



Exasol mit Java erweitern

Java User Group Frankfurt, 27. Okt. 2021

Exasol, was ist das?

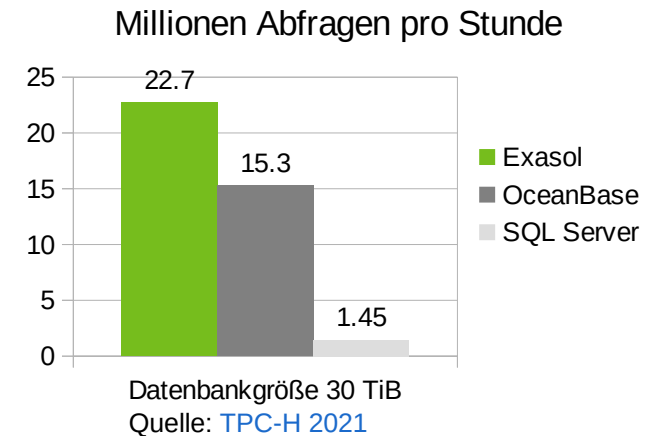
Eine **analytische** SQL-Datenbank.

Läuft **verteilt** auf Cluster-Knoten.

Hält die Daten **im Speicher**.

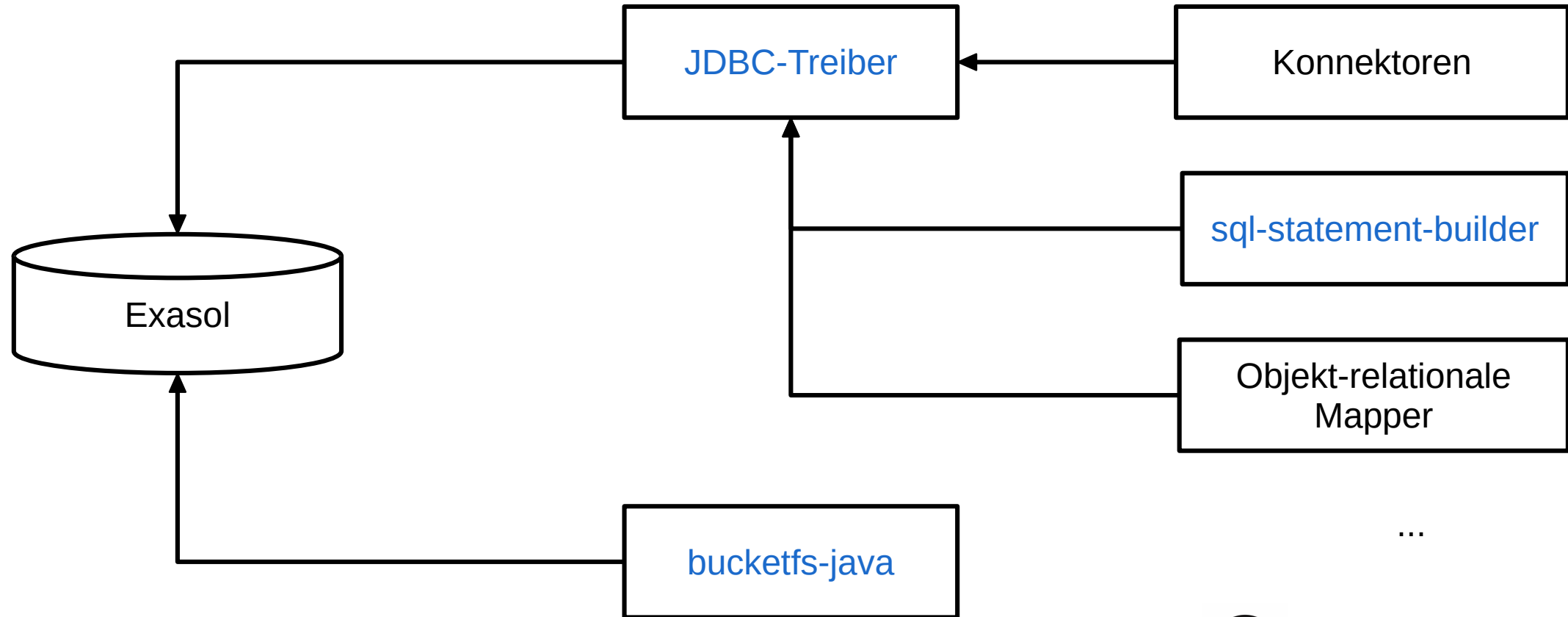
Optimiert sich selbst.

