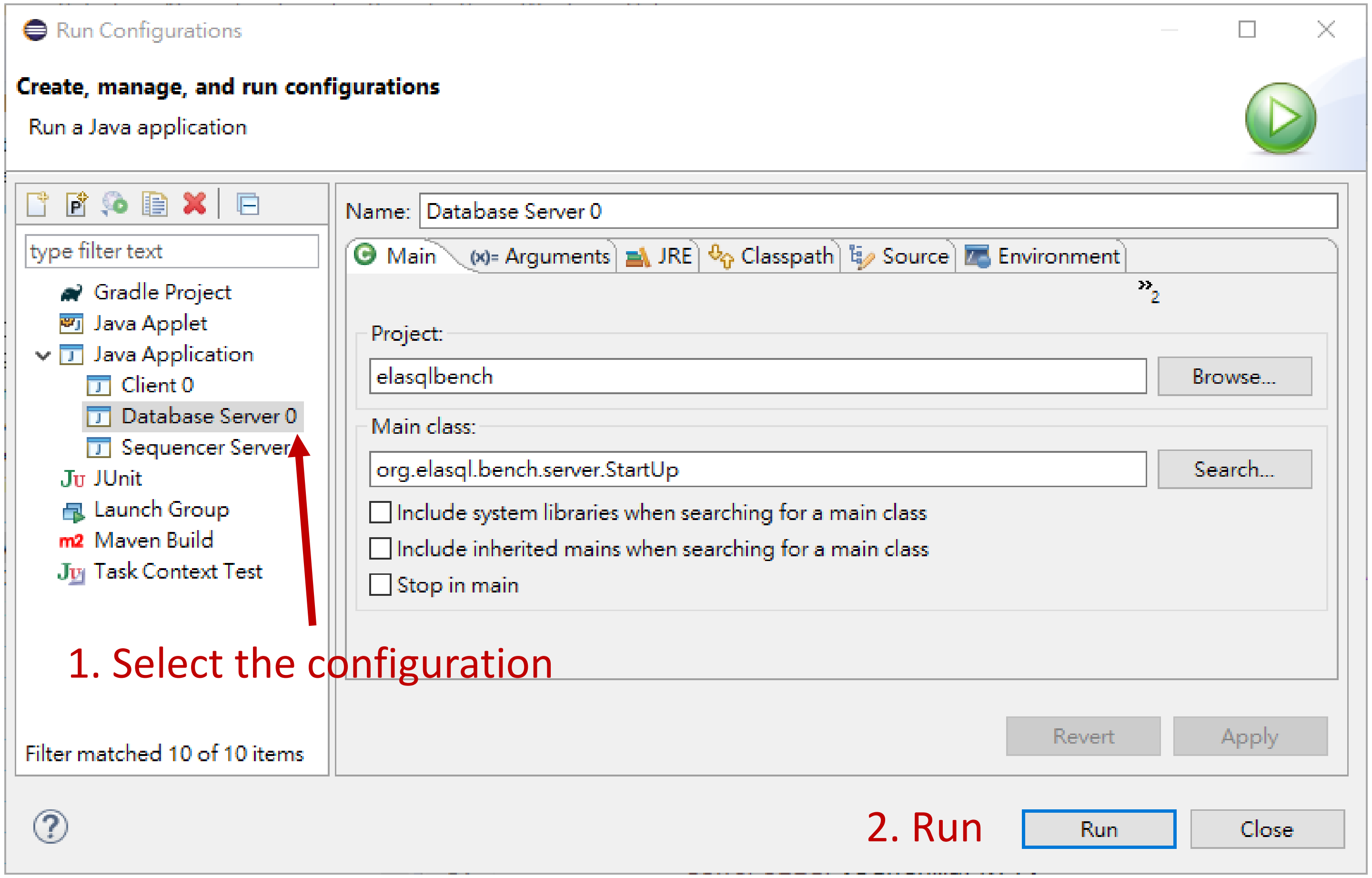
- Note that a client must first load a new testbed on a system before benchmarking it.

- VM Arguments: same as the servers

Testing inside Eclipse

1. Setup the properties files
   - Which includes the configurations for ElaSQL and ElaSQL-Bench
2. Setup run configurations
3. Loading a testbed
   1. Launch servers
   2. Launch clients
4. Benchmarking
   1. Launch servers
   2. Launch clients

Launching A Database Server



Run Configurations

Create, manage, and run configurations

Run a Java application

Name: Database Server 0

Main | Arguments | JRE | Classpath | Source | Environment

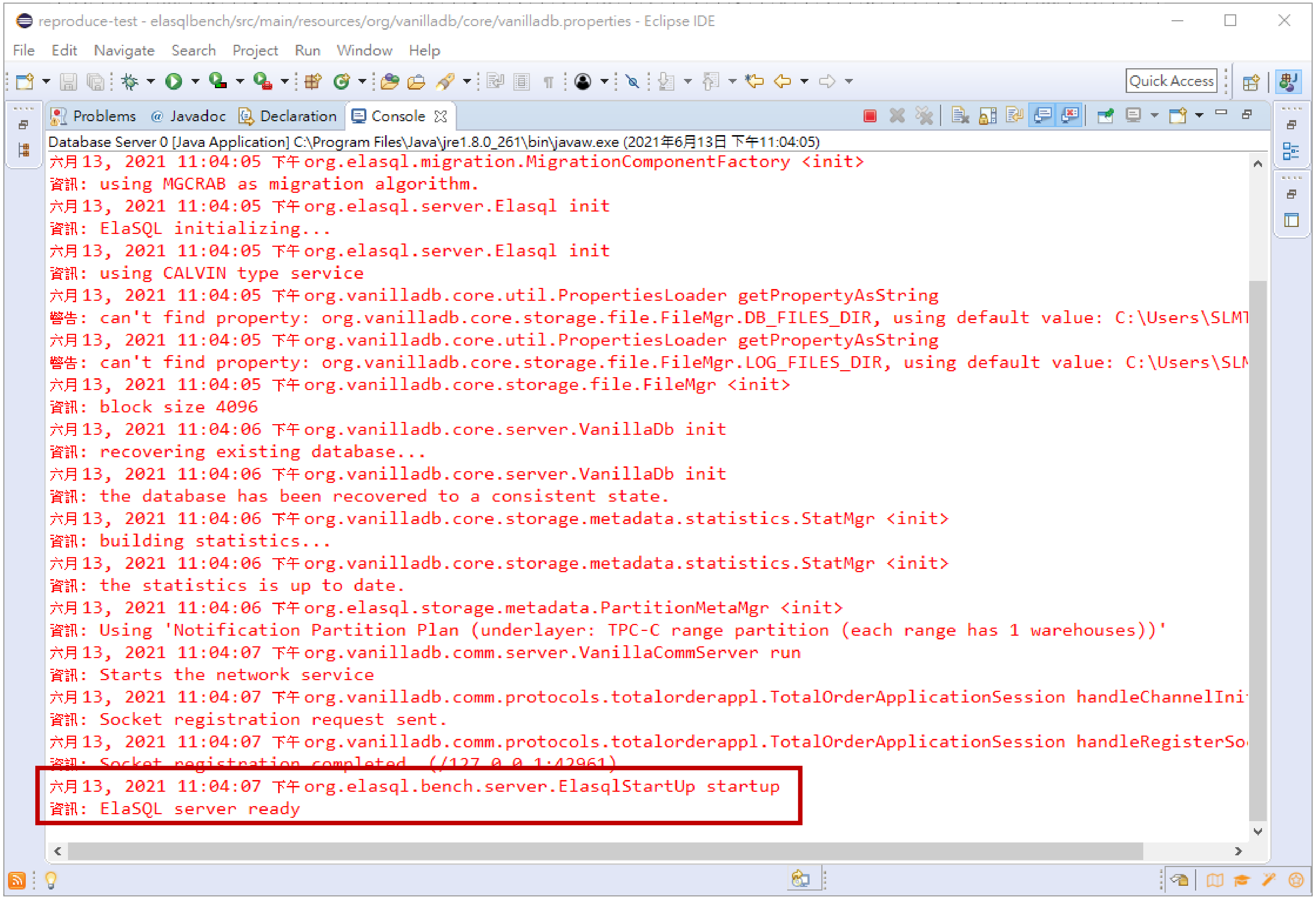
Project: elasqlbench

Main class: org.elasql.bench.server.StartUp

- [ ] Include system libraries when searching for a main class
- [ ] Include inherited mains when searching for a main class
- [ ] Stop in main

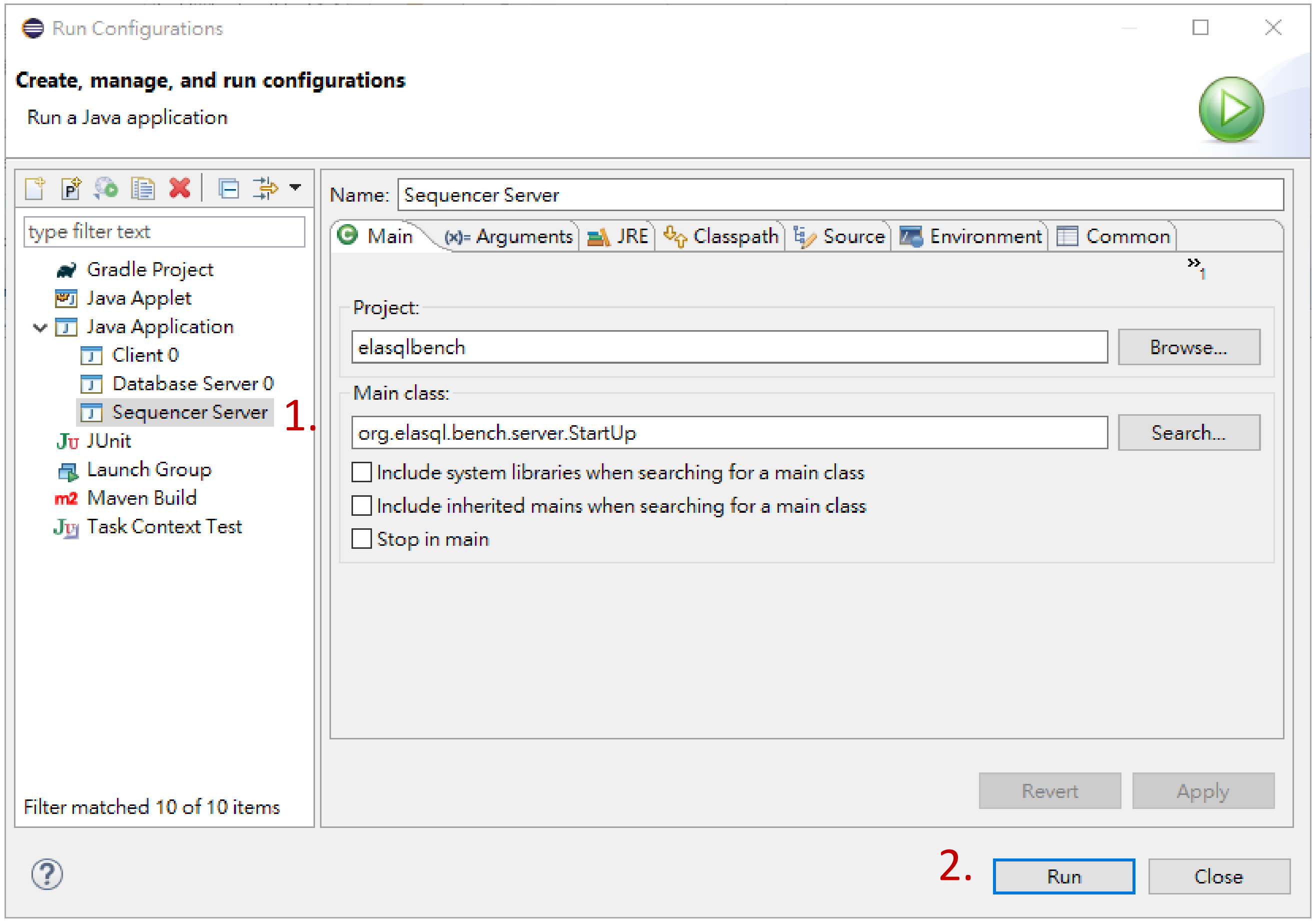
1. Select the configuration

2. Run


reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Quick Access
Problems
@ Javadoc
Declaration
Console

```
Database Server 0 [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (2021年6月13日 下午11:04:05)
六月 13, 2021 11:04:05 下午 org.elasql.migration.MigrationComponentFactory <init>
資訊: using MGCRAB as migration algorithm.
六月 13, 2021 11:04:05 下午 org.elasql.server.Elasql init
資訊: ElaSQL initializing...
六月 13, 2021 11:04:05 下午 org.elasql.server.Elasql init
資訊: using CALVIN type service
六月 13, 2021 11:04:05 下午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.core.storage.file.FileMgr.DB_FILES_DIR, using default value: C:\Users\SLMT
六月 13, 2021 11:04:05 下午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.core.storage.file.FileMgr.LOG_FILES_DIR, using default value: C:\Users\SLM
六月 13, 2021 11:04:05 下午 org.vanilladb.core.storage.file.FileMgr <init>
資訊: block size 4096
六月 13, 2021 11:04:06 下午 org.vanilladb.core.server.VanillaDb init
資訊: recovering existing database...
六月 13, 2021 11:04:06 下午 org.vanilladb.core.server.VanillaDb init
資訊: the database has been recovered to a consistent state.
六月 13, 2021 11:04:06 下午 org.vanilladb.core.storage.metadata.statistics.StatMgr <init>
資訊: building statistics...
六月 13, 2021 11:04:06 下午 org.vanilladb.core.storage.metadata.statistics.StatMgr <init>
資訊: the statistics is up to date.
六月 13, 2021 11:04:06 下午 org.elasql.storage.metadata.PartitionMetaMgr <init>
資訊: Using 'Notification Partition Plan (underlayer: TPC-C range partition (each range has 1 warehouses))'
六月 13, 2021 11:04:07 下午 org.vanilladb.comm.server.VanillaCommServer run
資訊: Starts the network service
六月 13, 2021 11:04:07 下午 org.vanilladb.comm.protocols.totalorderappl.TotalOrderApplicationSession handleChannelIni
資訊: Socket registration request sent.
六月 13, 2021 11:04:07 下午 org.vanilladb.comm.protocols.totalorderappl.TotalOrderApplicationSession handleRegisterSo
資訊: Socket registration completed. (/127.0.0.1:42961)
六月 13, 2021 11:04:07 下午 org.elasql.bench.server.ElasqlStartUp startup
資訊: ElaSQL server ready
```

Launching The Sequencer


Run Configurations

Create, manage, and run configurations

Run a Java application

type filter text

- Gradle Project
- Java Applet
- Java Application
  - Client 0
  - Database Server 0
  - Sequencer Server 1.
- JUnit
- Launch Group
- Maven Build
- Task Context Test

