Assignment: Enhance Wrangler with Byte Size and Time Duration Units Parsers

Repo:
link:https://github.com/adittya003/WRANGLER_ZEOTAP_INTERN_ASSIGNMENT

1. Background

The **CDAP Wrangler** library currently parses various data types but lacks built-in support for easily handling units like Kilobytes (KB), Megabytes (MB), milliseconds (ms), or seconds (s). Users often need to perform calculations or conversions on columns representing data sizes or time intervals, which requires complex multi-step recipes.

This assignment aims to enhance the Wrangler core library by adding native support for parsing and utilizing byte size and time duration units within recipes. This involves modifying the grammar, updating the parsing logic, extending the API, and implementing a new directive to demonstrate the usage

2. Objectives

Integrate new lexer tokens (**BYTE_SIZE**, **TIME_DURATION**) into the Wrangler grammar.

Update the relevant Java code (wrangler-api, wrangler-core) to handle these new token types.

Implement a new aggregate directive (**aggregate-stats**) that utilizes these new types.

Develop comprehensive test cases, including aggregation scenarios using the new units.

3.

A)Grammar Modification (wrangler-core/src/main/antlr4/.../Directives.g4):

Directives.g4:

```
directive

 : command

 (  codeblock

   | identifier

   | macro

   | text

   | number
```

```
    | bool
    | column
    | colList
    | numberList
    | boolList
    | stringList
    | numberRanges
    | properties
    | byteSize
    | timeDuration
  )*?
  ;
value
  : String | Number | Column | Bool | BYTE_SIZE | TIME_DURATION
  ;

BYTE_SIZE
  : Digit+ ('.' Digit+)? Space* BYTE_UNIT
  ;

/**
 * Represents a byte size with units such as KB, MB, GB, etc.
 */
fragment BYTE_UNIT
  : [Kk] [Bb]?
  | [Mm] [Bb]?
  | [Gg] [Bb]?
```

```
 | [Tt] [Bb]?
 | [Pp] [Bb]?
 | [Bb]
 ;

TIME_DURATION
 : Digit+ ('.' Digit+)? Space* TIME_UNIT
 ;

/**
 * Represents a time duration with units such as seconds (s), minutes (m), hours (h), etc.
 */
fragment TIME_UNIT
 : [Nn] [Ss]
 | [Uu] [Ss]
 | [Mm] [Ss]
 | [Ss]
 | [Mm] [Ii] [Nn]
 | [Hh] ([Rr])?
 | [Dd] ([Aa] [Yy])?
 | [Ww] ([Ee] [Ee] [Kk])?
 ;

byteSize
 : BYTE_SIZE
 ;
```

timeDuration

: TIME_DURATION

;

B)API Updates (wrangler-api module):

ByteSize.java:

```
/*
 * Copyright © 2017-2019 Cask Data, Inc.
 *
 * 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
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS, WITHOUT
 * WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the
 * License for the specific language governing permissions and limitations under
 * the License.
 */

package io.cdap.wrangler.api.parser;
```

```
import com.google.gson.JsonElement;
import com.google.gson.JsonPrimitive;
```

```
/**
 * Represents a ByteSize token used in parsing directives.
 * Accepts strings like "100B", "10KB", "5MB", "2GB", "1TB", "3PB"
 * and converts them to bytes.
 */
public class ByteSize implements Token {
```

```
    private final long bytes;
```

```
    public ByteSize(String input) {
        this.bytes = parseBytes(input);
    }
```

```
    /**
     * Parses a human-readable byte size string into its equivalent bytes.
     *
     * Supported formats:
     * - "B" for bytes
     * - "KB" for kilobytes (1024 bytes)
     * - "MB" for megabytes (1024^2 bytes)
     * - "GB" for gigabytes (1024^3 bytes)
     * - "TB" for terabytes (1024^4 bytes)
     * - "PB" for petabytes (1024^5 bytes)
     *
     * @param input The input string like "10MB", "2GB", "5TB", etc.
```

```
     * @return The number of bytes as a long
     * @throws IllegalArgumentException if the unit is unrecognized
     */
    private long parseBytes(String input) {
        input = input.trim().toUpperCase();
        if (input.endsWith("PB")) {
            return Long.parseLong(input.replace("PB", "")) * 1024L * 1024 * 1024 * 1024 * 1024;
        }
        if (input.endsWith("TB")) {
            return Long.parseLong(input.replace("TB", "")) * 1024L * 1024 * 1024 * 1024;
        }
        if (input.endsWith("GB")) {
            return Long.parseLong(input.replace("GB", "")) * 1024L * 1024 * 1024;
        }
        if (input.endsWith("MB")) {
            return Long.parseLong(input.replace("MB", "")) * 1024L * 1024;
        }
        if (input.endsWith("KB")) {
            return Long.parseLong(input.replace("KB", "")) * 1024L;
        }
        if (input.endsWith("B")) {
            return Long.parseLong(input.replace("B", ""));
        }
        throw new IllegalArgumentException("Invalid byte size: " + input);
    }
```

```
    public static long convertBytesToUnit(long bytes, String unit) {
        switch (unit.toLowerCase()) {
            case "b":
                return bytes;
            case "kb":
                return bytes / 1024;
            case "mb":
                return bytes / (1024 * 1024);
            case "gb":
                return bytes / (1024 * 1024 * 1024);
            case "tb":
                return bytes / (1024L * 1024 * 1024 * 1024);
            case "pb":
                return bytes / (1024L * 1024 * 1024 * 1024 * 1024);
            default:
                throw new IllegalArgumentException("Unsupported size unit: " + unit);
        }
    }
```

```
    public long getBytes() {
        return bytes;
    }
```

```
    @Override
    public Object value() {
        return bytes;
    }
```

```
    @Override
    public TokenType type() {
        return TokenType.BYTE_SIZE;
    }
```

```
    @Override
```

```
    public JsonElement toJson() {
        return new JsonPrimitive(bytes);
    }
}
```