Ist verdammt **schnell**.

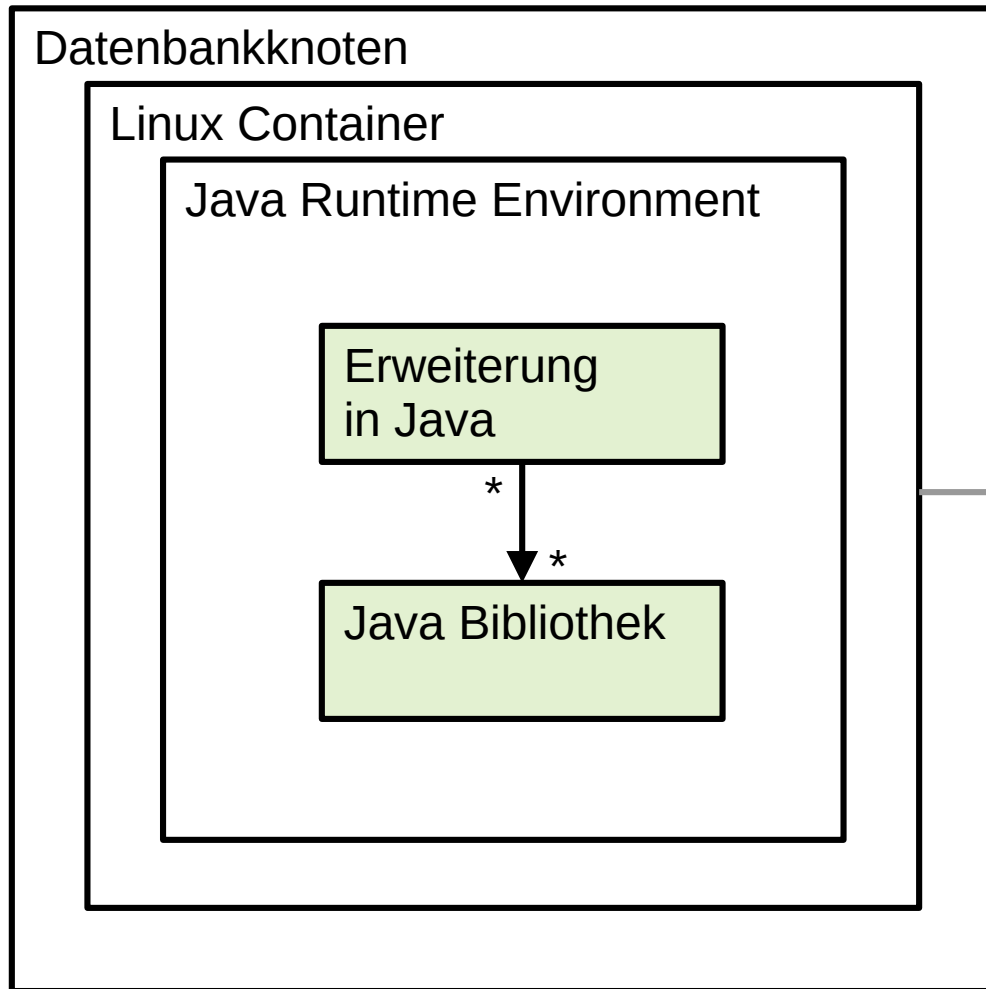


Erweiterungs- möglichkeiten

Datenbank erweitern von außen...