Filter matched 10 of 10 items

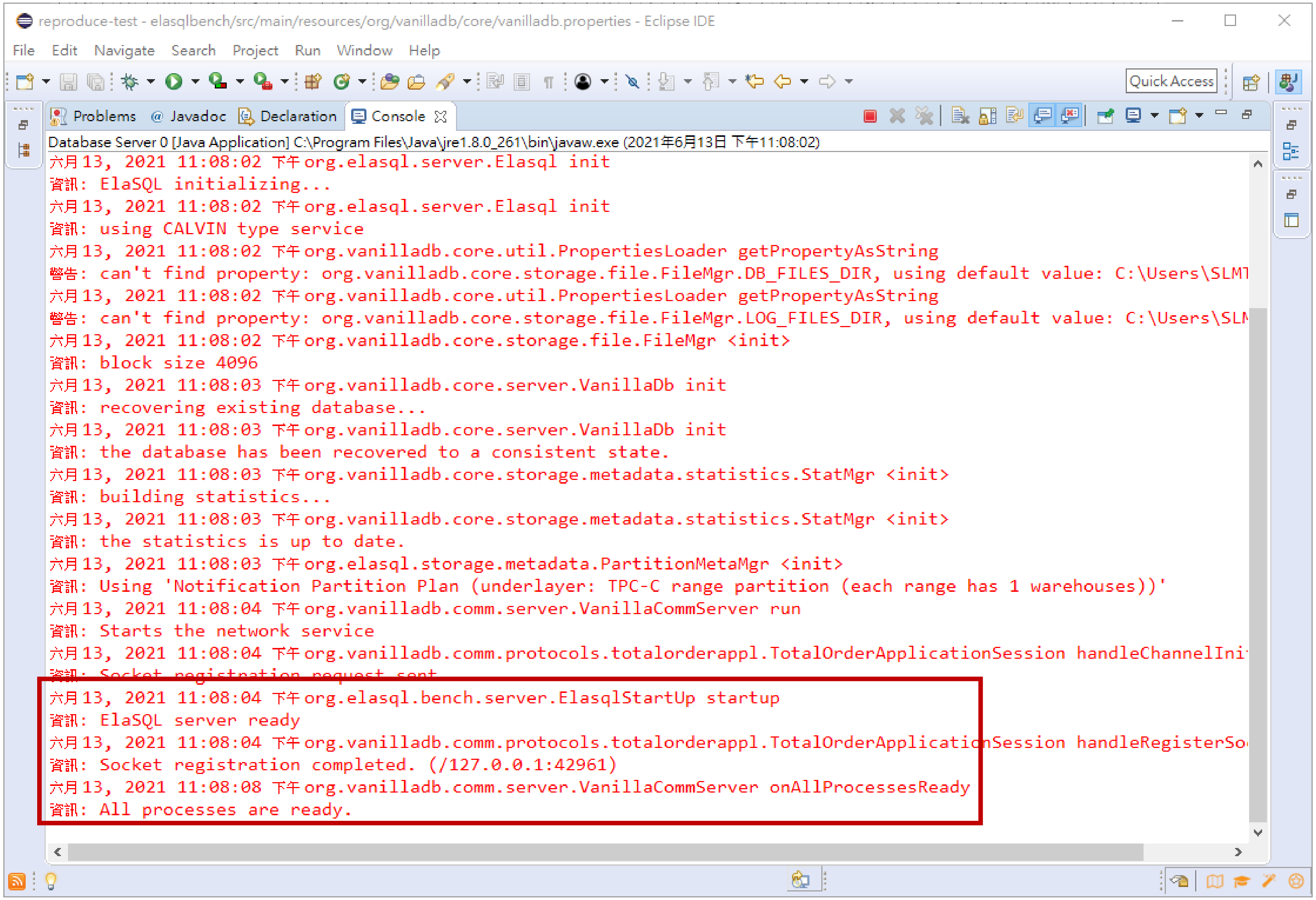
Name: Sequencer Server

Main | Arguments | JRE | Classpath | Source | Environment | Common

Project: elasqlbench

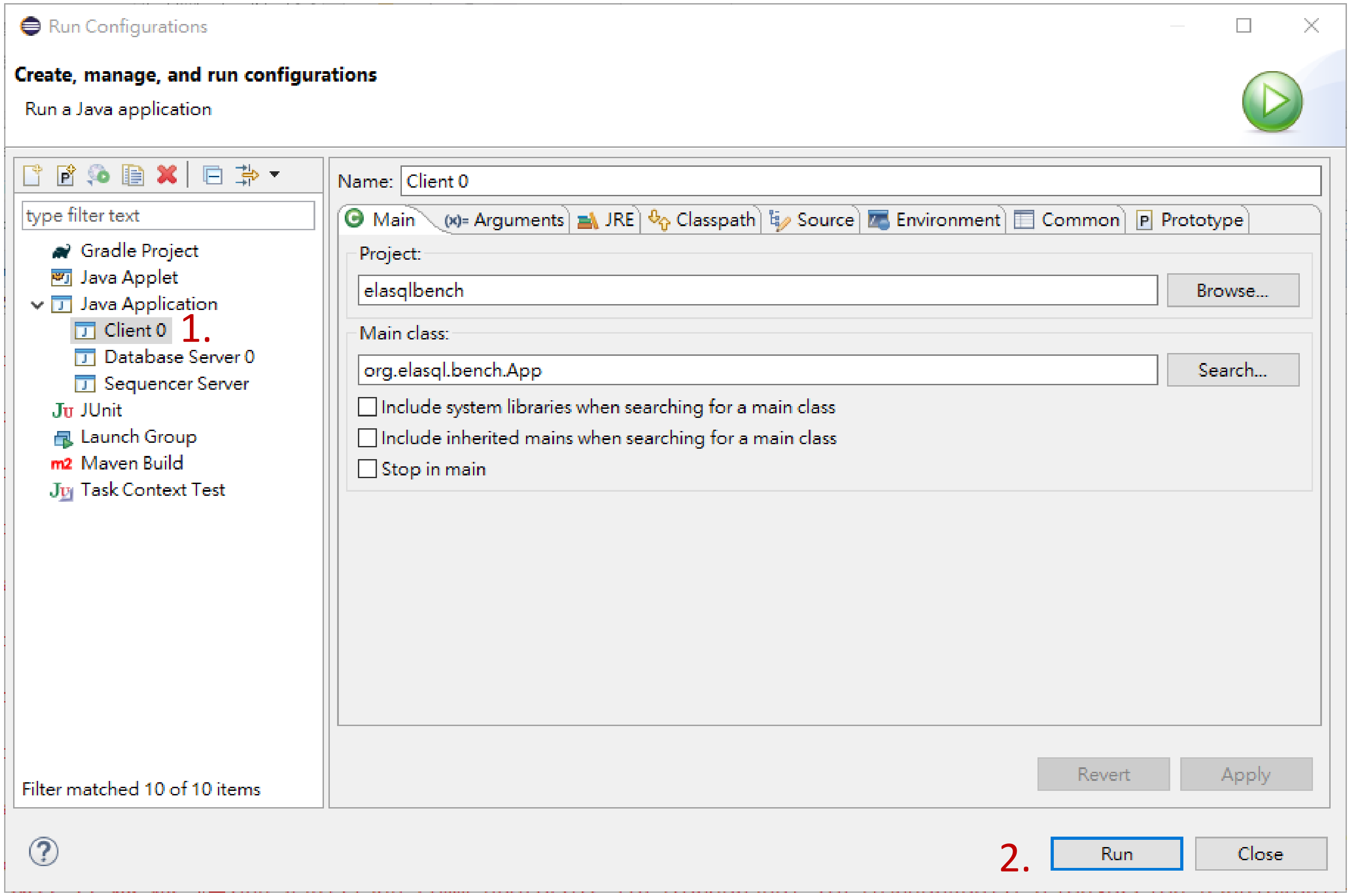
Main class: org.elasql.bench.server.StartUp

- [ ] Include system libraries when searching for a main class
- [ ] Include inherited mains when searching for a main class
- [ ] Stop in main


reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Quick Access
Problems
Javadoc
Declaration
Console

```
Database Server 0 [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (2021年6月13日 下午11:08:02)
六月 13, 2021 11:08:02 下午 org.elasql.server.Elasql init
資訊: ElaSQL initializing...
六月 13, 2021 11:08:02 下午 org.elasql.server.Elasql init
資訊: using CALVIN type service
六月 13, 2021 11:08:02 下午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.core.storage.file.FileMgr.DB_FILES_DIR, using default value: C:\Users\SLMT
六月 13, 2021 11:08:02 下午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.core.storage.file.FileMgr.LOG_FILES_DIR, using default value: C:\Users\SLM
六月 13, 2021 11:08:02 下午 org.vanilladb.core.storage.file.FileMgr <init>
資訊: block size 4096
六月 13, 2021 11:08:03 下午 org.vanilladb.core.server.VanillaDb init
資訊: recovering existing database...
六月 13, 2021 11:08:03 下午 org.vanilladb.core.server.VanillaDb init
資訊: the database has been recovered to a consistent state.
六月 13, 2021 11:08:03 下午 org.vanilladb.core.storage.metadata.statistics.StatMgr <init>
資訊: building statistics...
六月 13, 2021 11:08:03 下午 org.vanilladb.core.storage.metadata.statistics.StatMgr <init>
資訊: the statistics is up to date.
六月 13, 2021 11:08:03 下午 org.elasql.storage.metadata.PartitionMetaMgr <init>
資訊: Using 'Notification Partition Plan (underlayer: TPC-C range partition (each range has 1 warehouses))'
六月 13, 2021 11:08:04 下午 org.vanilladb.comm.server.VanillaCommServer run
資訊: Starts the network service
六月 13, 2021 11:08:04 下午 org.vanilladb.comm.protocols.totalorderappl.TotalOrderApplicationSession handleChannelInit
資訊: Socket registration request sent
六月 13, 2021 11:08:04 下午 org.elasql.bench.server.ElasqlStartUp startup
資訊: ElaSQL server ready
六月 13, 2021 11:08:04 下午 org.vanilladb.comm.protocols.totalorderappl.TotalOrderApplicationSession handleRegisterSo
資訊: Socket registration completed. (/127.0.0.1:42961)
六月 13, 2021 11:08:08 下午 org.vanilladb.comm.server.VanillaCommServer onAllProcessesReady
資訊: All processes are ready.
```

Launching a Client



Run Configurations

Create, manage, and run configurations

Run a Java application

type filter text

- Gradle Project
- Java Applet
- Java Application
  - Client 0
  - Database Server 0
  - Sequencer Server
- JUnit
- Launch Group
- Maven Build
- Task Context Test

Filter matched 10 of 10 items

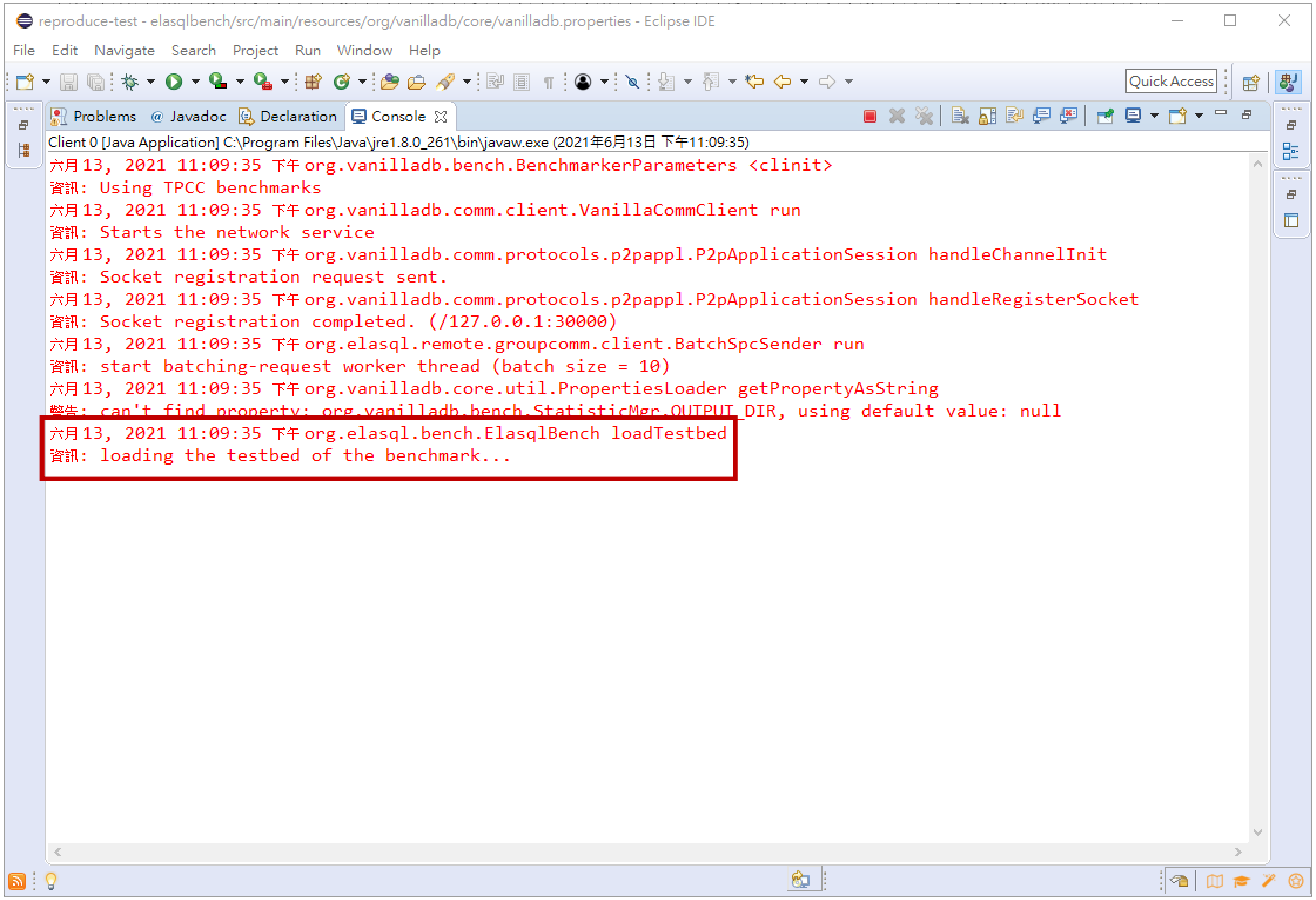
Name: Client 0

Main | Arguments | JRE | Classpath | Source | Environment | Common | Prototype

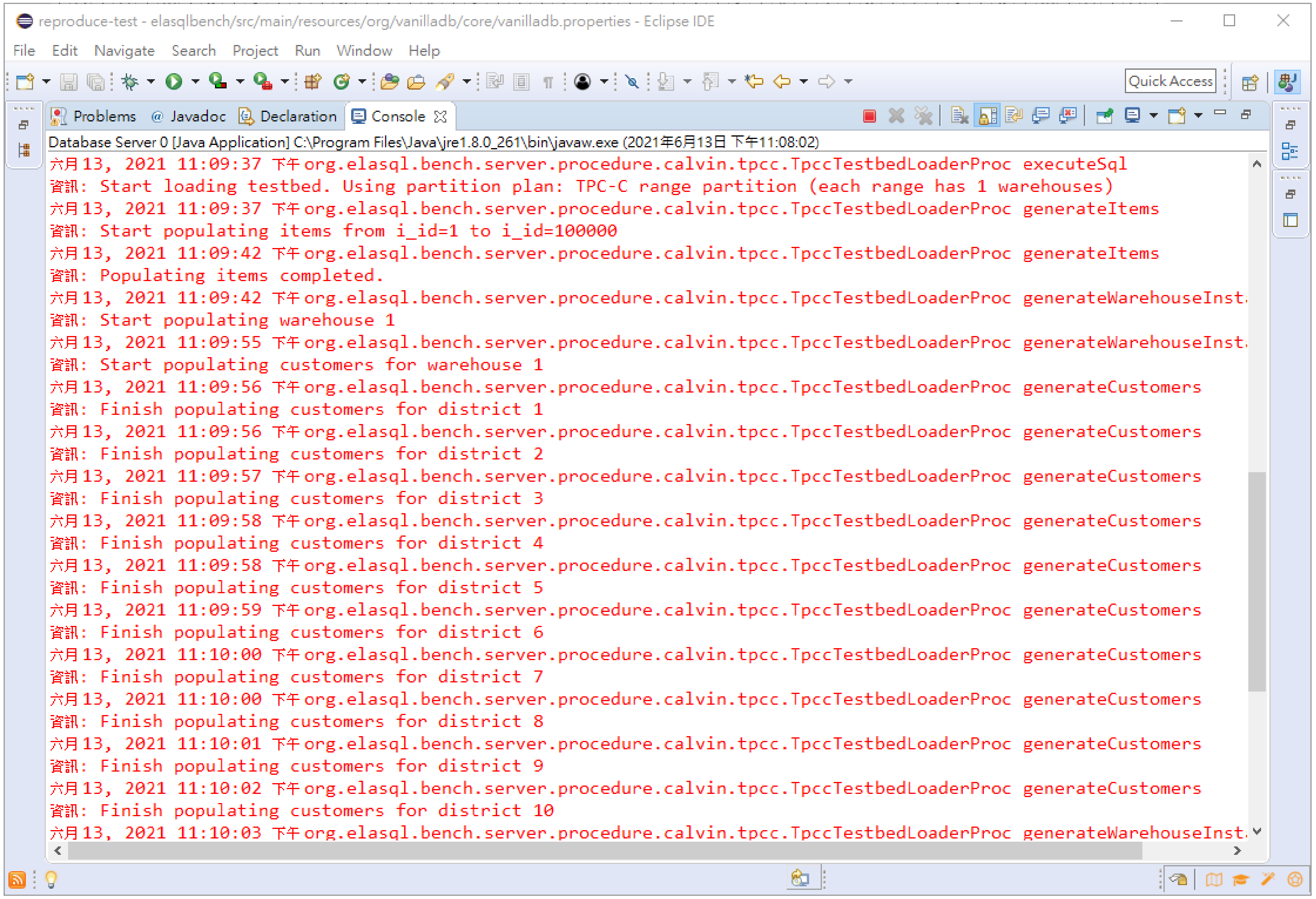
Project: elasqlbench

Main class: org.elasql.bench.App

- [ ] Include system libraries when searching for a main class
- [ ] Include inherited mains when searching for a main class
- [ ] Stop in main