TimeDuration.java:

```
/*
 * Copyright © 2017-2019 Cask Data, Inc.
 *
 * 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
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS, WITHOUT
 * WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the
 * License for the specific language governing permissions and limitations under
 * the License.
 */

package io.cdap.wrangler.api.parser;
```

```
import com.google.gson.JsonElement;
import com.google.gson.JsonPrimitive;
```

```
/**
 * Represents a TimeDuration token used in parsing directives.
 * Accepts strings like "10s", "5m", "2h", "100ms", "3d", "1w", "500us",
 * "1000ns"
 * and converts them to milliseconds.
 */
public class TimeDuration implements Token {
```

```
    // Stores the duration in milliseconds after parsing.
    private final long milliseconds;
```

```
    /**
     * Constructs the TimeDuration token by parsing the input.
     *
     * @param input The time duration string (e.g., "10s", "100ms").
     */
    public TimeDuration(String input) {
        this.milliseconds = parseDuration(input);
    }
```

```
    /**
     * Parses a human-readable time duration string into milliseconds.
     * Only whole-number durations are supported.
     *
     * @param input The input string (e.g., "10s", "5m").
     * @return The duration in milliseconds as a long.
     * @throws IllegalArgumentException if the unit is unrecognized.
     */
    private long parseDuration(String input) {
        input = input.trim().toLowerCase();
        if (input.endsWith("ns")) {
```

```
            return Long.parseLong(input.replace("ns", "").trim()) / 1_000_000;
        }
        if (input.endsWith("us")) {
            return Long.parseLong(input.replace("us", "").trim()) / 1_000;
        }
        if (input.endsWith("ms")) {
            return Long.parseLong(input.replace("ms", "").trim());
        }
        if (input.endsWith("s")) {
            return Long.parseLong(input.replace("s", "").trim()) * 1000L;
        }
        if (input.endsWith("m")) {
            return Long.parseLong(input.replace("m", "").trim()) * 60 * 1000L;
        }
        if (input.endsWith("h")) {
            return Long.parseLong(input.replace("h", "").trim()) * 60 * 60 * 1000L;
        }
        if (input.endsWith("d")) {
            return Long.parseLong(input.replace("d", "").trim()) * 24 * 60 * 60 * 1000L;
        }
        if (input.endsWith("w")) {
            return Long.parseLong(input.replace("w", "").trim()) * 7 * 24 * 60 * 60 * 1000L;
        }
        throw new IllegalArgumentException("Invalid time duration: " + input);
    }
```

```
    /**
     * Converts milliseconds to the specified unit.
     *
     * @param millis The milliseconds.
     * @param unit   The target unit (e.g., "s", "m").
     * @return The converted value as a long.
     */
    public static long convertMillisecondsToUnit(long millis, String unit) {
        switch (unit.toLowerCase()) {
            case "ms":
                return millis;
            case "s":
                return millis / 1000;
            case "m":
                return millis / (1000 * 60);
            case "h":
                return millis / (1000 * 60 * 60);
            case "d":
                return millis / (1000 * 60 * 60 * 24);
            case "w":
                return millis / (1000 * 60 * 60 * 24 * 7);
            default:
                throw new IllegalArgumentException("Unsupported time unit: " + unit);
        }
    }
```

```
    public long getMilliseconds() {
        return milliseconds;
    }
```

```
    @Override
    public Object value() {
        return milliseconds;
    }
```

```
    @Override
    public TokenType type() {
        return TokenType.TIME_DURATION;
    }
```

```
    @Override
    public JsonElement toJson() {
        return new JsonPrimitive(milliseconds);
    }
}
```

c)Core Parser Updates (wrangler-core module):