... und von innen



OpenJDK

python™



R logo (CC-BY-SA 4.0)

Skalare Skripten

Hello world!

```
CREATE SCHEMA JAVA_TUTORIAL;
```

```
CREATE JAVA SCALAR SCRIPT JAVA_TUTORIAL.HELLO() RETURNS VARCHAR(20) AS
```

```
    class HELLO {
```

```
        static String run(ExaMetadata metadata, ExaIterator context)
```

```
            throws Exception {
```

```
                return "Hello world!";
```

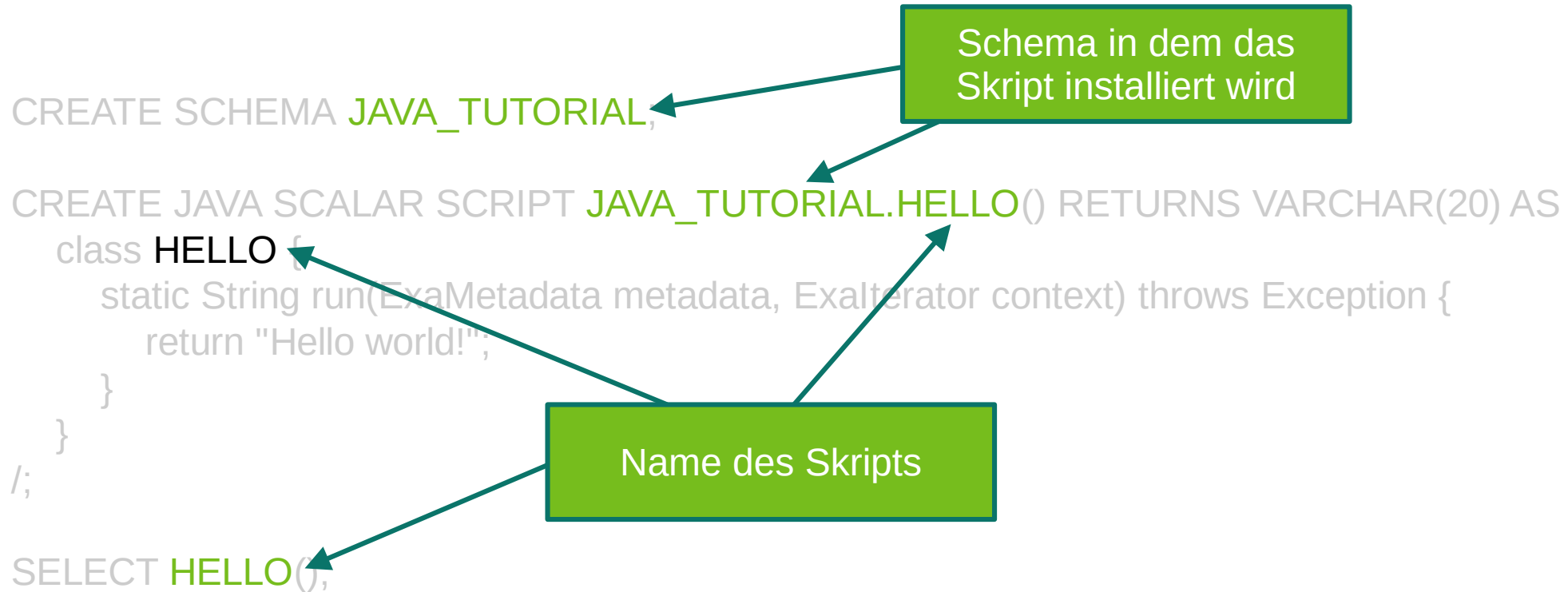
```
        }
```

```
    }
```

```
;/
```

```
SELECT HELLO();
```

