What's new in Java Drivers

Cassandra Driver 4 in a Nutshell

- 4.0 released in 2019, latest release 4.10.0
- Major rewrite from 3.x
- Asynchronous, non-blocking engine
- Execution profiles
- Global timeouts
- Improved load balancing policy
- Improved metrics
- Improved Object Mapper
- Support for Cassandra 4 and DataStax Astra

www.datastax.com/blog/2019/03/introducing-java-driver-4

Upgrading to Cassandra Driver 4

- Maven coordinates changed

Old:

```
<dependency>
  <groupId>com.datastax.cassandra</groupId>
  <artifactId>cassandra-driver-core</artifactId>
  <version>3.10.1</version>
</dependency>
```

New:

```
<dependency>
  <groupId>com.datastax.oss</groupId>
  <artifactId>java-driver-core</artifactId>
  <version>4.8.0</version>
</dependency>
```

- Package names changed
  `com.datastax.driver` -> `com.datastax.oss.driver`

- Having trouble migrating to driver 4? We can help!
  - Follow the Upgrade guide: docs.datastax.com/en/developer/java-driver/latest/upgrade_guide
  - Ask questions at community.datastax.com
  - Driver mailing list: groups.google.com/a/lists.datastax.com/g/java-driver-user

Upgrading to Cassandra Driver 4

- `Cluster + Session = CqlSession`

Old:

```
Cluster cluster = null;
try {
  cluster = Cluster.builder()
    .addContactPoints(...)
    .build();

  Session session = cluster.connect();
  session.execute(...);

} finally {
  if (cluster != null) cluster.close();
}
```

New:

```
try (CqlSession session =
         CqlSession
           .builder()
           .addContactPoint(...)
           .build()) {

      session.execute(...);

}
```

Upgrading to Cassandra Driver 4

- No more Guava!

Old:

```
Futures.addCallback(
  session.executeAsync(...), // Guava future
  new FutureCallback<ResultSet>() {
    public void onSuccess(ResultSet rs) {
      Row row = rs.one();
      process(row);
    }
    public void onFailure(Throwable t) {
      t.printStackTrace();
    }

  });
```

New:

```
session
    .executeAsync(...) // Java 8 future
    .thenApply(rs -> rs.one())
    .whenComplete(
        (row, error) -> {
          if (error == null) {
            process(row);
          } else {
            error.printStackTrace();
          }
        });
```

Upgrading to Cassandra Driver 4

- Immutability by default, except for builders

Old:

```
PreparedStatement ps = ...;
BoundStatement bs = ps.bind();
bs.setInt("k", 42);
```

New:

```
PreparedStatement ps = ...;
BoundStatement bs = ps.bind();
bs = bs.setInt("k", 42); // bs is immutable!!
```

Alternatively, switch to builders (new):

```
BoundStatementBuilder builder =
  ps.boundStatementBuilder();
builder.setInt("k", 42); // OK, mutable
bs = builder.build();
```

Upgrading to Cassandra Driver 4

- New asynchronous pagination API

Old:

```
ResultSetFuture rs = session.executeAsync(...);
ListenableFuture<Void> done =
  Futures.transform(rs, process(1));

AsyncFunction<ResultSet, Void> process(int page) {
  return rs -> {
    // process current page
    int remaining = rs.getAvailableWithoutFetching();
    for (Row row : rs) {
      ...; if (--remaining == 0) break;
    }
    // process next page, if any
    boolean hasMorePages =
      rs.getExecutionInfo().getPagingState() != null;
    return hasMorePages
      ? Futures.transform(
          rs.fetchMoreResults(), process(page + 1))
      : Futures.immediateFuture(null);
  };
}
```

New:

```
CompletionStage<AsyncResultSet> rs =
  session.executeAsync(...);

CompletionStage<Void> done =
  rs.thenCompose(this::process);

CompletionStage<Void> process(AsyncResultSet rs) {

  // process current page
  rs.currentPage().forEach(row -> ...);


  // process next page, if any
  return rs.hasMorePages()
   ? rs.fetchNextPage().thenCompose(this::process)
   : CompletableFuture.completedFuture(null);
}
```

Cassandra Driver 4 Highlights

- New Reactive API

```
// ReactiveResultSet extends Publisher<ReactiveRow>
ReactiveResultSet rs = session.executeReactive("SELECT ...");

// Wrap with Reactor (or RxJava)
Flux.from(rs)
  .doOnNext(System.out::println)
  .blockLast(); // query execution happens here
```

docs.datastax.com/en/developer/java-driver/latest/manual/core/reactive

Cassandra 4 Support

- Multiple ports per node
  - Contact points now must be entered with a port number

- Virtual tables
  - Can be queried like normal tables
  - New methods:
    `KeyspaceMetadata.isVirtual()`
    `TableMetadata.isVirtual()`

What's new in
Spring data Cassandra 3.0

DataStax Developers

Spring Data Cassandra 3.0

- Upgraded to Cassandra driver 4 in Neumann release train (3.0)
- Embedded Objects (`@Embedded(prefix = ...)`)
- `@Value` support for object creation
- Customizable NamingStrategy API

Upgrading to Spring Data Cassandra 3.0

- Your mileage varies depending on level of data access abstraction, meaning:
  - Repository vs. CassandraOperations usage
  - Usage of CqlOperations and async CqlOperations Statement API requires special attention
  - Driver Statement objects are now immutable
- Migration guide shipped with reference documentation
- Lots of internal changes as consequence of driver design

Upgrade Tasks

- Dependency Upgrades (Driver, Spring Data)
- Adapt mapped entities to
  - changed **DATE** type (com.datastax.driver.core.LocalDate -> java.time.LocalDate)
  - Changes in **@CassandraType** (CassandraType.Name enum)
  - **forceQuote** in annotations deprecated now
- Review and adapt configuration

Configuration

- Recommended: Use **Spring Boot**

- Still using XML: cassandra:cluster and cassandra:session now cassandra:session and cassandra:session-factory namespace elements
- Programmatic configuration mostly remains the same (YMMV!)