Updated RecipeVisitor:

```
/**
* Visitor method for parsing ByteSize tokens.
*
* This method is triggered when the parser encounters a BYTE_SIZE token
* in the directive recipe (e.g., "10KB", "2MB", "1GB", etc.). It extracts the
* raw string using ctx.getText(), constructs a ByteSize token object (which
* internally parses and stores the value in bytes), and adds it to the current
* TokenGroup being built.
*
* @param ctx The parsing context for the BYTE_SIZE token.
* @return The updated RecipeSymbol.Builder with the new ByteSize token added.
*/

@Override
public RecipeSymbol.Builder visitByteSize(DirectivesParser.ByteSizeContext ctx) {
  builder.addToken(new ByteSize(ctx.getText()));
  return builder;
}
```

```
/**
* Visitor method for parsing TimeDuration tokens.
*
* This method is invoked when the parser encounters a TIME_DURATION token
* in the directive recipe (e.g., "100ms", "5s", "3h", "2d", etc.). It takes
* the raw string from the parse context, constructs a TimeDuration token
* (which parses and stores the value in milliseconds), and adds it to the
* TokenGroup.
*
* @param ctx The parsing context for the TIME_DURATION token.
* @return The updated RecipeSymbol.Builder with the new TimeDuration token added.
*/
```

```
@Override
public RecipeSymbol.Builder visitTimeDuration(DirectivesParser.TimeDurationContext ctx) {
  builder.addToken(new TimeDuration(ctx.getText()));
  return builder;
}
```

```
}
```

Updated TokenType:

```
  /**
   * Represents the enumerated type for the object of type {@code ByteSize}.
   * This type is associated with a token that represents a byte size value.
   */
  BYTE_SIZE,

  /**
   * Represents the enumerated type for the object of type {@code TimeDuration}.
   * This type is associated with a token that represents a time duration value.
   */
  TIME_DURATION
```

D and E:

AggerateSizeDuration.java:

```
/*
 * Copyright © 2017-2019 Cask Data, Inc.
 *
 * 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
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS, WITHOUT
 * WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the
 * License for the specific language governing permissions and limitations under
 * the License.
 */

package io.cdap.directives.aggregates;
```

```
import io.cdap.cdap.api.annotation.Plugin;
import io.cdap.wrangler.api.Arguments;
import io.cdap.wrangler.api.Directive;
import io.cdap.wrangler.api.DirectiveExecutionException;
import io.cdap.wrangler.api.DirectiveParseException;
import io.cdap.wrangler.api.ErrorRowException;
import io.cdap.wrangler.api.ExecutorContext;
import io.cdap.wrangler.api.ReportErrorAndProceed;
import io.cdap.wrangler.api.Row;
import io.cdap.wrangler.api.parser.ByteSize;
import io.cdap.wrangler.api.parser.ColumnName;
```

```
import io.cdap.wrangler.api.parser.Text;
import io.cdap.wrangler.api.parser.TimeDuration;
import io.cdap.wrangler.api.parser.TokenType;
import io.cdap.wrangler.api.parser.UsageDefinition;
```

```
import java.util.Collections;
import java.util.List;
```

```
/**
 * A directive that aggregates byte size and time duration values across rows.
 * Computes sum or average of specified columns, converts results to target
 * units, and outputs a single aggregated row.
 */
@Plugin(type = Directive.TYPE)
public class AggregateSizeDuration implements Directive {
```

```
    private String sourceSizeColumn;
    private String sourceTimeColumn;
    private String targetSizeColumn;
    private String targetTimeColumn;
    private String sizeUnit;
    private String timeUnit;
    private String aggregationType;
```

```
    // Running totals; totalSize is a long to hold fractional sizes.
    private long totalSize = 0;
    private long totalTime = 0;
    private long count = 0;
```

```
    @Override
    public UsageDefinition define() {
        UsageDefinition.Builder builder = UsageDefinition.builder("AggregateSizeDuration");
        builder.define("sourceSizeColumn", TokenType.COLUMN_NAME);
        builder.define("sourceTimeColumn", TokenType.COLUMN_NAME);
        builder.define("targetSizeColumn", TokenType.COLUMN_NAME);
        builder.define("targetTimeColumn", TokenType.COLUMN_NAME);
        builder.define("sizeUnit", TokenType.TEXT, true);
        builder.define("timeUnit", TokenType.TEXT, true);
        builder.define("aggregationType", TokenType.TEXT, true);
        return builder.build();
    }
```

```
    @Override
    public void initialize(Arguments args) throws DirectiveParseException {
        try {
            ColumnName sourceSize = (ColumnName) args.value("sourceSizeColumn");
            ColumnName sourceTime = (ColumnName) args.value("sourceTimeColumn");
            ColumnName targetSize = (ColumnName) args.value("targetSizeColumn");
            ColumnName targetTime = (ColumnName) args.value("targetTimeColumn");
```

```
            this.sourceSizeColumn = sourceSize.value();
            this.sourceTimeColumn = sourceTime.value();
            this.targetSizeColumn = targetSize.value();
            this.targetTimeColumn = targetTime.value();
```

```
            if (args.contains("sizeUnit")) {
                Text unit = (Text) args.value("sizeUnit");
                this.sizeUnit = unit.value();
            }
            if (args.contains("timeUnit")) {
```

```
                Text tunit = (Text) args.value("timeUnit");
                this.timeUnit = tunit.value();
            }
            if (args.contains("aggregationType")) {
                Text aggTypeText = (Text) args.value("aggregationType");
                this.aggregationType = aggTypeText.value();
            }
        } catch (Exception e) {
            throw new DirectiveParseException(
                    "Failed to parse arguments for AggregateSizeDuration directive.", e);
        }
    }
```

```
    @Override
    public List<Row> execute(List<Row> rows, ExecutorContext context)
            throws DirectiveExecutionException, ErrorRowException, ReportErrorAndProceed {
        for (Row row : rows) {
            try {
                Object sizeVal = row.getValue(sourceSizeColumn);
                Object timeVal = row.getValue(sourceTimeColumn);
```

```
                if (sizeVal == null || timeVal == null) {
                    throw new DirectiveExecutionException(
                            "Null encountered in required fields: " + sourceSizeColumn + " or
" + sourceTimeColumn);
                }
```

```
                // Get the size in bytes as a long.
                long sizeBytes = new ByteSize(sizeVal.toString()).getBytes();
                long durationMillis = new TimeDuration(timeVal.toString()).getMilliseconds();
```

```
                totalSize += sizeBytes;
                totalTime += durationMillis;
                count++;
            } catch (Exception e) {
                throw new DirectiveExecutionException("Error processing row: " +
row.toString(), e);
            }
        }
        return Collections.emptyList();
    }
```

```
    public List<Row> finalize(ExecutorContext context) throws DirectiveExecutionException {
        long finalSizeBytes = totalSize;
        long finalTimeMillis = totalTime;
```

```
        if ("avg".equalsIgnoreCase(aggregationType) && count > 0) {
            finalSizeBytes = totalSize / count;
            finalTimeMillis = totalTime / count;
        }
```

```
        String sizeTargetUnit = (sizeUnit != null && !sizeUnit.isEmpty()) ? sizeUnit : "MB";
        String timeTargetUnit = (timeUnit != null && !timeUnit.isEmpty()) ? timeUnit : "s";
```

```
        double convertedSize;
        long convertedTime;
        try {
            convertedSize = ByteSize.convertBytesToUnit(finalSizeBytes, sizeTargetUnit);
        } catch (IllegalArgumentException e) {
```

```
            throw new DirectiveExecutionException("Invalid size unit specified: " +
sizeTargetUnit, e);
        }
        try {
            convertedTime = TimeDuration.convertMillisecondsToUnit(finalTimeMillis,
timeTargetUnit);
        } catch (IllegalArgumentException e) {
            throw new DirectiveExecutionException("Invalid time unit specified: " +
timeTargetUnit, e);
        }
```

```
        Row resultRow = new Row();
        resultRow.add(targetSizeColumn, convertedSize);
        resultRow.add(targetTimeColumn, convertedTime);
```

```
        return Collections.singletonList(resultRow);
    }
```

```
    @Override
    public void destroy() {
        // No resources to clean up in this implementation.
    }
}
```

Test Modules:

ByteSizeTest.java

```
/*
 * Copyright © 2017-2019 Cask Data, Inc.
 *
 * 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
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS, WITHOUT
 * WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the
 * License for the specific language governing permissions and limitations under
 * the License.
 */

package io.cdap.wrangler.parser;
```

```
import io.cdap.wrangler.api.parser.ByteSize;
import org.junit.Assert;
import org.junit.Test;
```

```
/**
```

```
 * Tests for the ByteSize token.
 */
public class ByteSizeTest {
```

```
    /**
     * Verifies parsing of valid byte size units.
     * Note: Decimal values (e.g., "1.5MB") are not supported in this version.
     */
    @Test
    public void testByteParsing() {
        Assert.assertEquals(1024L, new ByteSize("1KB").getBytes());
        Assert.assertEquals(1048576L, new ByteSize("1MB").getBytes());
        Assert.assertEquals(1073741824L, new ByteSize("1GB").getBytes());
        Assert.assertEquals(1L, new ByteSize("1B").getBytes());
    }
```

```
    /**
     * Ensures an exception is thrown for invalid units.
     */
    @Test(expected = IllegalArgumentException.class)
    public void testInvalidUnit() {
        new ByteSize("5XY").getBytes();
    }
}
```

TimeDurationTest.java

```
/*
 * Copyright © 2017-2019 Cask Data, Inc.
 *
 * 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
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS, WITHOUT
 * WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the
 * License for the specific language governing permissions and limitations under
 * the License.
 */

package io.cdap.wrangler.parser;
```

```
import io.cdap.wrangler.api.parser.TimeDuration;
import org.junit.Assert;
import org.junit.Test;
```

```
/**
 * Tests for the TimeDuration token.
```

```
 */
public class TimeDurationTest {
```

```
    @Test
    public void testDurationParsing() {
        Assert.assertEquals(5000L, new TimeDuration("5s").getMilliseconds());
        Assert.assertEquals(2L, new TimeDuration("2ms").getMilliseconds());
        Assert.assertEquals(60000L, new TimeDuration("1m").getMilliseconds());
        Assert.assertEquals(3600000L, new TimeDuration("1h").getMilliseconds());
    }
```

```
    @Test(expected = IllegalArgumentException.class)
    public void testInvalidUnit() {
        new TimeDuration("10lightyears").getMilliseconds();
    }
}
```

AggregateSizeDuration.java:

```
/*
 * Copyright © 2017-2019 Cask Data, Inc.
 *
 * 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
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS, WITHOUT
 * WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the
 * License for the specific language governing permissions and limitations under
 * the License.
 */
package io.cdap.directives.aggregates;

import io.cdap.wrangler.TestingRig;
import io.cdap.wrangler.api.Row;
import org.junit.Assert;
import org.junit.Test;
```

```
import java.util.Arrays;
import java.util.List;
```

```
/**
 * Tests {@link AggregateSizeDuration}
 */
public class AggregateSizeDurationTest {
```

```
        @Test
        public void testAggregateSumMBAndSeconds() throws Exception {
                // Create sample rows with whole numbers
                List<Row> inputRows = Arrays.asList(
                                new Row().add("data_transfer_size", "1MB").add("response_time",
"2s"),
                                new Row().add("data_transfer_size",
"512KB").add("response_time", "1s"),
```

```
                                new Row().add("data_transfer_size", "2MB").add("response_time",
"2s"));

                String[] recipe = new String[] {
                                "aggregate-sizeduration :data_transfer_size :response_time
total_size_mb " +
                                                                "total_time_sec MB s"
                };

                List<Row> results = TestingRig.execute(recipe, inputRows);

                // Expected: 1MB + 0.5MB + 2MB = 3.5MB and 2s + 1s + 2s = 5s (whole numbers
                // only)
                Assert.assertEquals(1, results.size());
                long actualSize = (Long) results.get(0).getValue("total_size_mb"); // Use long
for whole numbers
                long actualTime = (Long) results.get(0).getValue("total_time_sec"); // Use
long for whole numbers

                Assert.assertEquals(3, actualSize); // Adjusted for whole numbers
                Assert.assertEquals(5, actualTime); // Adjusted for whole numbers
        }

        @Test
        public void testAggregateAvgBytesAndMs() throws Exception {
                // Create sample rows with whole numbers
                List<Row> inputRows = Arrays.asList(
                                new Row().add("data_transfer_size",
"1024B").add("response_time", "100ms"),
                                new Row().add("data_transfer_size",
"2048B").add("response_time", "300ms"));

                String[] recipe = new String[] {
                                "aggregate-sizeduration :data_transfer_size :response_time
avg_size avg_time B ms avg"
                };

                List<Row> results = TestingRig.execute(recipe, inputRows);

                // Expected average: (1024 + 2048) / 2 = 1536B and (100 + 300) / 2 = 200ms
                // (whole numbers only)
                Assert.assertEquals(1, results.size());
                long actualSize = (Long) results.get(0).getValue("avg_size"); // Use long for
whole numbers
                long actualTime = (Long) results.get(0).getValue("avg_time"); // Use long for
whole numbers

                Assert.assertEquals(1536, actualSize); // Adjusted for whole numbers
                Assert.assertEquals(200, actualTime); // Adjusted for whole numbers
        }
}
```