Skripte identifizieren

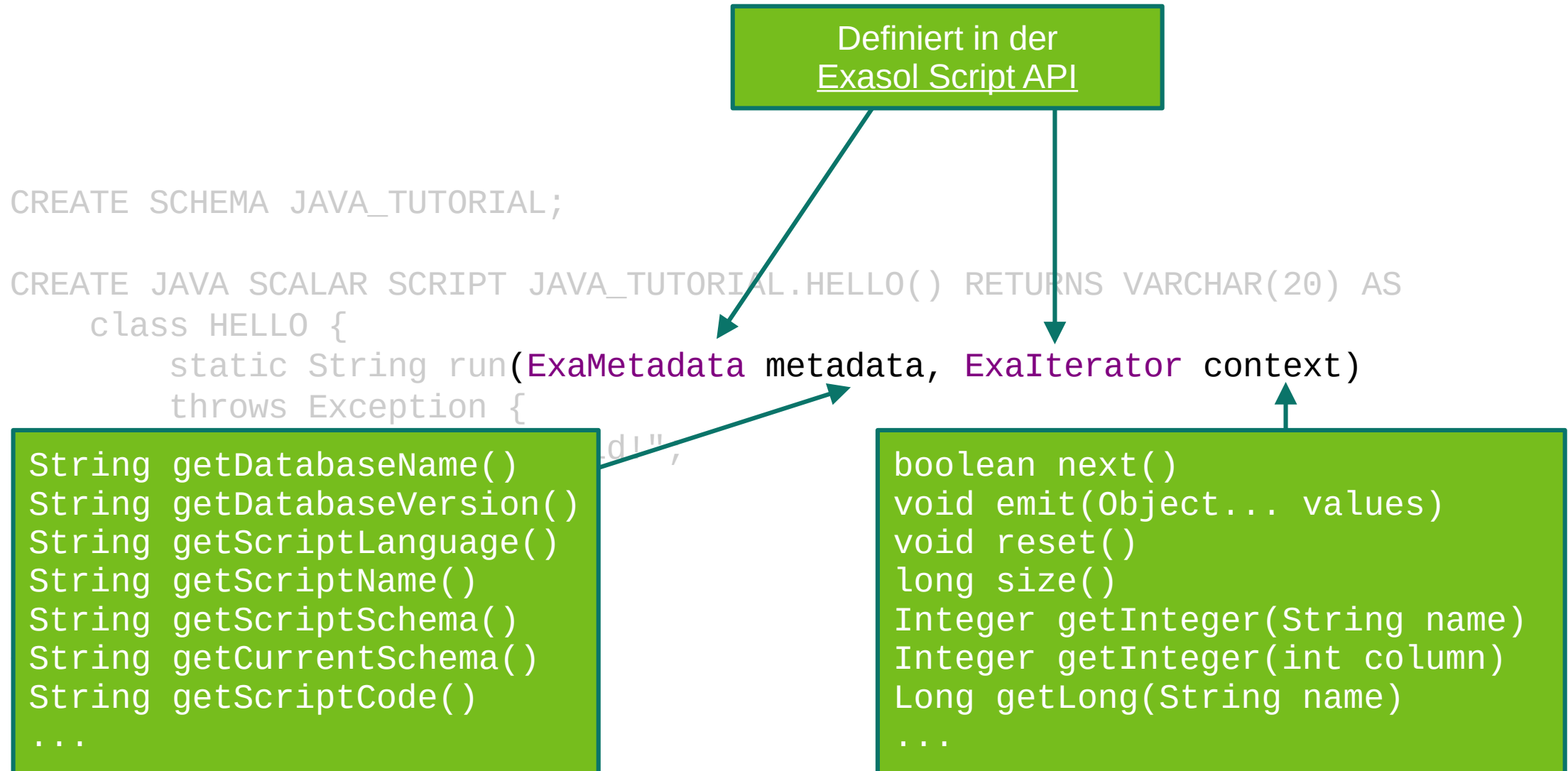


Rückgabewert

```
CREATE SCHEMA JAVA_TUTORIAL;  
  
CREATE JAVA SCALAR SCRIPT JAVA_TUTORIAL.HELLO() RETURNS VARCHAR(20) AS  
    class HELLO {  
        static String run(ExaMetadata metadata, ExaIterator context)  
        throws Exception {  
            return "Hello world!";  
        }  
    }  
/;  
  
SELECT HELLO();
```