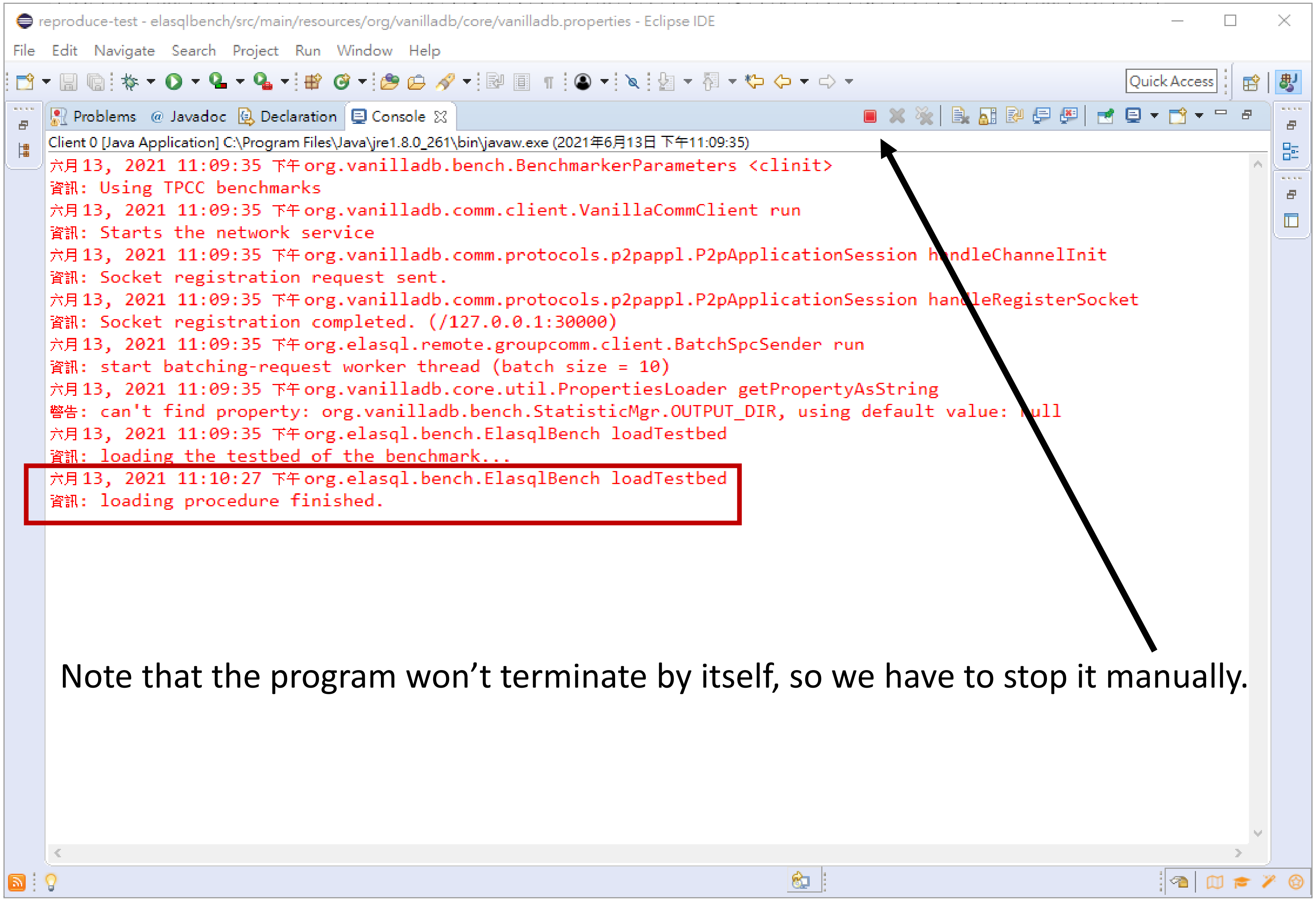

reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Quick Access
Problems @ Javadoc Declaration Console

```
Client 0 [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (2021年6月13日 下午11:09:35)
六月13, 2021 11:09:35 下午 org.vanilladb.bench.BenchmarkerParameters <clinit>
資訊: Using TPCC benchmarks
六月13, 2021 11:09:35 下午 org.vanilladb.comm.client.VanillaCommClient run
資訊: Starts the network service
六月13, 2021 11:09:35 下午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSession handleChannelInit
資訊: Socket registration request sent.
六月13, 2021 11:09:35 下午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSession handleRegisterSocket
資訊: Socket registration completed. (/127.0.0.1:30000)
六月13, 2021 11:09:35 下午 org.elasql.remote.groupcomm.client.BatchSpcSender run
資訊: start batching-request worker thread (batch size = 10)
六月13, 2021 11:09:35 下午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.bench.StatisticMgr.OUTPUT_DIR, using default value: null
六月13, 2021 11:09:35 下午 org.elasql.bench.ElasqlBench loadTestbed
資訊: loading the testbed of the benchmark...
```


reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Quick Access
Problems @ Javadoc Declaration Console

```
Database Server 0 [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (2021年6月13日 下午11:08:02)
六月 13, 2021 11:09:37 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc executeSql
資訊: Start loading testbed. Using partition plan: TPC-C range partition (each range has 1 warehouses)
六月 13, 2021 11:09:37 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateItems
資訊: Start populating items from i_id=1 to i_id=100000
六月 13, 2021 11:09:42 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateItems
資訊: Populating items completed.
六月 13, 2021 11:09:42 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateWarehouseInst
資訊: Start populating warehouse 1
六月 13, 2021 11:09:55 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateWarehouseInst
資訊: Start populating customers for warehouse 1
六月 13, 2021 11:09:56 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 1
六月 13, 2021 11:09:56 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 2
六月 13, 2021 11:09:57 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 3
六月 13, 2021 11:09:58 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 4
六月 13, 2021 11:09:58 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 5
六月 13, 2021 11:09:59 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 6
六月 13, 2021 11:10:00 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 7
六月 13, 2021 11:10:00 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 8
六月 13, 2021 11:10:01 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 9
六月 13, 2021 11:10:02 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateCustomers
資訊: Finish populating customers for district 10
六月 13, 2021 11:10:03 下午 org.elasql.bench.server.procedure.calvin.tpcc.TpccTestbedLoaderProc generateWarehouseInst
```

The database server will show some messages about the loaded data.


reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Quick Access
Problems
@ Javadoc
Declaration
Console

```
Client 0 [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (2021年6月13日 下午11:09:35)
六月 13, 2021 11:09:35 下午 org.vanilladb.bench.BenchmarkerParameters <clinit>
資訊: Using TPCC benchmarks
六月 13, 2021 11:09:35 下午 org.vanilladb.comm.client.VanillaCommClient run
資訊: Starts the network service
六月 13, 2021 11:09:35 下午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSession handleChannelInit
資訊: Socket registration request sent.
六月 13, 2021 11:09:35 下午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSession handleRegisterSocket
資訊: Socket registration completed. (/127.0.0.1:30000)
六月 13, 2021 11:09:35 下午 org.elasql.remote.groupcomm.client.BatchSpcSender run
資訊: start batching-request worker thread (batch size = 10)
六月 13, 2021 11:09:35 下午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.bench.StatisticMgr.OUTPUT_DIR, using default value: null
六月 13, 2021 11:09:35 下午 org.elasql.bench.ElasqlBench loadTestbed
資訊: loading the testbed of the benchmark...
六月 13, 2021 11:10:27 下午 org.elasql.bench.ElasqlBench loadTestbed
資訊: loading procedure finished.
```

Note that the program won't terminate by itself, so we have to stop it manually.

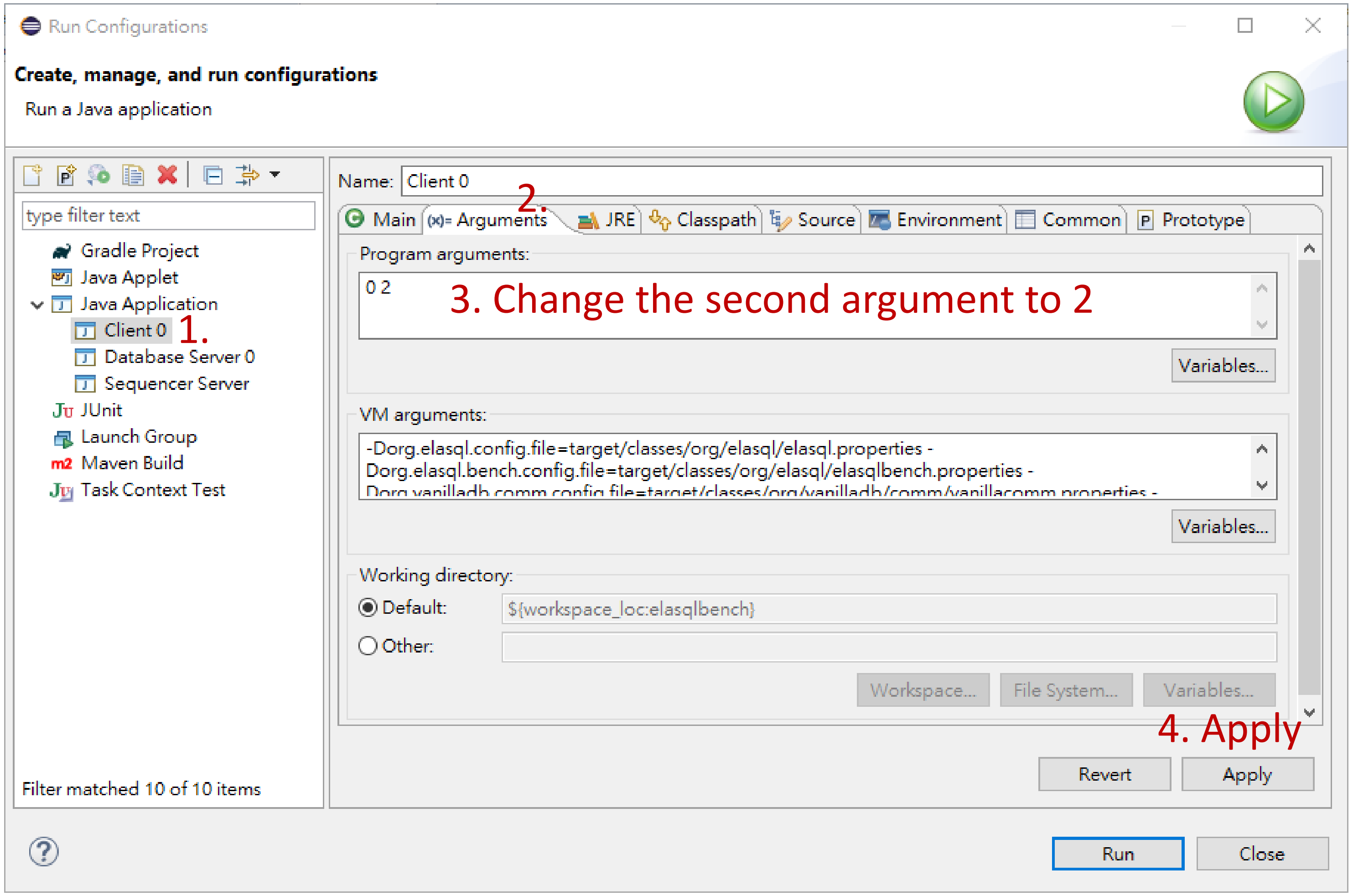
A Note

- Since the database system may change the state of database files after each benchmarking test, in order to ensure the consistency of the benchmarking result, we suggest to
  1. Terminate all the processes immediately after the loading procedure
  2. Backup the database directory (usually in your home directory)
  3. Replace the database directory with the backup before each benchmarking run.

Testing inside Eclipse

1. Setup the properties files
   - Which includes the configurations for ElaSQL and ElaSQL-Bench
2. Setup run configurations
3. Loading a testbed
   1. Launch servers
   2. Launch clients
4. Benchmarking
   1. Launch servers
   2. Launch clients

Changing the Client to Benchmarking Mode


Run Configurations

Create, manage, and run configurations

Run a Java application

Name: Client 0

Program arguments:

```
0 2
```

3. Change the second argument to 2

VM arguments:

```
-Dorg.elasql.config.file=target/classes/org/elasql/elasql.properties -
Dorg.elasql.bench.config.file=target/classes/org/elasql/elasqlbench.properties -
Dorg.vanilladb.comm.config.file=target/classes/org/vanilladb/comm/vanillacomm.properties -
```

Working directory:

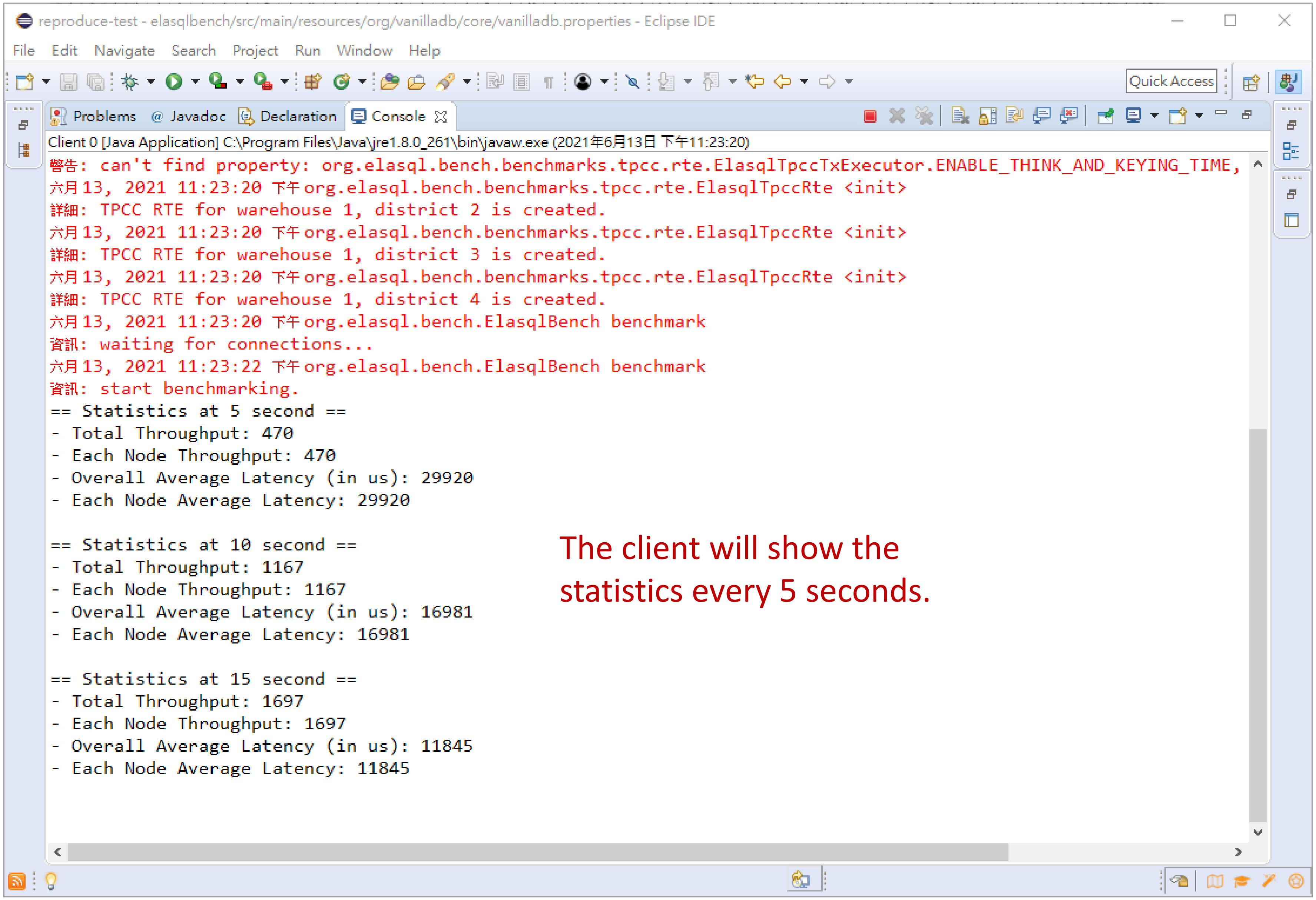
Default [✓]: ${workspace_loc:elasqlbench}