Execution Profiles

- Associate Statement with settings
- Driver configuration referenced by Statement API objects

```
datastax-java-driver {
  profiles {
    oltp {
      basic.request.timeout = 100 milliseconds
      basic.request.consistency = ONE
    }
    olap {
      basic.request.timeout = 5 seconds
      basic.request.consistency = QUORUM
    }
}
```

Execution Profiles (code)

```
CqlTemplate template = new CqlTemplate();

SimpleStatement simpleStatement = QueryBuilder.….build();
SimpleStatement newStatement = simpleStatement.setExecutionProfileName("olap");
template.queryForList(newStatement);



CqlTemplate olapTemplate = new CqlTemplate();
olapTemplate.setExecutionProfile("olap");



CassandraTemplate cassandraTemplate = …
InsertOptions options = InsertOptions.builder().executionProfile("oltp").build();
cassandraTemplate.insert(person, options);
```

Embedded Objects

```
public class Customer {

     @Id
     UUID id;

     @Embedded.Nullable(prefix = "billing_")
     Address billing;

     @Embedded.Nullable(prefix = "shipping_")
     Address shipping;

     static class Address {
          String street;
          String city;
          String zip;
     }
}
```

Changes in Spring boot 2.3+

Spring Boot 2.3

- Upgraded to Cassandra driver 4 in 2.3

- Configuration must be done either:
  - In application.properties or application.yaml
    - Under `spring.data.cassandra` prefix
  - Or programmatically
- Changes to `spring.data.cassandra` properties:
  - Some properties were renamed
  - Contact points must now contain a port (`host:port`)
  - Local datacenter is now required
    - Except for Astra

Configuration Upgrade Example

Old:

```
spring.data.cassandra:
  cluster-name: prod1
  contact-points: 127.0.0.1
  port: 9042
  keyspace-name: ks1
  read-timeout: 10s
  consistency-level: LOCAL_QUORUM
  fetch-size: 1000
```

New:

```
spring.data.cassandra:
  session-name: prod1
  contact-points: 127.0.0.1:9042
  local-datacenter: dc1
  keyspace-name: ks1
  request:
    timeout: 10s
    consistency: LOCAL_QUORUM
    page-size: 1000
```

Upgrading Application Properties

docs.spring.io/spring-boot/docs/current/reference/html/appendix-application-properties.html

Old property	New property
spring.data.cassandra.cluster-name	spring.data.cassandra.**session-name**
N/A	**spring.data.cassandra.local-datacenter**
spring.data.cassandra.read-timeout spring.data.cassandra.connect-timeout	spring.data.cassandra.**request.timeout** spring.data.cassandra.**connection.connect-timeout** spring.data.cassandra.**connection.init-query-timeout**
spring.data.cassandra.consistency-level spring.data.cassandra.serial-consistency-level	spring.data.cassandra.**request.consistency** spring.data.cassandra.**request.serial-consistency**
spring.data.cassandra.fetch-size	spring.data.cassandra.**request.page-size**
spring.data.cassandra.jmx-enabled	N/A (driver JMX config)

Customizing the Cassandra Session

- Avoid declaring your own `CqlSession` bean!
  - You would lose Spring Boot's auto-configuration support

- Customizers FTW!
  - Callbacks that can be easily implemented by users
  - Spring Boot 2.3+ customizers:
    - `SessionBuilderCustomizer` (High-level)
    - `DriverConfigLoaderBuilderCustomizer` (Low-level, execution profiles)
  - Declare as regular beans

Customizing the Cassandra Session

- Session customization examples

```
@Bean
public CqlSessionBuilderCustomizer sslCustomizer() {
    return builder -> builder.withSslContext(…);
}

@Bean
public CqlSessionBuilderCustomizer credentialsCustomizer() {
    return builder -> builder.withAuthCredentials(…);
}

@Bean
public DriverConfigLoaderBuilderCustomizer oltpProfile() {
  return builder -> builder.startProfile("oltp"). … .endProfile();
}
```

Connecting to Astra

- Typical configuration

```
spring.data.cassandra:
  username: <astra user>
  password: <astra password>
  keyspace-name: <astra keyspace>
  # no contact-points and no local-datacenter for Astra!

datastax.astra:
  secure-connect-bundle: </path/to/secure-connect-bundle.zip>
```

docs.datastax.com/en/developer/java-driver/latest/manual/cloud

Connecting to Astra

- Passing the secure connect bundle: with a customizer bean

```
@Value("${datastax.astra.secure-connect-bundle}")
private String astraBundle;

@Bean
public CqlSessionBuilderCustomizer sessionBuilderCustomizer() {
  return builder ->
    builder.withCloudSecureConnectBundle(Paths.get(astraBundle));
}
```

docs.datastax.com/en/developer/java-driver/latest/manual/cloud

Compatibility Matrix

Your driver version works with...

Cassandra Driver	Spring Data Cassandra	Spring Boot
3.x	2.2 and older	2.2 and older
4.x	3.0 and newer	2.3 and newer

- Can't mix versions from different lines