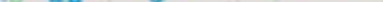


Parameter



```
1 CREATE SCHEMA JAVA_TUTORIAL;
2
3 --/
4 CREATE JAVA SCALAR SCRIPT JAVA_TUTORIAL.HELLO() RETURNS VARCHAR(20) AS
5   class HELLO {
6     static String run(ExaMetadata metadata, ExaIterator context) throws Exception {
7       return "Hello world!";
8     }
9   }
10 /;
11
12 SELECT HELLO();
```

Results 1 x

SELECT HELLO()  Enter a SQL expression to filter results (use Ctrl+Space) 

A screenshot of the Visual Studio Code interface. The main editor area displays the text 'Hello world!'. On the right side, there is a sidebar labeled 'Panels' which contains a tab titled 'Value x'.

Save Cancel Script 1 row(s) fetched - 569ms

Navigation icons: back, forward, search, etc. | Page 200 | 1

ord Rows: 1

Writable	Smart Insert	12:16:281	Sel: 0 0	...
----------	--------------	-----------	------------	-----

Writable	Smart Insert	12:16:281	Sel: 0 0	...
----------	--------------	-----------	------------	-----

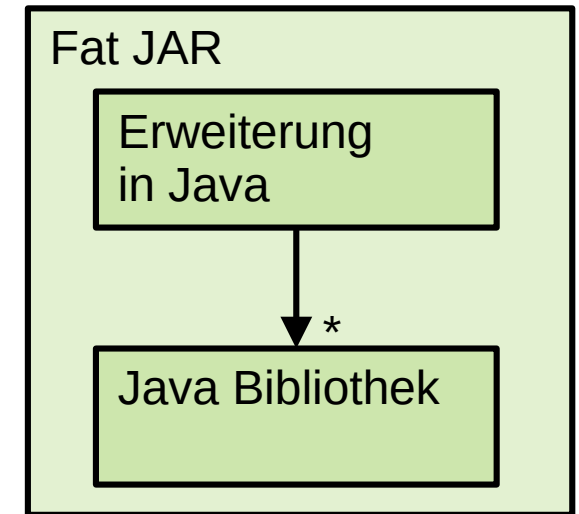
Writable	Smart Insert	12:16:281	Sel: 0 0	...
----------	--------------	-----------	------------	-----

Writable	Smart Insert	12:16:281	Sel: 0 0	...
----------	--------------	-----------	------------	-----