Output:

ByteSizeTest:

```
PS C:\Users\aditt\Desktop\Coding\wrangler> cd wrangler-core
PS C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core> mvn -Dtest=ByteSizeTest test
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------< io.cdap.wrangler:wrangler-core >------------------
[INFO] Building Wrangler Core 4.12.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[WARNING] 1 problem was encountered while building the effective model for org.apache.hadoop:hadoop-annotations:jar:2.6.5 during dependency collection step for
project (use -X to see details)
[WARNING] 2 problems were encountered while building the effective model for org.apache.yetus:audience-annotations:jar:0.5.0 during dependency collection step f
or project (use -X to see details)
[WARNING] 1 problem was encountered while building the effective model for org.javassist:javassist:jar:3.18.2-GA during dependency collection step for project (
use -X to see details)
[INFO]
[INFO] --- apache-rat:0.10:check (rat-check) @ wrangler-core ---
[INFO] 51 implicit excludes (use -debug for more details).
[INFO] Exclude: cov-int/**
[INFO] Exclude: *.md
[INFO] Exclude: **/*.md
[INFO] Exclude: **/*.json
[INFO] Exclude: **/resources/**
[INFO] Exclude: wrangler-demos/**
[INFO] Exclude: **/com/example/**
[INFO] Exclude: /**/icons/**
[INFO] 307 resources included (use -debug for more details)
[INFO] Rat check: Summary of files. Unapproved: 0 unknown: 0 generated: 0 approved: 307 licence.
[INFO]
[INFO] --- antlr4:4.7:antlr4 (default) @ wrangler-core ---
[INFO] No grammars to process
[INFO] ANTLR 4: Processing source directory C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core\src\main\antlr4
[INFO]
[INFO] --- buildnumber:1.0:create (default) @ wrangler-core ---
[INFO] Storing buildNumber: 2025-04-13-19:48:43_aditt at timestamp: 1744553923231
[INFO] Executing: cmd.exe /X /C "git show"
[INFO] Working directory: C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core
[INFO] Storing buildScmBranch: UNKNOWN
```

```
[INFO] Storing buildScmBranch: UNKNOWN
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ wrangler-core ---
[INFO] Copying 12 resources from src\main\resources to target\classes
[INFO] The encoding used to copy filtered properties files have not been set. This means that the same encoding will be used to copy filtered properties files a
s when copying other filtered resources. This might not be what you want! Run your build with --debug to see which files might be affected. Read more at https:/
/maven.apache.org/plugins/maven-resources-plugin/examples/filtering-properties-files.html
[INFO]
[INFO] --- compiler:3.13.0:compile (default-compile) @ wrangler-core ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ wrangler-core ---
[INFO] Copying 5 resources from src\test\resources to target\test-classes
[INFO]
[INFO] --- compiler:3.13.0:testCompile (default-testCompile) @ wrangler-core ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] --- checkstyle:2.17:check (validate) @ wrangler-core ---
[WARNING]  Parameter 'sourceDirectory' is deprecated: instead use {@link #sourceDirectories}
[INFO] Starting audit...
Audit done.
[INFO]
[INFO] --- surefire:2.14.1:test (default-test) @ wrangler-core ---
[INFO] Surefire report directory: C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core\target\surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running io.cdap.wrangler.parser.ByteSizeTest
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.074 sec

Results :

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

```
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  15.429 s
[INFO] Finished at: 2025-04-13T19:48:52+05:30
[INFO] ------------------------------------------------------------------------
```

TimeDurationTest:

```
PS C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core> mvn -Dtest=TimeDurationTest test
[INFO] Scanning for projects...
[INFO]
[INFO] -------------------< io.cdap.wrangler:wrangler-core >-------------------
[INFO] Building Wrangler Core 4.12.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[WARNING] 1 problem was encountered while building the effective model for org.apache.hadoop:hadoop-annotations:jar:2.6.5 during dependency collection step for
project (use -X to see details)
[WARNING] 2 problems were encountered while building the effective model for org.apache.yetus:audience-annotations:jar:0.5.0 during dependency collection step f
or project (use -X to see details)
[WARNING] 1 problem was encountered while building the effective model for org.javassist:javassist:jar:3.18.2-GA during dependency collection step for project (
use -X to see details)
[INFO]
[INFO] --- apache-rat:0.10:check (rat-check) @ wrangler-core ---
[INFO] 51 implicit excludes (use -debug for more details).
[INFO] Exclude: cov-int/**
[INFO] Exclude: *.md
[INFO] Exclude: **/*.md
[INFO] Exclude: **/*.json
[INFO] Exclude: **/resources/**
[INFO] Exclude: wrangler-demos/**
[INFO] Exclude: **/com/example/**
[INFO] Exclude: /**/icons/**
[INFO] 307 resources included (use -debug for more details)
[INFO] Rat check: Summary of files. Unapproved: 0 unknown: 0 generated: 0 approved: 307 licence.
[INFO]
[INFO] --- antlr4:4.7:antlr4 (default) @ wrangler-core ---
[INFO] No grammars to process
[INFO] ANTLR 4: Processing source directory C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core\src\main\antlr4
[INFO]
[INFO] --- buildnumber:1.0:create (default) @ wrangler-core ---
[INFO] Storing buildNumber: 2025-04-13-19:51:58_aditt at timestamp: 1744554118674
[INFO] Executing: cmd.exe /X /C "git show"
[INFO] Working directory: C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core
[INFO] Storing buildScmBranch: UNKNOWN
[INFO]
```