Other:

4. Apply

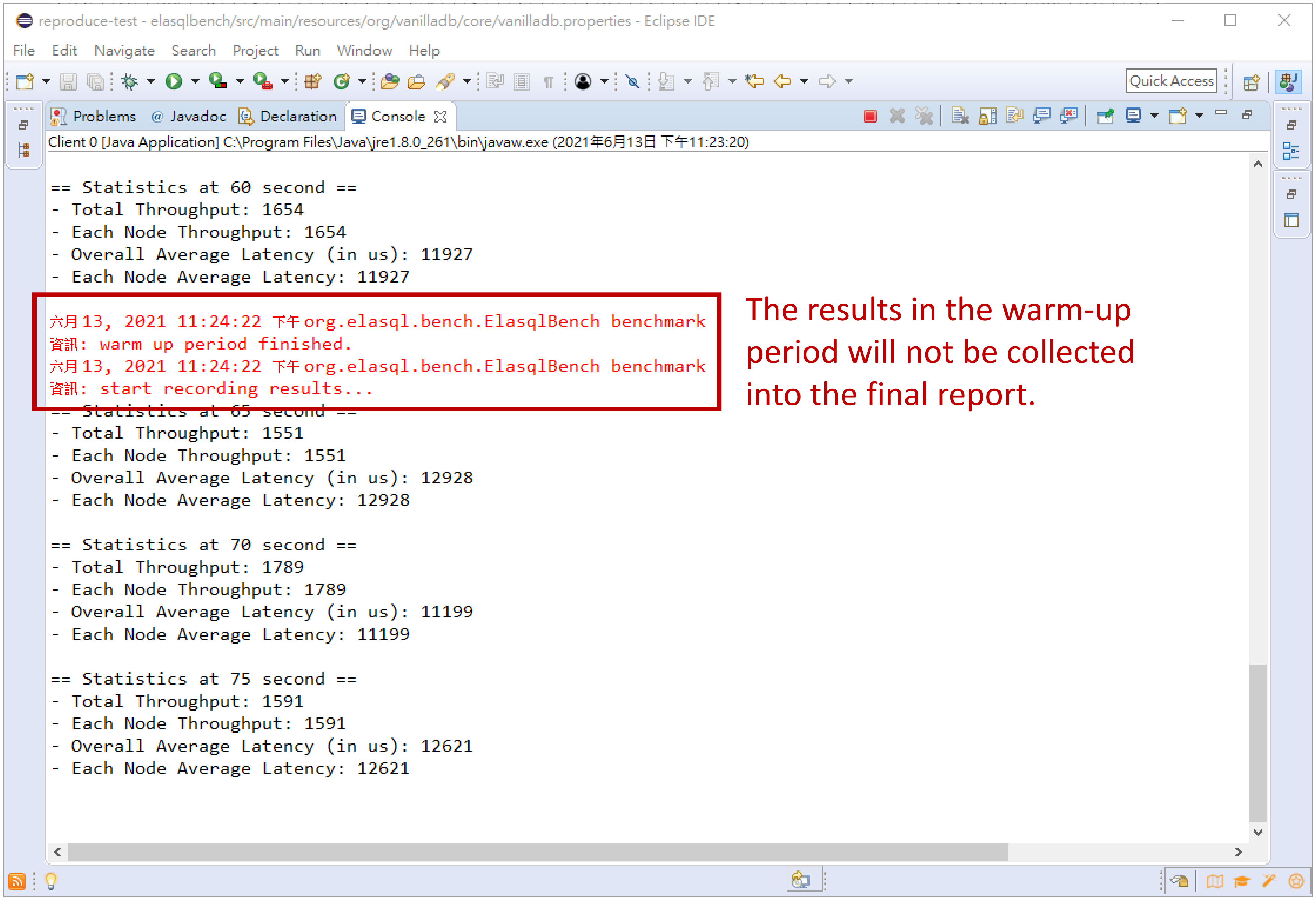
Launching the Servers and the Client

- Just follow the same launch procedure as loading a testbed.
  1. Launch the database server
  2. Launch the sequencer server
  3. Wait for the server ready
  4. Launch the client


reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Quick Access
Problems
Javadoc
Declaration
Console

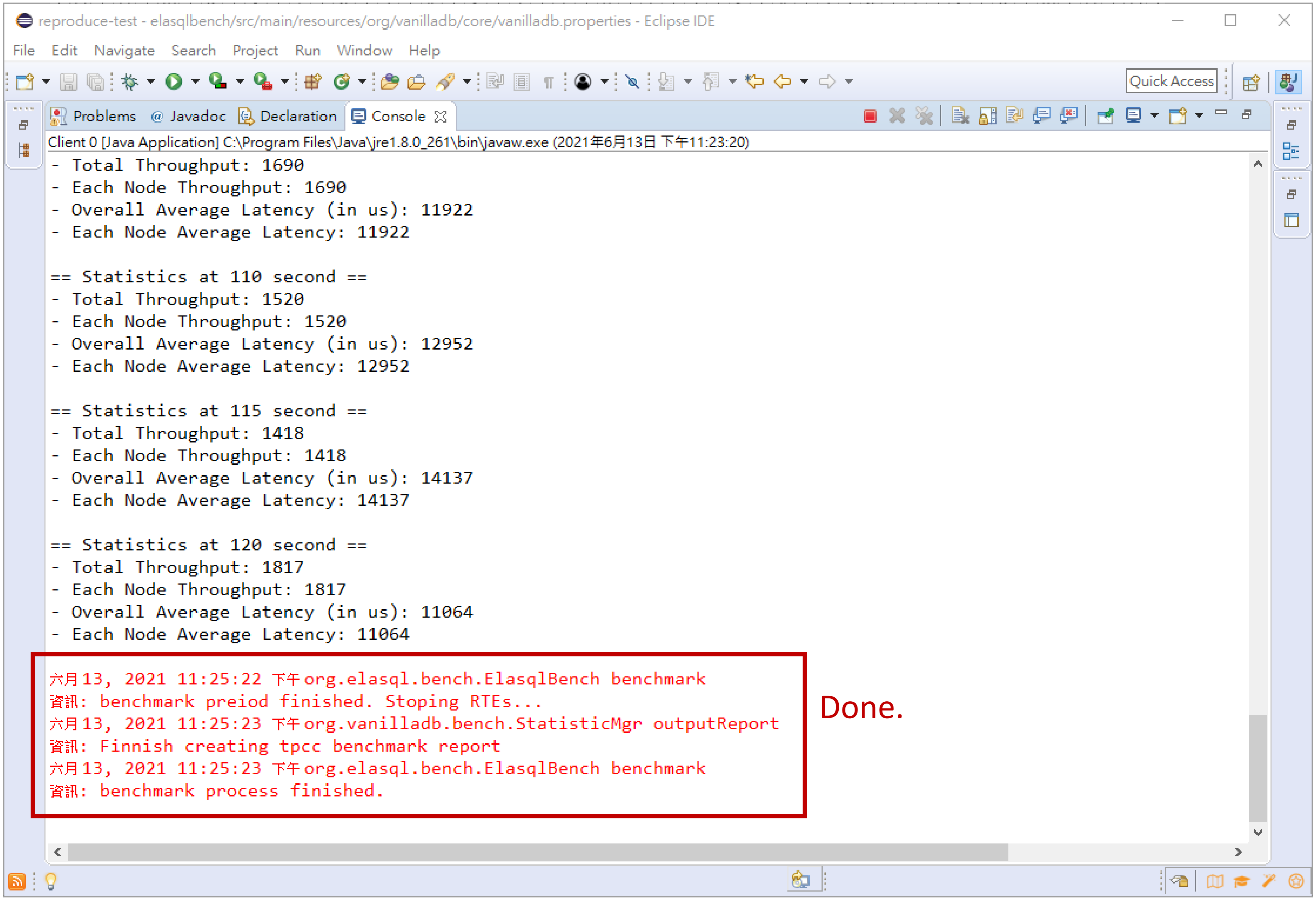
```
Client 0 [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (2021年6月13日 下午11:23:20)
警告: can't find property: org.elasql.bench.benchmarks.tpcc.rte.ElasqlTpccTxExecutor.ENABLE_THINK_AND_KEYING_TIME,
六月 13, 2021 11:23:20 下午 org.elasql.bench.benchmarks.tpcc.rte.ElasqlTpccRte <init>
詳細: TPCC RTE for warehouse 1, district 2 is created.
六月 13, 2021 11:23:20 下午 org.elasql.bench.benchmarks.tpcc.rte.ElasqlTpccRte <init>
詳細: TPCC RTE for warehouse 1, district 3 is created.
六月 13, 2021 11:23:20 下午 org.elasql.bench.benchmarks.tpcc.rte.ElasqlTpccRte <init>
詳細: TPCC RTE for warehouse 1, district 4 is created.
六月 13, 2021 11:23:20 下午 org.elasql.bench.ElasqlBench benchmark
資訊: waiting for connections...
六月 13, 2021 11:23:22 下午 org.elasql.bench.ElasqlBench benchmark
資訊: start benchmarking.
== Statistics at 5 second ==
- Total Throughput: 470
- Each Node Throughput: 470
- Overall Average Latency (in us): 29920
- Each Node Average Latency: 29920

== Statistics at 10 second ==
- Total Throughput: 1167
- Each Node Throughput: 1167
- Overall Average Latency (in us): 16981
- Each Node Average Latency: 16981

== Statistics at 15 second ==
- Total Throughput: 1697
- Each Node Throughput: 1697
- Overall Average Latency (in us): 11845
- Each Node Average Latency: 11845
```

The client will show the statistics every 5 seconds.


reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Quick Access
Problems @ Javadoc Declaration Console

```
Client 0 [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (2021年6月13日 下午11:23:20)

== Statistics at 60 second ==
- Total Throughput: 1654
- Each Node Throughput: 1654
- Overall Average Latency (in us): 11927
- Each Node Average Latency: 11927

六月 13, 2021 11:24:22 下午 org.elasql.bench.ElasqlBench benchmark
資訊: warm up period finished.
六月 13, 2021 11:24:22 下午 org.elasql.bench.ElasqlBench benchmark
資訊: start recording results...
== Statistics at 65 second ==
- Total Throughput: 1551
- Each Node Throughput: 1551
- Overall Average Latency (in us): 12928
- Each Node Average Latency: 12928

== Statistics at 70 second ==
- Total Throughput: 1789
- Each Node Throughput: 1789
- Overall Average Latency (in us): 11199
- Each Node Average Latency: 11199

== Statistics at 75 second ==
- Total Throughput: 1591
- Each Node Throughput: 1591
- Overall Average Latency (in us): 12621
- Each Node Average Latency: 12621
```

The results in the warm-up period will not be collected into the final report.


reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Quick Access
Problems @ Javadoc Declaration Console

```
Client 0 [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (2021年6月13日 下午11:23:20)
- Total Throughput: 1690
- Each Node Throughput: 1690
- Overall Average Latency (in us): 11922
- Each Node Average Latency: 11922

== Statistics at 110 second ==
- Total Throughput: 1520
- Each Node Throughput: 1520
- Overall Average Latency (in us): 12952
- Each Node Average Latency: 12952

== Statistics at 115 second ==
- Total Throughput: 1418
- Each Node Throughput: 1418
- Overall Average Latency (in us): 14137
- Each Node Average Latency: 14137

== Statistics at 120 second ==
- Total Throughput: 1817
- Each Node Throughput: 1817
- Overall Average Latency (in us): 11064
- Each Node Average Latency: 11064

六月 13, 2021 11:25:22 下午 org.elasql.bench.ElasqlBench benchmark
資訊: benchmark preiod finished. Stoping RTEs...
六月 13, 2021 11:25:23 下午 org.vanilladb.bench.StatisticMgr outputReport
資訊: Finnish creating tpcc benchmark report
六月 13, 2021 11:25:23 下午 org.elasql.bench.ElasqlBench benchmark
資訊: benchmark process finished.
```

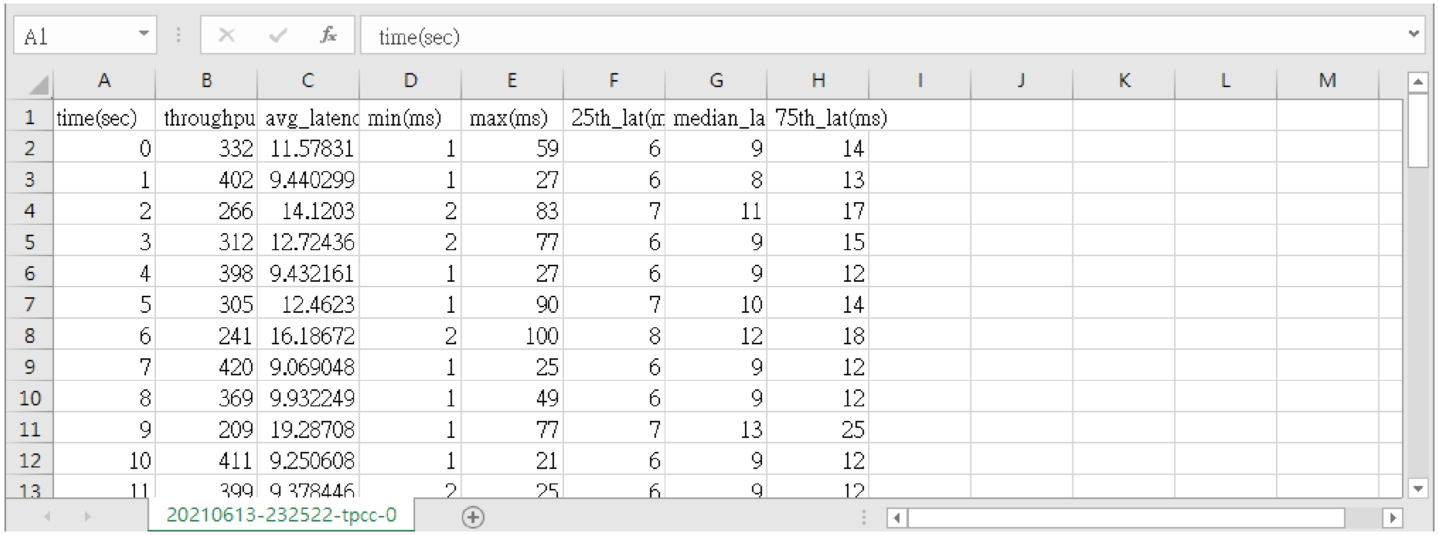
Done.

Reports

- The report will be put in "$HOME$/benchmark_results" by default.
  - You can change this in vanillabench.properties

- There are two report will be generated:
  - [Datetime]-[Benchmark Name]-[Client ID].csv
    - Record the timeline of system performance
  - [Datetime]-[Benchmark Name]-[Client ID].txt
    - Summary the result for each transaction type.