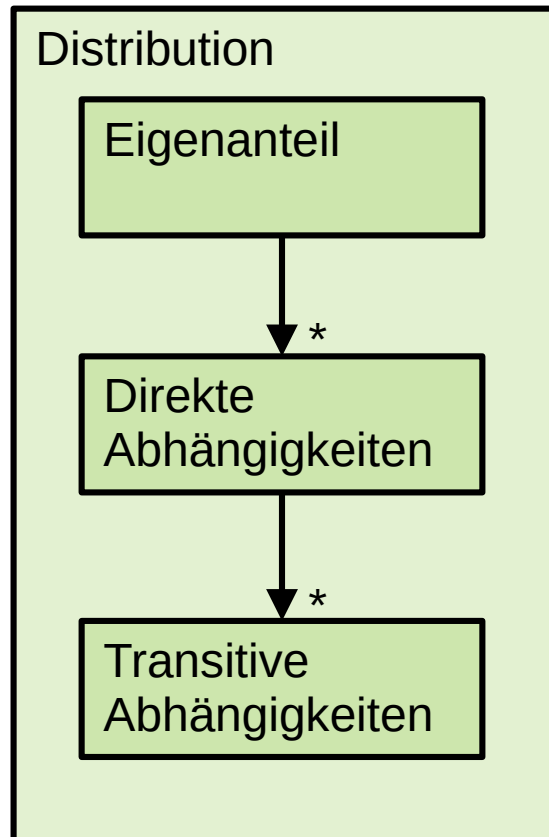
Alles-in-einem-JAR-Erweiterung

1. JAR-Archiv bauen inklusive aller Abhängigkeiten
2. JAR in BucketFS hochladen
3. Scalarfunktion definieren

```
CREATE JAVA SCALAR SCRIPT JAVA_TUTORIAL.MDSTAT(MDTEXT VARCHAR(2000))  
EMITS (WORDS INTEGER, HEADINGS INTEGER, PARAGRAPHS INTEGER) AS  
    %scriptclass com.exasol.javatutorial.markdown.MdStat;  
    %jar /buckets/bfsdefault/default/exasol-java-tutorial.jar;  
/
```



Exkurs: Abhängigkeiten und Verantwortung



Wer Abhängigkeiten mitliefert, ist für sie **verantwortlich**.

1. Distribution bei Sicherheitsupdates sofort aktualisieren!
2. Überprüfen, ob die Abhängigkeit noch gewartet wird
3. Notfalls selbst warten oder ersetzen

Markdown Elemente zählen

```
package com.exasol.javatutorial.markdown;
import com.exasol.ExaIterator;
import com.exasol.ExaMetadata;

public class MdStat {
    public static void run(final ExaMetadata metadata,
                           final ExaIterator context) throws Exception {

        final String markdownText = context.getString("MDTEXT");

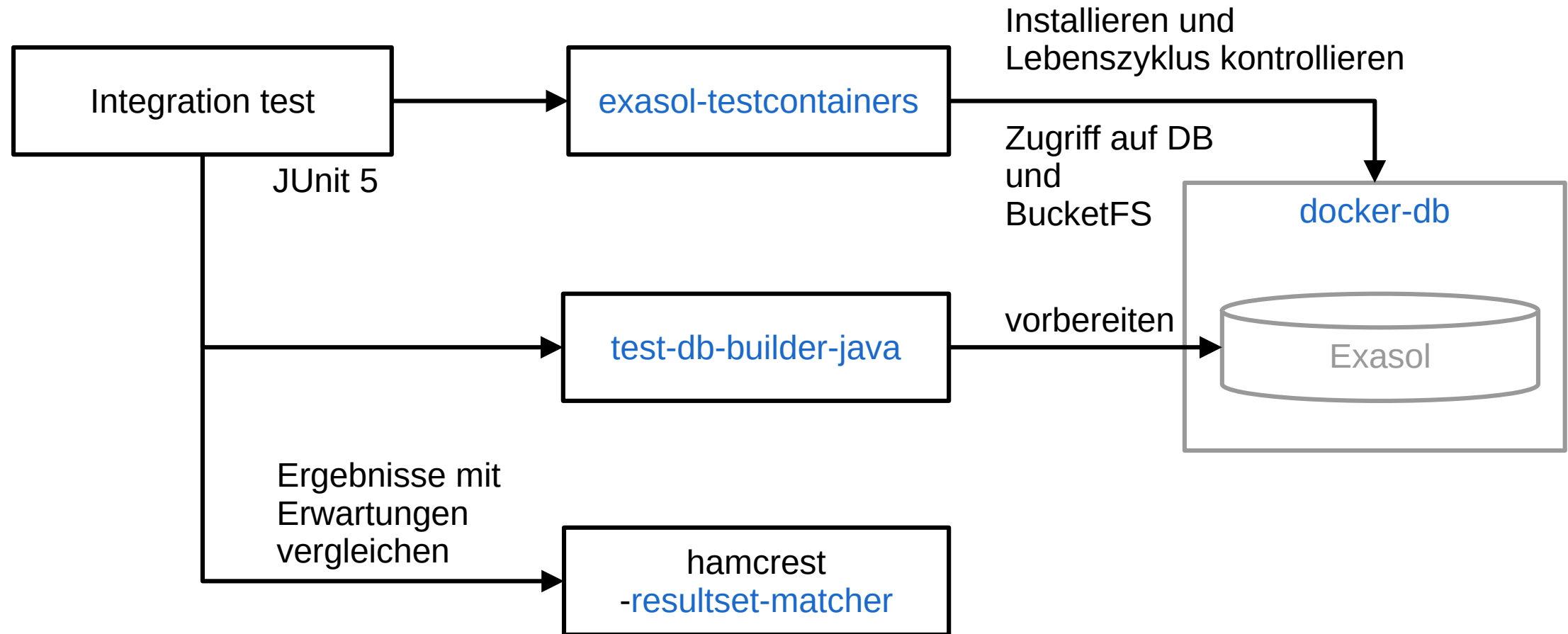
        final MarkdownStatisticsScanner scanner =
            new MarkdownStatisticsScanner();
        final TextStatistics statistics = scanner.scan(markdownText);

        context.emit(statistics.getWords(), statistics.getHeadings(),
                     statistics.getParagraphs());
    }
}
```