```
[INFO] Storing buildScmBranch: UNKNOWN
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ wrangler-core ---
[INFO] Copying 12 resources from src\main\resources to target\classes
[INFO] The encoding used to copy filtered properties files have not been set. This means that the same encoding will be used to copy filtered properties files a
s when copying other filtered resources. This might not be what you want! Run your build with --debug to see which files might be affected. Read more at https:/
/maven.apache.org/plugins/maven-resources-plugin/examples/filtering-properties-files.html
[INFO]
[INFO] --- compiler:3.13.0:compile (default-compile) @ wrangler-core ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ wrangler-core ---
[INFO] Copying 5 resources from src\test\resources to target\test-classes
[INFO]
[INFO] --- compiler:3.13.0:testCompile (default-testCompile) @ wrangler-core ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] --- checkstyle:2.17:check (validate) @ wrangler-core ---
[WARNING]  Parameter 'sourceDirectory' is deprecated: instead use {@link #sourceDirectories}
[INFO] Starting audit...
Audit done.
[INFO]
[INFO] --- surefire:2.14.1:test (default-test) @ wrangler-core ---
[INFO] Surefire report directory: C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core\target\surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running io.cdap.wrangler.parser.TimeDurationTest
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.108 sec

Results :

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
```

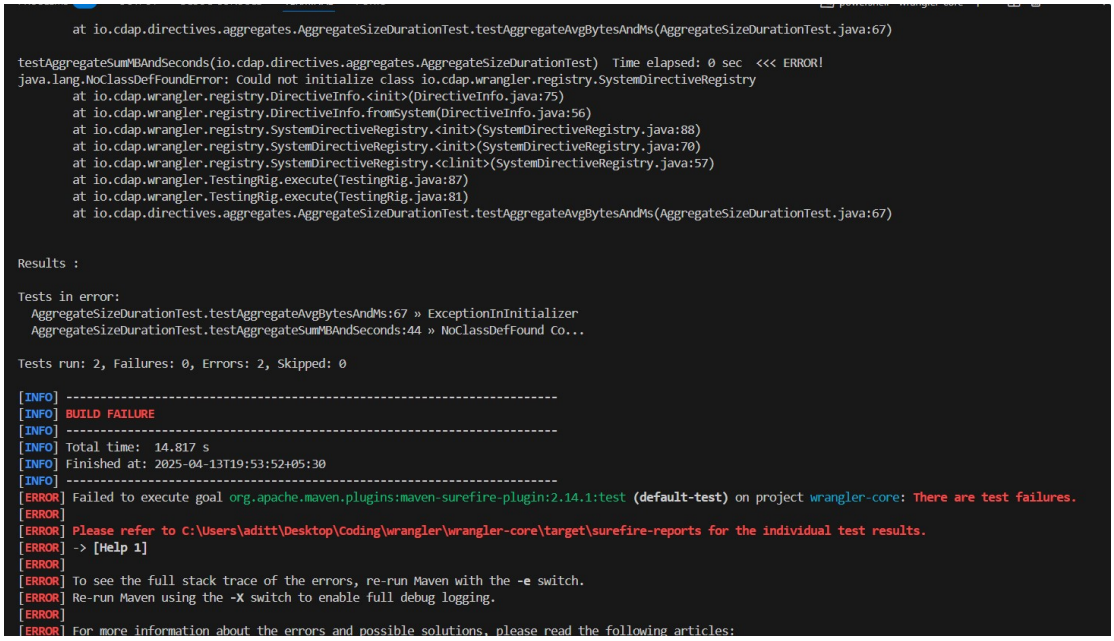
```
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  14.359 s
[INFO] Finished at: 2025-04-13T19:52:07+05:30
[INFO] ------------------------------------------------------------------------
PS C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core>
```

AggregateSizeDurationTest