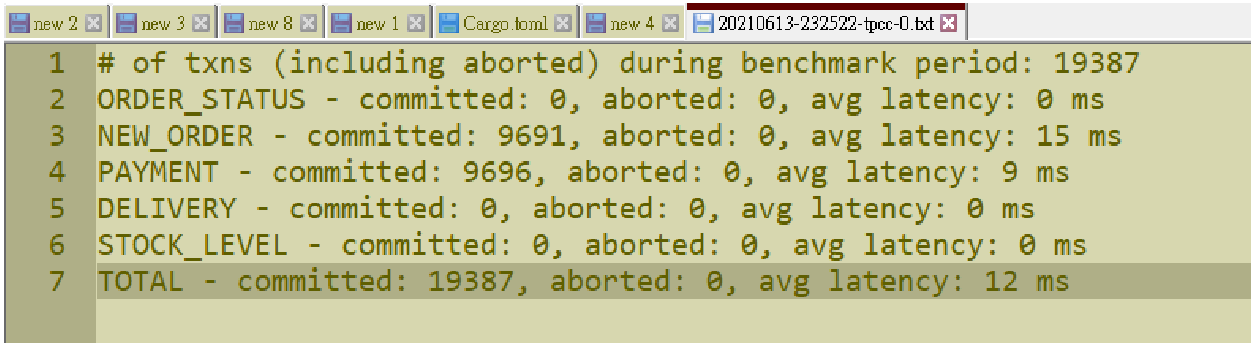
Examples of Reports



A1 time(sec)

	A	B	C	D	E	F	G	H
1	time(sec)	throughpu	avg_latenc	min(ms)	max(ms)	25th_lat(m	median_la	75th_lat(ms)
2	0	332	11.57831	1	59	6	9	14
3	1	402	9.440299	1	27	6	8	13
4	2	266	14.1203	2	83	7	11	17
5	3	312	12.72436	2	77	6	9	15
6	4	398	9.432161	1	27	6	9	12
7	5	305	12.4623	1	90	7	10	14
8	6	241	16.18672	2	100	8	12	18
9	7	420	9.069048	1	25	6	9	12
10	8	369	9.932249	1	49	6	9	12
11	9	209	19.28708	1	77	7	13	25
12	10	411	9.250608	1	21	6	9	12
13	11	399	9.378446	2	25	6	9	12

20210613-232522-tpcc-0



```
# of txns (including aborted) during benchmark period: 19387
ORDER_STATUS - committed: 0, aborted: 0, avg latency: 0 ms
NEW_ORDER - committed: 9691, aborted: 0, avg latency: 15 ms
PAYMENT - committed: 9696, aborted: 0, avg latency: 9 ms
DELIVERY - committed: 0, aborted: 0, avg latency: 0 ms
STOCK_LEVEL - committed: 0, aborted: 0, avg latency: 0 ms
TOTAL - committed: 19387, aborted: 0, avg latency: 12 ms
```

Outline

- Introduction to ElaSQL project
- How to test/benchmark the system?
  - Let’s meet ElaSQL-Bench
  - Setting up development environment
  - Testing inside a Java IDE
  - Testing with runnable JARs
  - Testing in a cluster

Testing with Runnable JARs

- In most of time, you may want to test ElaSQL in clean environments without interfere, so running with an IDE may not be a proper way.

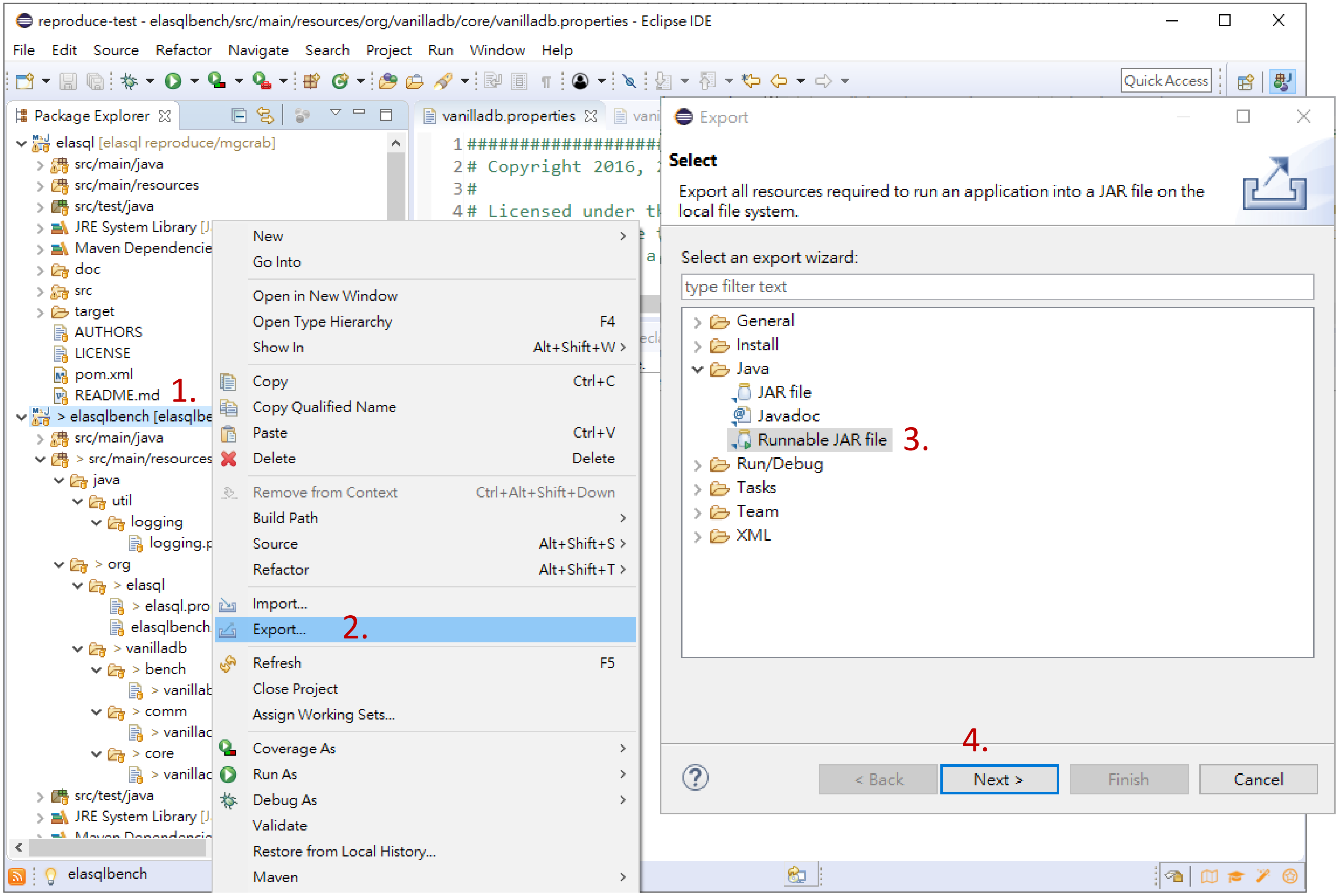
- In that case, we export the projects as runnable JARs and run with scripts.

Steps to Run with Runnable JARs

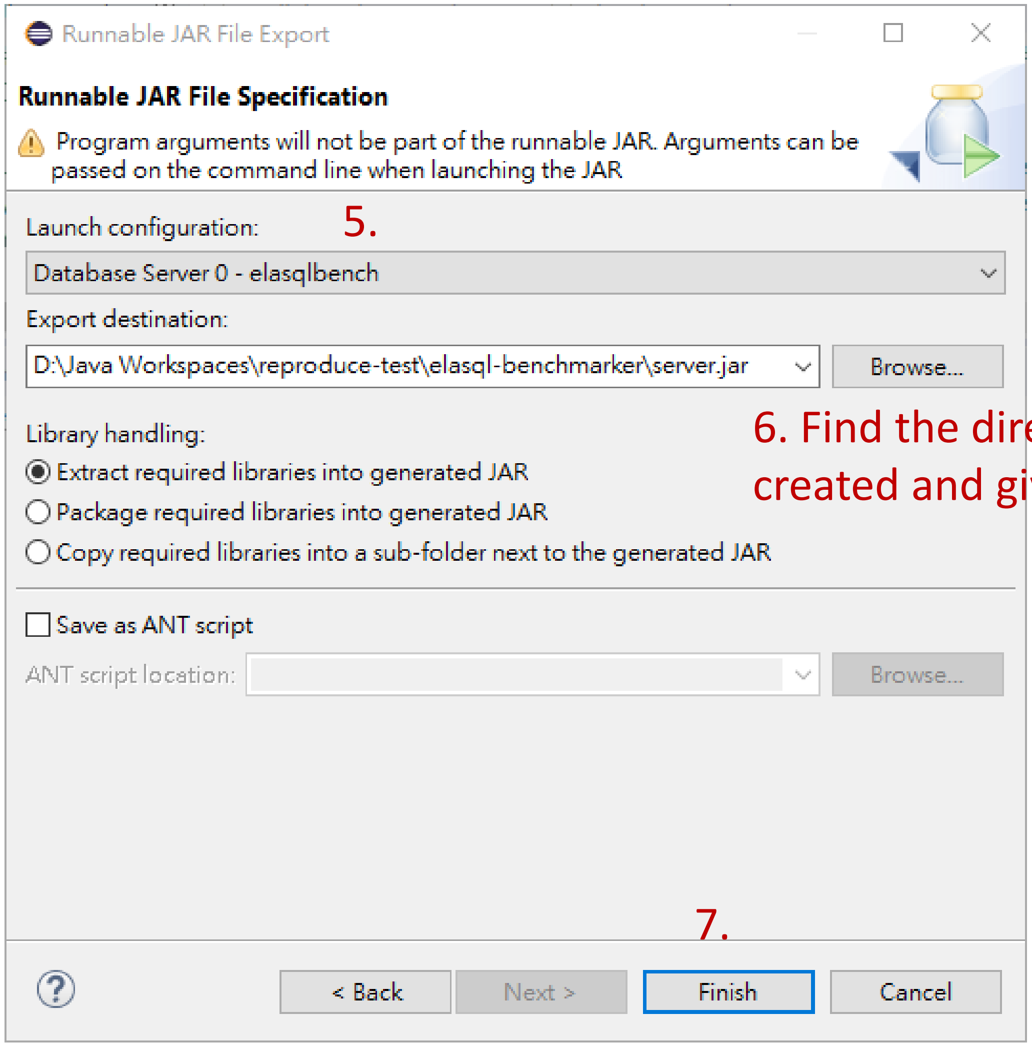
1. Create a directory to put all things together
2. Export the projects into runnable JARs
   - One for servers (including the sequencer) and one for clients
3. Copy the properties files
4. Writing scripts
5. Run with scripts!

Steps to Run with Runnable JARs

1. Create a directory to put all things together
2. Export the projects into runnable JARs
   - One for servers (including the sequencer) and one for clients
3. Copy the properties files
4. Writing scripts
5. Run with scripts!


reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
elasql [elasql reproduce/mgcrab]
src/main/java
src/main/resources
src/test/java
JRE System Library
Maven Dependencies
doc
src
target
AUTHORS
LICENSE
pom.xml
README.md
1.
> elasqlbench [elasqlbe
src/main/java
> src/main/resources
java
util
logging
logging.p
> org
> elasql
> elasql.pro
elasqlbench
> vanilladb
> bench
> vanillab
> comm
> vanillac
> core
> vanillac
src/test/java
JRE System Library
elasqlbench
vanilladb.properties
1 ###################
2 # Copyright 2016,
3 #
4 # Licensed under th
New
Go Into
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
2.
Refresh F5
Close Project
Assign Working Sets...
Coverage As
Run As
Debug As
Validate
Restore from Local History...
Maven
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
3.
Run/Debug
Tasks
Team
XML
4.
< Back
Next >
Finish
Cancel

Exporting a Server JAR



Runnable JAR File Export

Runnable JAR File Specification

⚠ Program arguments will not be part of the runnable JAR. Arguments can be passed on the command line when launching the JAR

Launch configuration: 5.

Database Server 0 - elasqlbench

Export destination:

D:\Java Workspaces\reproduce-test\elasql-benchmarker\server.jar

Library handling:

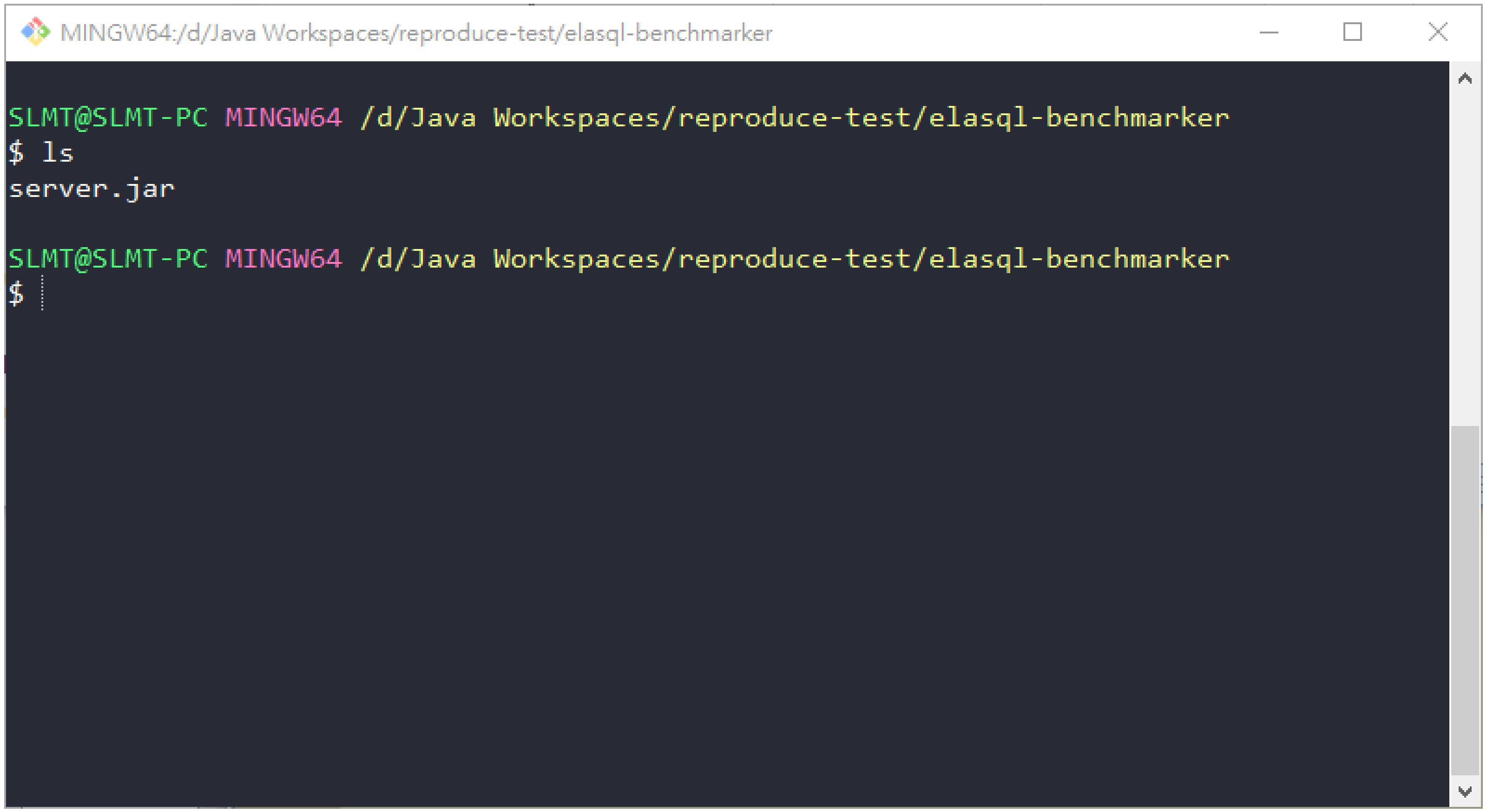
- [✓ Extract required libraries into generated JAR]
- Package required libraries into generated JAR
- Copy required libraries into a sub-folder next to the generated JAR

☐ Save as ANT script

ANT script location:

6. Find the directory you just created and give the JAR a name

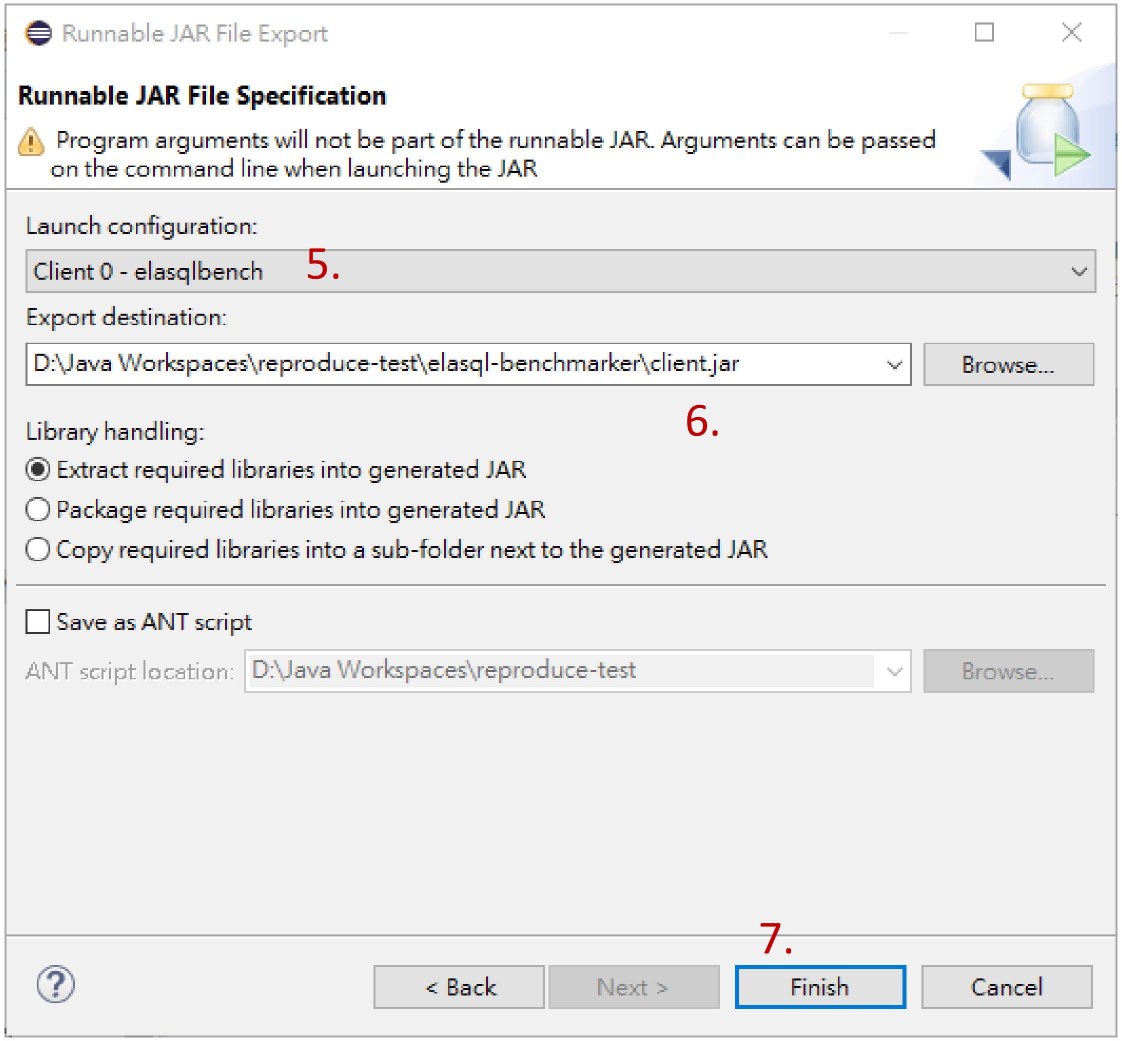
Server JAR


MINGW64:/d/Java Workspaces/reproduce-test/elasql-benchmarker

```
SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$ ls
server.jar

SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$
```

Exporting a Client JAR

Step 1~4 are same



Runnable JAR File Export

Runnable JAR File Specification

Program arguments will not be part of the runnable JAR. Arguments can be passed on the command line when launching the JAR

Launch configuration:
Client 0 - elasqlbench

5.

Export destination:
D:\Java Workspaces\reproduce-test\elasql-benchmarker\client.jar

6.

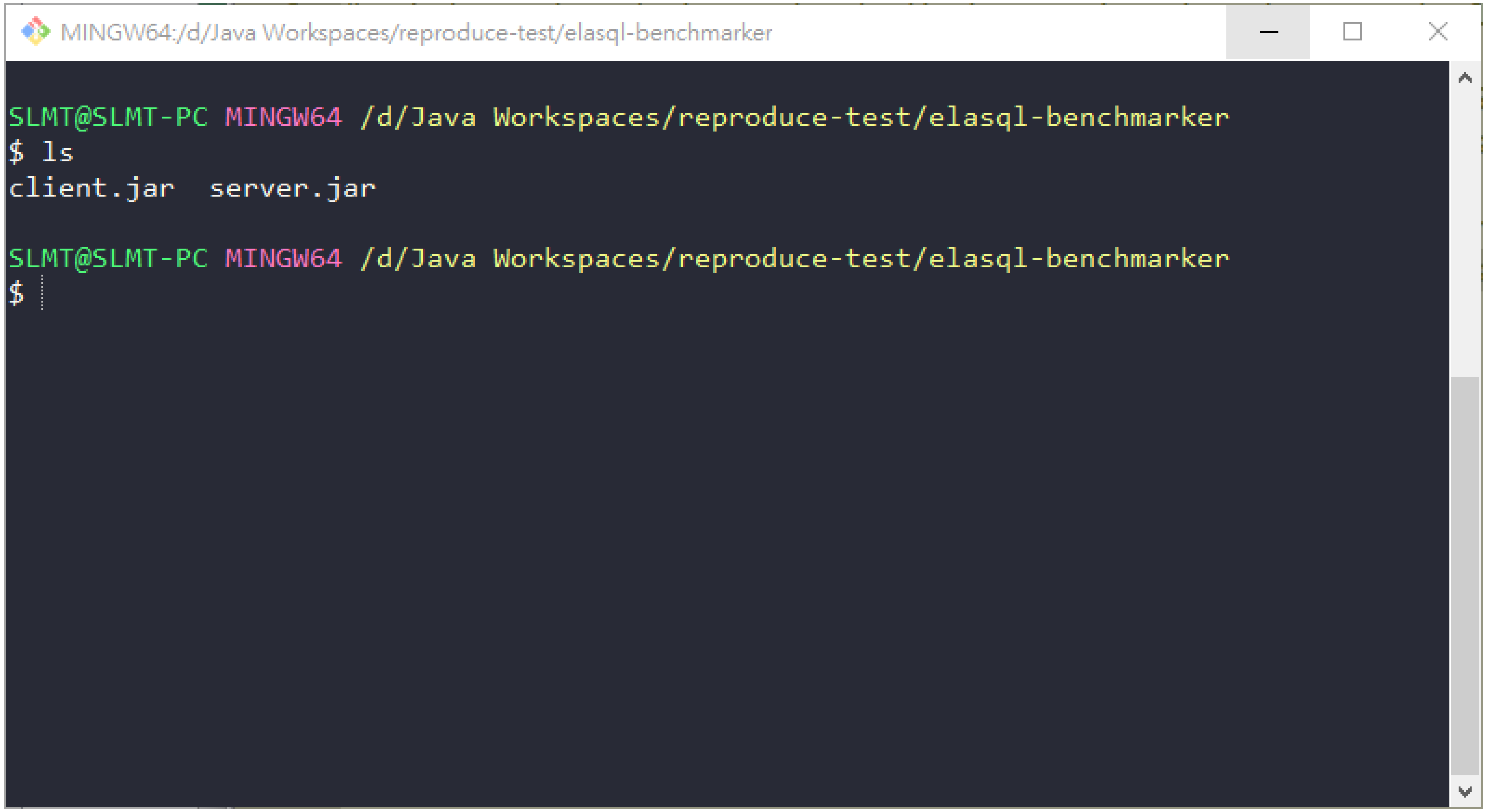
Library handling:
- [✓ Extract required libraries into generated JAR]
- Package required libraries into generated JAR
- Copy required libraries into a sub-folder next to the generated JAR

☐ Save as ANT script

ANT script location: D:\Java Workspaces\reproduce-test

7.

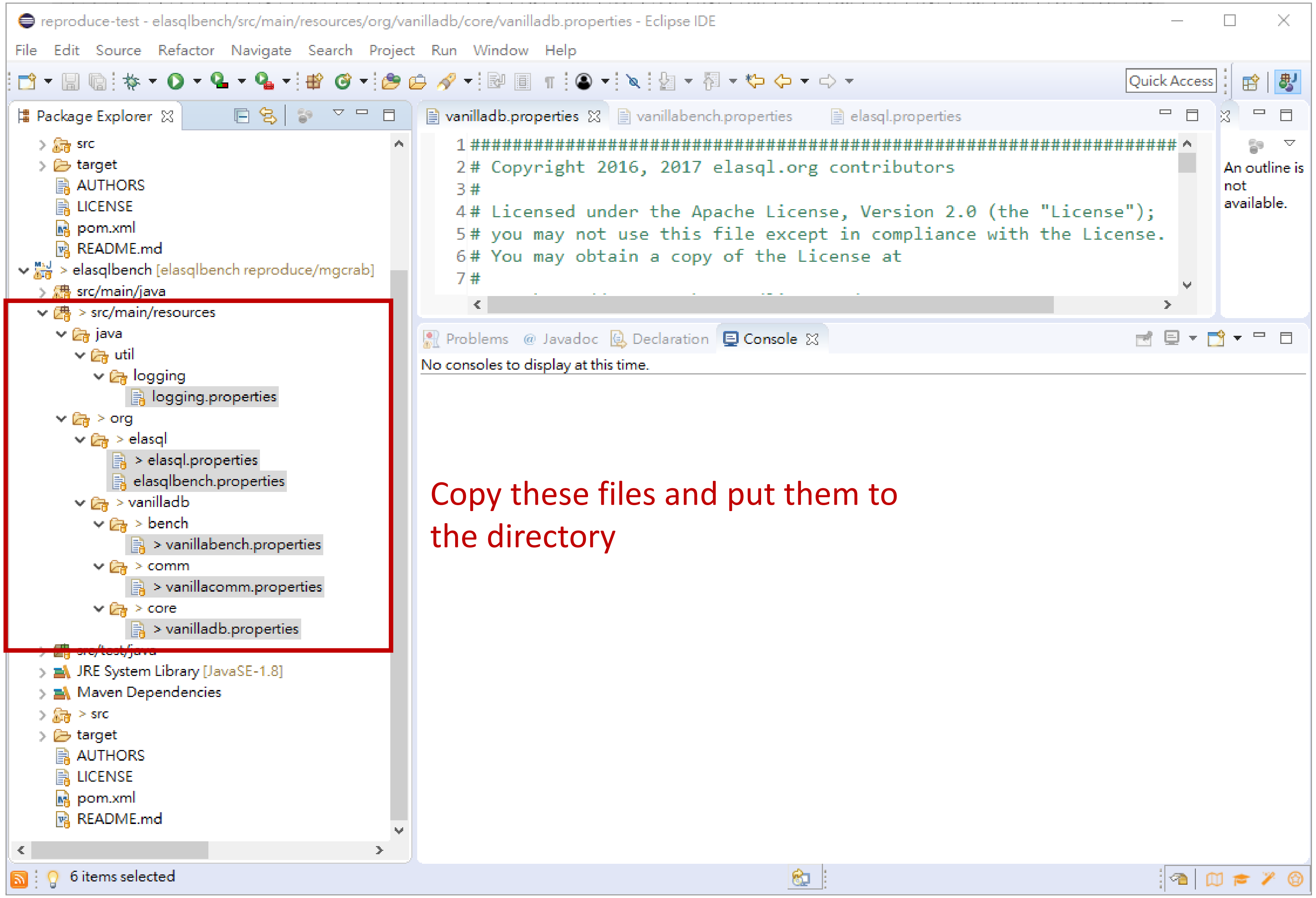
Server & Client JARs


MINGW64:/d/Java Workspaces/reproduce-test/elasql-benchmarker

```
SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$ ls
client.jar  server.jar

SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$
```

Steps to Run with Runnable JARs

1. Create a directory to put all things together
2. Export the projects into runnable JARs
    - One for servers (including the sequencer) and one for clients
3. Copy the properties files
4. Writing scripts
5. Run with scripts!



reproduce-test - elasqlbench/src/main/resources/org/vanilladb/core/vanilladb.properties - Eclipse IDE

```
src
target
AUTHORS
LICENSE
pom.xml
README.md
> elasqlbench [elasqlbench reproduce/mgcrab]
  src/main/java
  > src/main/resources
    java
      util
        logging
          logging.properties
    > org
      > elasql
        > elasql.properties
        elasqlbench.properties
      > vanilladb
        > bench
          > vanillabench.properties
        > comm
          > vanillacomm.properties
        > core
          > vanilladb.properties
  src/test/java
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  > src
  target
  AUTHORS
  LICENSE
  pom.xml
  README.md
```

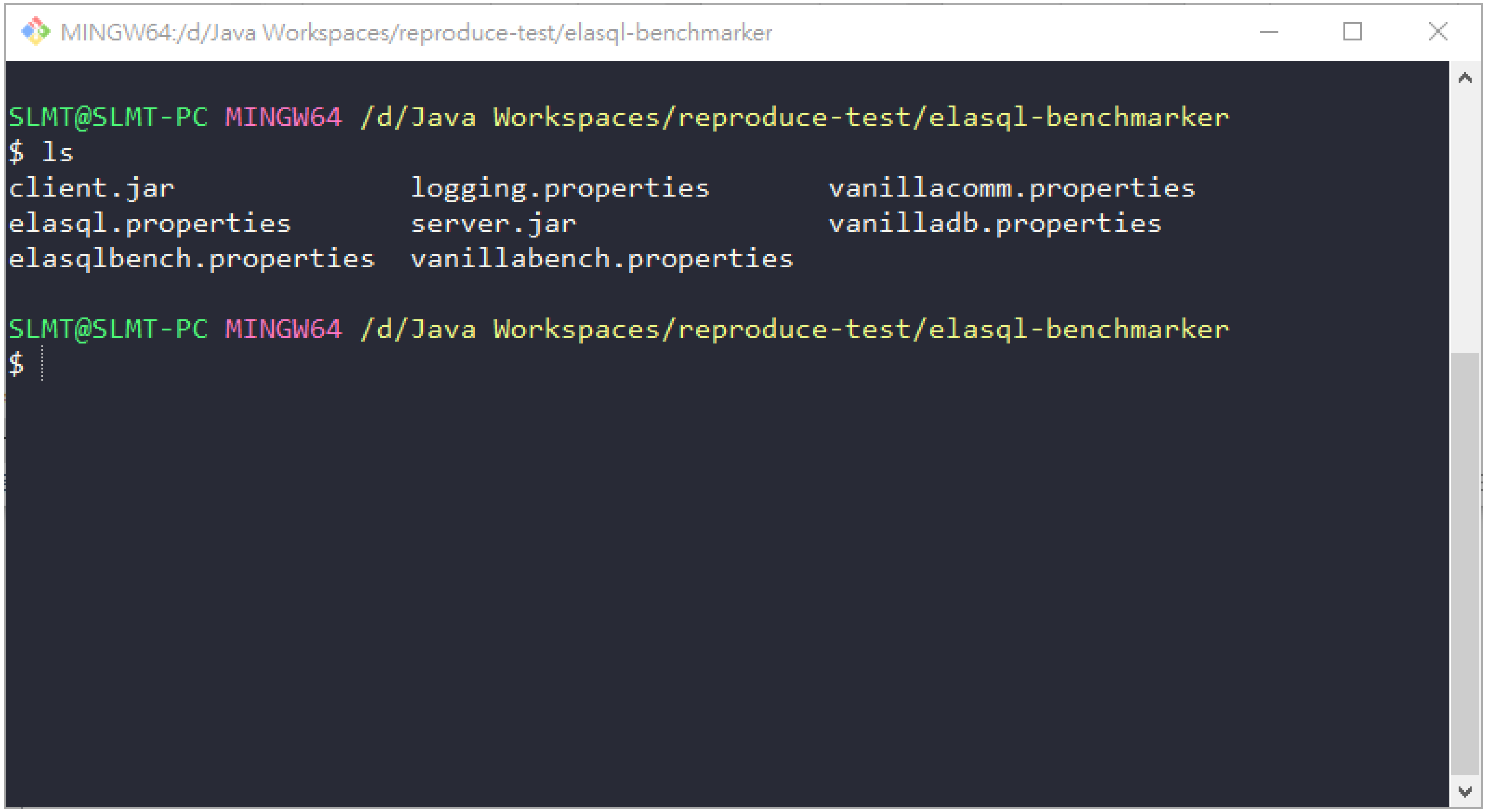
```
###################################################################
# Copyright 2016, 2017 elasql.org contributors
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
```

No consoles to display at this time.