```
PS C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core> mvn -Dtest=AggregateSizeDurationTest test
[INFO] Scanning for projects...
[INFO]
[INFO] -------------------< io.cdap.wrangler:wrangler-core >-------------------
[INFO] Building Wrangler Core 4.12.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[WARNING] 1 problem was encountered while building the effective model for org.apache.hadoop:hadoop-annotations:jar:2.6.5 during dependency collection step for
project (use -X to see details)
[WARNING] 2 problems were encountered while building the effective model for org.apache.yetus:audience-annotations:jar:0.5.0 during dependency collection step f
or project (use -X to see details)
[WARNING] 1 problem was encountered while building the effective model for org.javassist:javassist:jar:3.18.2-GA during dependency collection step for project (
use -X to see details)
[INFO]
[INFO] --- apache-rat:0.10:check (rat-check) @ wrangler-core ---
[INFO] 51 implicit excludes (use -debug for more details).
[INFO] Exclude: cov-int/**
[INFO] Exclude: *.md
[INFO] Exclude: **/*.md
[INFO] Exclude: **/*.json
[INFO] Exclude: **/resources/**
[INFO] Exclude: wrangler-demos/**
[INFO] Exclude: **/com/example/**
[INFO] Exclude: /**/icons/**
[INFO] 307 resources included (use -debug for more details)
[INFO] Rat check: Summary of files. Unapproved: 0 unknown: 0 generated: 0 approved: 307 licence.
[INFO]
[INFO] --- antlr4:4.7:antlr4 (default) @ wrangler-core ---
[INFO] No grammars to process
[INFO] ANTLR 4: Processing source directory C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core\src\main\antlr4
[INFO]
[INFO] --- buildnumber:1.0:create (default) @ wrangler-core ---
[INFO] Storing buildNumber: 2025-04-13-19:53:43_aditt at timestamp: 1744554223003
[INFO] Executing: cmd.exe /X /C "git show"
[INFO] Working directory: C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core
[INFO] Storing buildScmBranch: UNKNOWN
[INFO]
```

```
[INFO] --- resources:3.3.1:resources (default-resources) @ wrangler-core ---
[INFO] Copying 12 resources from src\main\resources to target\classes
[INFO] The encoding used to copy filtered properties files have not been set. This means that the same encoding will be used to copy filtered properties files a
s when copying other filtered resources. This might not be what you want! Run your build with --debug to see which files might be affected. Read more at https:/
/maven.apache.org/plugins/maven-resources-plugin/examples/filtering-properties-files.html
[INFO]
[INFO] --- compiler:3.13.0:compile (default-compile) @ wrangler-core ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ wrangler-core ---
[INFO] Copying 5 resources from src\test\resources to target\test-classes
[INFO]
[INFO] --- compiler:3.13.0:testCompile (default-testCompile) @ wrangler-core ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] --- checkstyle:2.17:check (validate) @ wrangler-core ---
[WARNING]  Parameter 'sourceDirectory' is deprecated: instead use {@link #sourceDirectories}
[INFO] Starting audit...
Audit done.
[INFO]
[INFO] --- surefire:2.14.1:test (default-test) @ wrangler-core ---
[INFO] Surefire report directory: C:\Users\aditt\Desktop\Coding\wrangler\wrangler-core\target\surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running io.cdap.directives.aggregates.AggregateSizeDurationTest
Tests run: 2, Failures: 0, Errors: 2, Skipped: 0, Time elapsed: 1.331 sec <<< FAILURE!
testAggregateAvgBytesAndMs(io.cdap.directives.aggregates.AggregateSizeDurationTest)  Time elapsed: 1.244 sec  <<< ERROR!
java.lang.ExceptionInInitializerError: null
        at io.cdap.wrangler.registry.DirectiveInfo.<init>(DirectiveInfo.java:75)
        at io.cdap.wrangler.registry.DirectiveInfo.fromSystem(DirectiveInfo.java:56)
        at io.cdap.wrangler.registry.SystemDirectiveRegistry.<init>(SystemDirectiveRegistry.java:88)
        at io.cdap.wrangler.registry.SystemDirectiveRegistry.<init>(SystemDirectiveRegistry.java:70)
        at io.cdap.wrangler.registry.SystemDirectiveRegistry.<clinit>(SystemDirectiveRegistry.java:57)
        at io.cdap.wrangler.TestingRig.execute(TestingRig.java:87)
        at io.cdap.wrangler.TestingRig.execute(TestingRig.java:81)
```



Prompt.txt in github repo