6 items selected

Copy these files and put them to the directory

Server & Client JARs + Properties Files



MINGW64:/d/Java Workspaces/reproduce-test/elasql-benchmarker

```
SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$ ls
client.jar               logging.properties       vanillacomm.properties
elasql.properties        server.jar               vanilladb.properties
elasqlbench.properties   vanillabench.properties

SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$
```

Steps to Run with Runnable JARs

1. Create a directory to put all things together
2. Export the projects into runnable JARs
   - One for servers (including the sequencer) and one for clients
3. Copy the properties files
4. Writing scripts
5. Run with scripts!

Writing a Script for Servers (Including the Sequencer)

- Copy the script below and save it as server.sh

```
java \
-Dorg.elasql.config.file=elasql.properties \
-Dorg.elasql.bench.config.file=elasqlbench.properties \
-Dorg.vanilladb.comm.config.file=vanillacomm.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 \
```

You can also copy the content of the script from here.

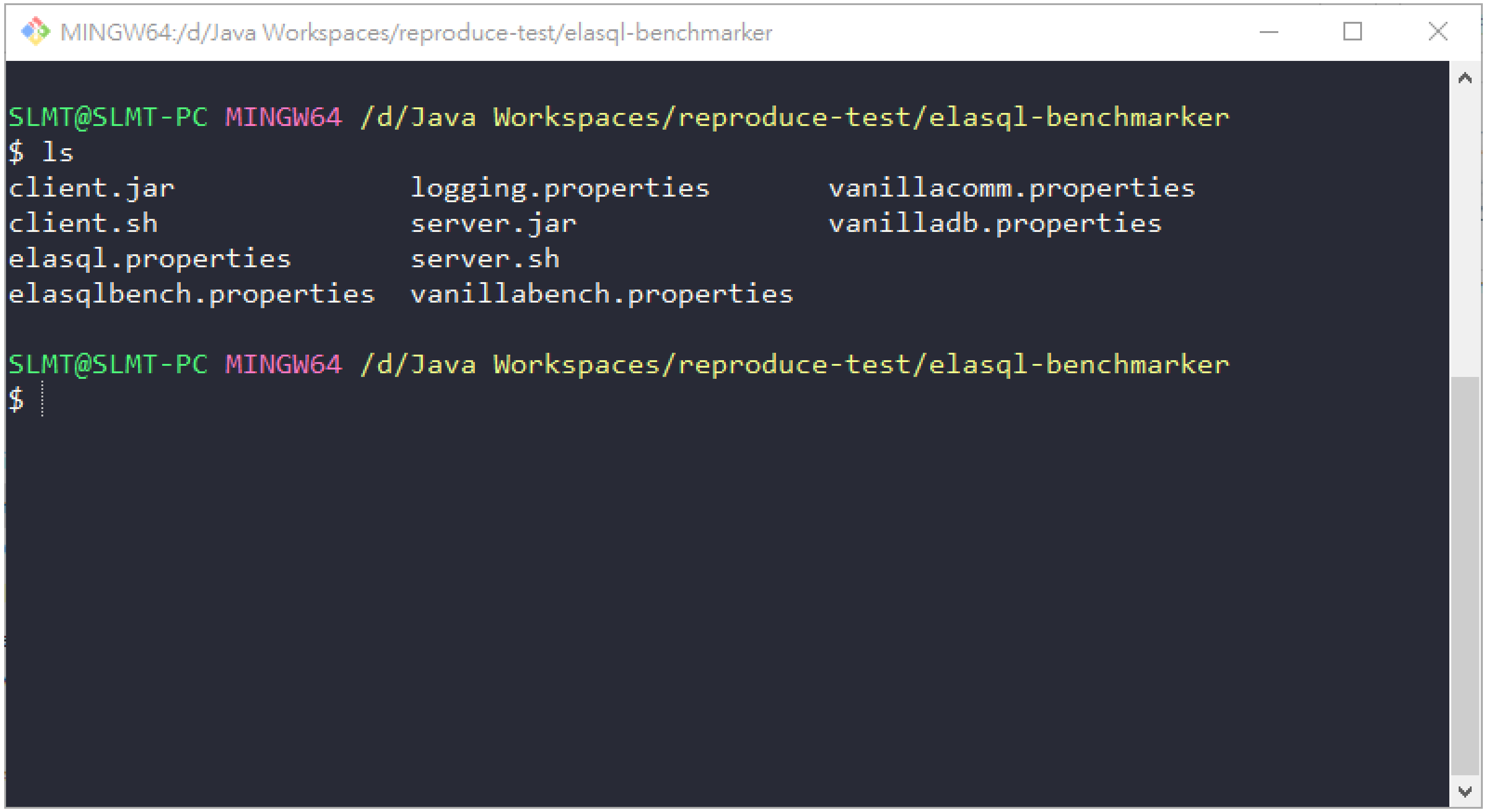
Writing a Script for Clients

- Copy the script below and save it as server.sh

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

You can also copy the content of the script from here.

All the Things We Need Are Now In Place



MINGW64:/d/Java Workspaces/reproduce-test/elasql-benchmarker

```
SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$ ls
client.jar              logging.properties      vanillacomm.properties
client.sh               server.jar              vanilladb.properties
elasql.properties       server.sh
elasqlbench.properties  vanillabench.properties

SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$
```

Steps to Run with Runnable JARs

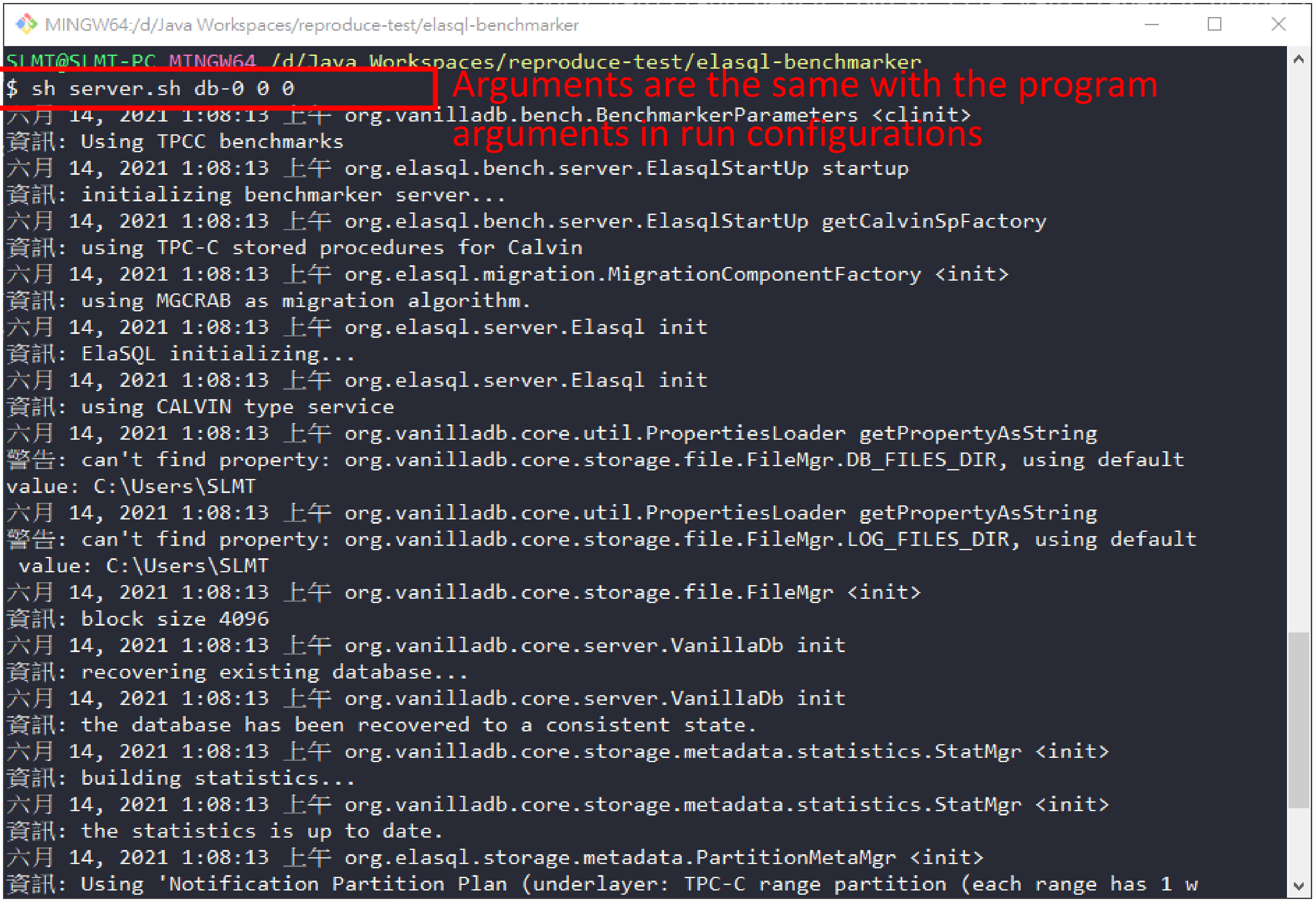
1. Create a directory to put all things together
2. Export the projects into runnable JARs
   - One for servers (including the sequencer) and one for clients
3. Copy the properties files
4. Writing scripts
5. Run with scripts!

Running the Servers & Clients

- The procedure to run the servers and clients are identical with running in Eclipse.

- The only difference is that we start processes with scripts.

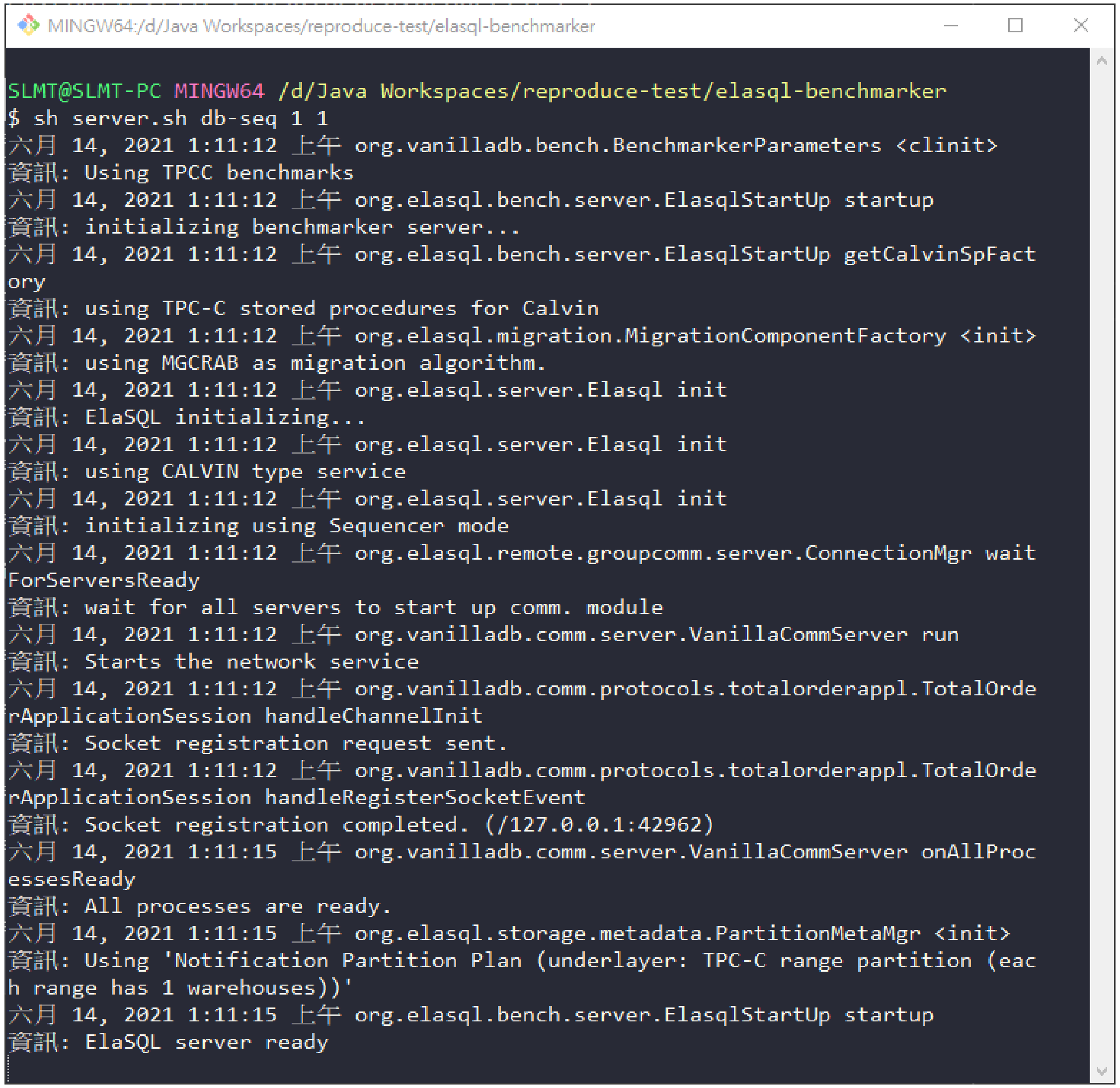
Starting a Database Server


MINGW64:/d/Java Workspaces/reproduce-test/elasql-benchmarker

Arguments are the same with the program arguments in run configurations

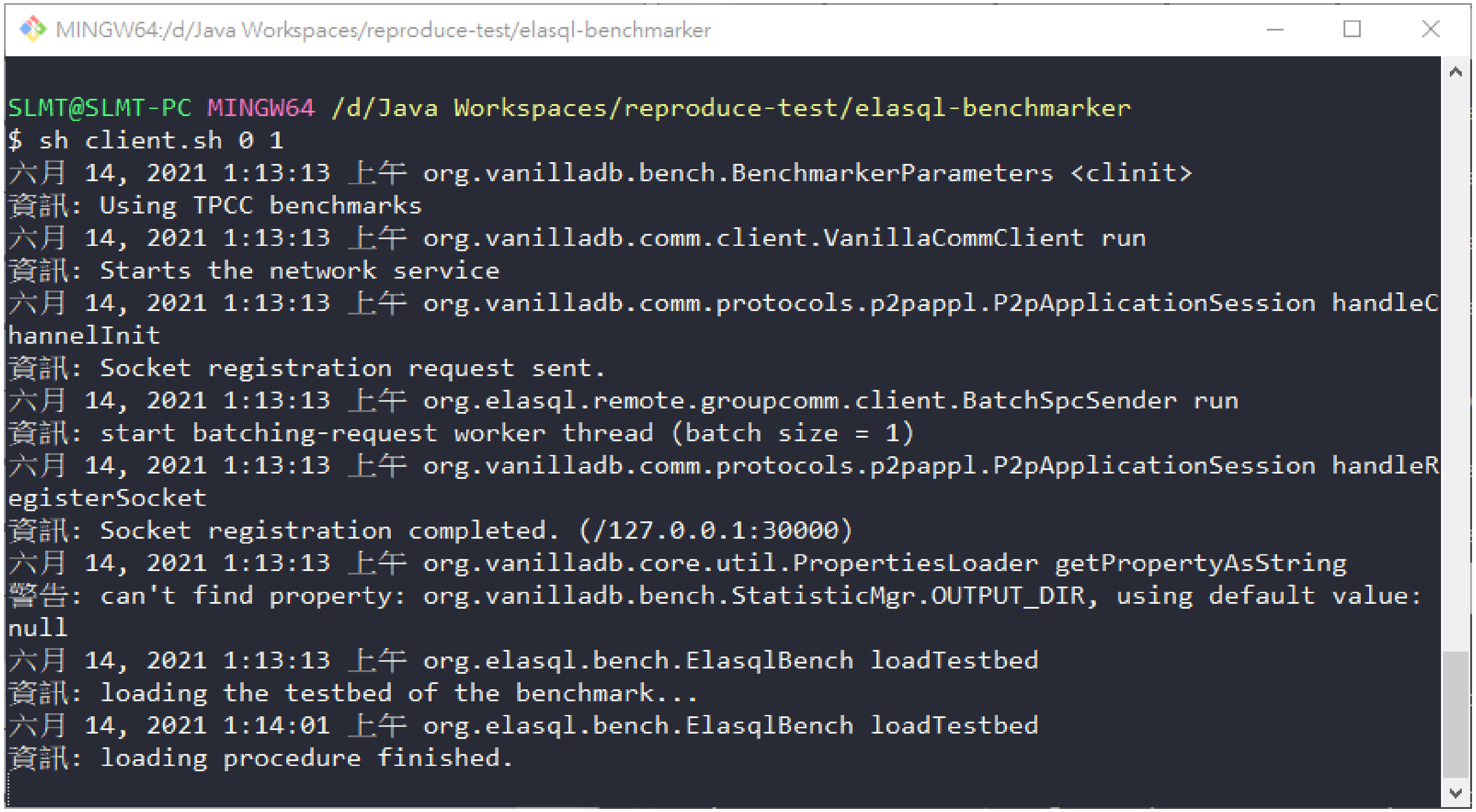
```
SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$ sh server.sh db-0 0 0
六月 14, 2021 1:08:13 上午 org.vanilladb.bench.BenchmarkerParameters <clinit>
資訊: Using TPCC benchmarks
六月 14, 2021 1:08:13 上午 org.elasql.bench.server.ElasqlStartUp startup
資訊: initializing benchmarker server...
六月 14, 2021 1:08:13 上午 org.elasql.bench.server.ElasqlStartUp getCalvinSpFactory
資訊: using TPC-C stored procedures for Calvin
六月 14, 2021 1:08:13 上午 org.elasql.migration.MigrationComponentFactory <init>
資訊: using MGCRAB as migration algorithm.
六月 14, 2021 1:08:13 上午 org.elasql.server.Elasql init
資訊: ElaSQL initializing...
六月 14, 2021 1:08:13 上午 org.elasql.server.Elasql init
資訊: using CALVIN type service
六月 14, 2021 1:08:13 上午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.core.storage.file.FileMgr.DB_FILES_DIR, using default
value: C:\Users\SLMT
六月 14, 2021 1:08:13 上午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.core.storage.file.FileMgr.LOG_FILES_DIR, using default
 value: C:\Users\SLMT
六月 14, 2021 1:08:13 上午 org.vanilladb.core.storage.file.FileMgr <init>
資訊: block size 4096
六月 14, 2021 1:08:13 上午 org.vanilladb.core.server.VanillaDb init
資訊: recovering existing database...
六月 14, 2021 1:08:13 上午 org.vanilladb.core.server.VanillaDb init
資訊: the database has been recovered to a consistent state.
六月 14, 2021 1:08:13 上午 org.vanilladb.core.storage.metadata.statistics.StatMgr <init>
資訊: building statistics...
六月 14, 2021 1:08:13 上午 org.vanilladb.core.storage.metadata.statistics.StatMgr <init>
資訊: the statistics is up to date.
六月 14, 2021 1:08:13 上午 org.elasql.storage.metadata.PartitionMetaMgr <init>
資訊: Using 'Notification Partition Plan (underlayer: TPC-C range partition (each range has 1 w
```

Starting a Sequencer Server


MINGW64:/d/Java Workspaces/reproduce-test/elasql-benchmarker

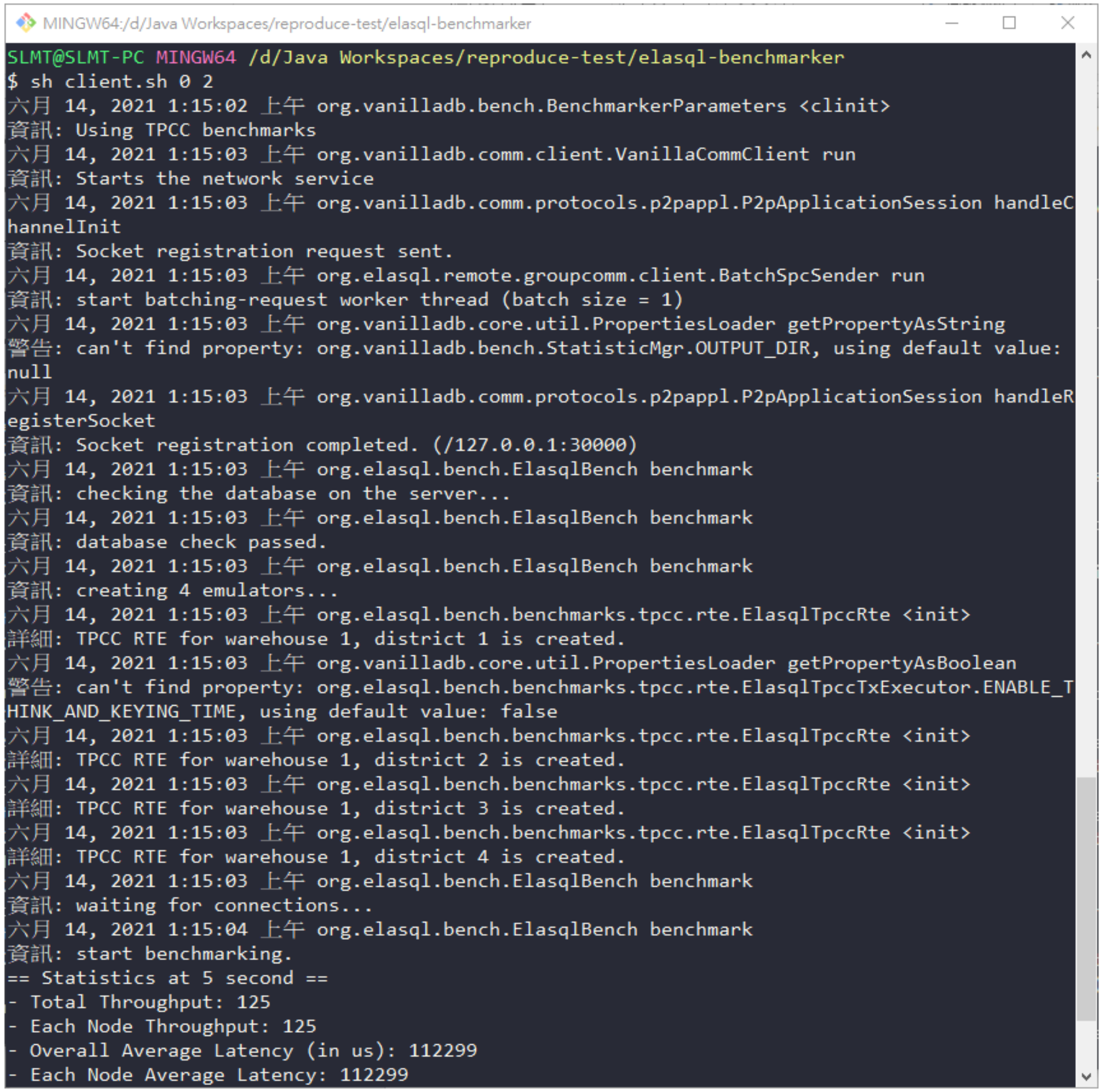
```
SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$ sh server.sh db-seq 1 1
六月 14, 2021 1:11:12 上午 org.vanilladb.bench.BenchmarkerParameters <clinit>
資訊: Using TPCC benchmarks
六月 14, 2021 1:11:12 上午 org.elasql.bench.server.ElasqlStartUp startup
資訊: initializing benchmarker server...
六月 14, 2021 1:11:12 上午 org.elasql.bench.server.ElasqlStartUp getCalvinSpFact
ory
資訊: using TPC-C stored procedures for Calvin
六月 14, 2021 1:11:12 上午 org.elasql.migration.MigrationComponentFactory <init>
資訊: using MGCRAB as migration algorithm.
六月 14, 2021 1:11:12 上午 org.elasql.server.Elasql init
資訊: ElaSQL initializing...
六月 14, 2021 1:11:12 上午 org.elasql.server.Elasql init
資訊: using CALVIN type service
六月 14, 2021 1:11:12 上午 org.elasql.server.Elasql init
資訊: initializing using Sequencer mode
六月 14, 2021 1:11:12 上午 org.elasql.remote.groupcomm.server.ConnectionMgr wait
ForServersReady
資訊: wait for all servers to start up comm. module
六月 14, 2021 1:11:12 上午 org.vanilladb.comm.server.VanillaCommServer run
資訊: Starts the network service
六月 14, 2021 1:11:12 上午 org.vanilladb.comm.protocols.totalorderappl.TotalOrde
rApplicationSession handleChannelInit
資訊: Socket registration request sent.
六月 14, 2021 1:11:12 上午 org.vanilladb.comm.protocols.totalorderappl.TotalOrde
rApplicationSession handleRegisterSocketEvent
資訊: Socket registration completed. (/127.0.0.1:42962)
六月 14, 2021 1:11:15 上午 org.vanilladb.comm.server.VanillaCommServer onAllProc
essesReady
資訊: All processes are ready.
六月 14, 2021 1:11:15 上午 org.elasql.storage.metadata.PartitionMetaMgr <init>
資訊: Using 'Notification Partition Plan (underlayer: TPC-C range partition (eac
h range has 1 warehouses))'
六月 14, 2021 1:11:15 上午 org.elasql.bench.server.ElasqlStartUp startup
資訊: ElaSQL server ready
```

Starting a Client for Loading


MINGW64:/d/Java Workspaces/reproduce-test/elasql-benchmarker

```
SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$ sh client.sh 0 1
六月 14, 2021 1:13:13 上午 org.vanilladb.bench.BenchmarkerParameters <clinit>
資訊: Using TPCC benchmarks
六月 14, 2021 1:13:13 上午 org.vanilladb.comm.client.VanillaCommClient run
資訊: Starts the network service
六月 14, 2021 1:13:13 上午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSession handleC
hannelInit
資訊: Socket registration request sent.
六月 14, 2021 1:13:13 上午 org.elasql.remote.groupcomm.client.BatchSpcSender run
資訊: start batching-request worker thread (batch size = 1)
六月 14, 2021 1:13:13 上午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSession handleR
egisterSocket
資訊: Socket registration completed. (/127.0.0.1:30000)
六月 14, 2021 1:13:13 上午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.bench.StatisticMgr.OUTPUT_DIR, using default value:
null
六月 14, 2021 1:13:13 上午 org.elasql.bench.ElasqlBench loadTestbed
資訊: loading the testbed of the benchmark...
六月 14, 2021 1:14:01 上午 org.elasql.bench.ElasqlBench loadTestbed
資訊: loading procedure finished.
```

Starting a Client for Benchmarking



MINGW64:/d/Java Workspaces/reproduce-test/elasql-benchmarker

```
SLMT@SLMT-PC MINGW64 /d/Java Workspaces/reproduce-test/elasql-benchmarker
$ sh client.sh 0 2
六月 14, 2021 1:15:02 上午 org.vanilladb.bench.BenchmarkerParameters <clinit>
資訊: Using TPCC benchmarks
六月 14, 2021 1:15:03 上午 org.vanilladb.comm.client.VanillaCommClient run
資訊: Starts the network service
六月 14, 2021 1:15:03 上午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSession handleC
hannelInit
資訊: Socket registration request sent.
六月 14, 2021 1:15:03 上午 org.elasql.remote.groupcomm.client.BatchSpcSender run
資訊: start batching-request worker thread (batch size = 1)
六月 14, 2021 1:15:03 上午 org.vanilladb.core.util.PropertiesLoader getPropertyAsString
警告: can't find property: org.vanilladb.bench.StatisticMgr.OUTPUT_DIR, using default value:
null
六月 14, 2021 1:15:03 上午 org.vanilladb.comm.protocols.p2pappl.P2pApplicationSession handleR
egisterSocket
資訊: Socket registration completed. (/127.0.0.1:30000)
六月 14, 2021 1:15:03 上午 org.elasql.bench.ElasqlBench benchmark
資訊: checking the database on the server...
六月 14, 2021 1:15:03 上午 org.elasql.bench.ElasqlBench benchmark
資訊: database check passed.
六月 14, 2021 1:15:03 上午 org.elasql.bench.ElasqlBench benchmark
資訊: creating 4 emulators...
六月 14, 2021 1:15:03 上午 org.elasql.bench.benchmarks.tpcc.rte.ElasqlTpccRte <init>
詳細: TPCC RTE for warehouse 1, district 1 is created.
六月 14, 2021 1:15:03 上午 org.vanilladb.core.util.PropertiesLoader getPropertyAsBoolean
警告: can't find property: org.elasql.bench.benchmarks.tpcc.rte.ElasqlTpccTxExecutor.ENABLE_T
HINK_AND_KEYING_TIME, using default value: false
六月 14, 2021 1:15:03 上午 org.elasql.bench.benchmarks.tpcc.rte.ElasqlTpccRte <init>
詳細: TPCC RTE for warehouse 1, district 2 is created.
六月 14, 2021 1:15:03 上午 org.elasql.bench.benchmarks.tpcc.rte.ElasqlTpccRte <init>
詳細: TPCC RTE for warehouse 1, district 3 is created.
六月 14, 2021 1:15:03 上午 org.elasql.bench.benchmarks.tpcc.rte.ElasqlTpccRte <init>
詳細: TPCC RTE for warehouse 1, district 4 is created.
六月 14, 2021 1:15:03 上午 org.elasql.bench.ElasqlBench benchmark
資訊: waiting for connections...
六月 14, 2021 1:15:04 上午 org.elasql.bench.ElasqlBench benchmark
資訊: start benchmarking.
== Statistics at 5 second ==
- Total Throughput: 125
- Each Node Throughput: 125
- Overall Average Latency (in us): 112299
- Each Node Average Latency: 112299
```

Outline

- Introduction to ElaSQL project
- How to test/benchmark the system?
  - Let's meet ElaSQL-Bench
  - Setting up development environment
  - Testing inside a Java IDE
  - Testing with runnable JARs
  - Testing in a cluster

Testing in a Cluster

- It is not hard to manually start a few servers and clients on some machines.

- However, things get mess when there are tens of servers and clients to run on a cluster.
  - Imagine to run a scalability experiment with 20 servers, 1 sequencer, and 20 clients.

- Since we have known how to run the projects with scripts, you can write your own scripts to deal with the large scale experiments.

Auto-Bencher

- Or, you can just use the one we created:
  - https://github.com/SLMT/auto-bencher

- Key Features
  - Setting up testing environments on a clean machine.
  - Deploying ElaSQL-Bench JARs to testing machines.
  - Backing up testbeds.
  - Organizing different parameters into a test set.
  - Collecting and summaries the reports from clients.

- Note: we are currently working on migrating this project to JavaScript, so the above one may get outdated soon.

Have Fun!