Integrationstests

Automatisch besser testen



Exasol-Testcontainer

@Testcontainers

```
class MarkdownStatisticsScalarScriptIT {
```

@Container

```
private static final ExasolContainer<? extends ExasolContainer<?>> EXASOL =  
    new ExasolContainer<>().withReuse(true);
```

```
private static Connection connection;  
private static DatabaseObjectFactory factory;
```

@BeforeAll

```
static void beforeAll() throws SQLException {  
    connection = EXASOL.createConnection();  
    factory = new ExasolObjectFactory(connection);  
}
```

```
...
```

```
}
```

Installiert und steuert
Exasol-Instanz

Wiederverwendung
spart Testzeit



exasol-java-tutorial

Testdaten erstellen

```

@Testcontainers
class MarkdownStatisticsScalar
    @Container
    private static final ExasolContainer container =
        new ExasolContainer();

    private static Connection connection;
    private static DatabaseObjectFactory factory;

    @BeforeAll
    static void beforeAll() throws SQLException {
        connection = EXASOL.createConnection();
        factory = new ExasolObjectFactory(connection);
    }

    ...
}

```

DatabaseObjectFactory

```

User createUser(final String name);
User createUser(final String name, final String password);
User createLoginUser(final String name);
User createLoginUser(final String name, final String password);
void executeSqlFile(final Path... sqlFiles);
Schema createSchema(final String name);

```



test-db-builder-java

exasol-java-tutorial - exasol-java-tutorial/src/test/java/com/exasol/javatutorial/markdown/MarkdownStatisticsScalarScriptIT.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Project Outline Type Hi

src/test/java

com.exasol.javatutorial.markdown

MarkdownStatisticsScalarScriptIT.java

MarkdownStatisticsScannerTest.java

JUnit

Finished after 3.407 seconds

Runs: 1/1 Errors: 0 Failures: 0

MarkdownStatisticsScalarScriptIT [Runner: JUnit4]

testGetMarkdownStatistics() (0.944 s)

Failure Trace

MarkdownStatisticsScalarScriptIT.java

```
45 @Test
46 void testGetMarkdownStatistics() {
47     final Schema schema = factory.createSchema("SCHEMA_FOR_MARKDOWN_STATISTICS");
48     schema.createTable("TEXTS", "TEXT", "VARCHAR(2000)");
49     .insert("# A Headline\n\nAnd some text with _emphasis_.");
50     installMdStatsScript();
51     final ResultSet result = execute("SELECT MDSTAT(TEXT) FROM TEXTS");
52     assertThat(result, table().row(7, 1, 1).matches(NO_JAVA_TYPE_CHECK));
53 }
```

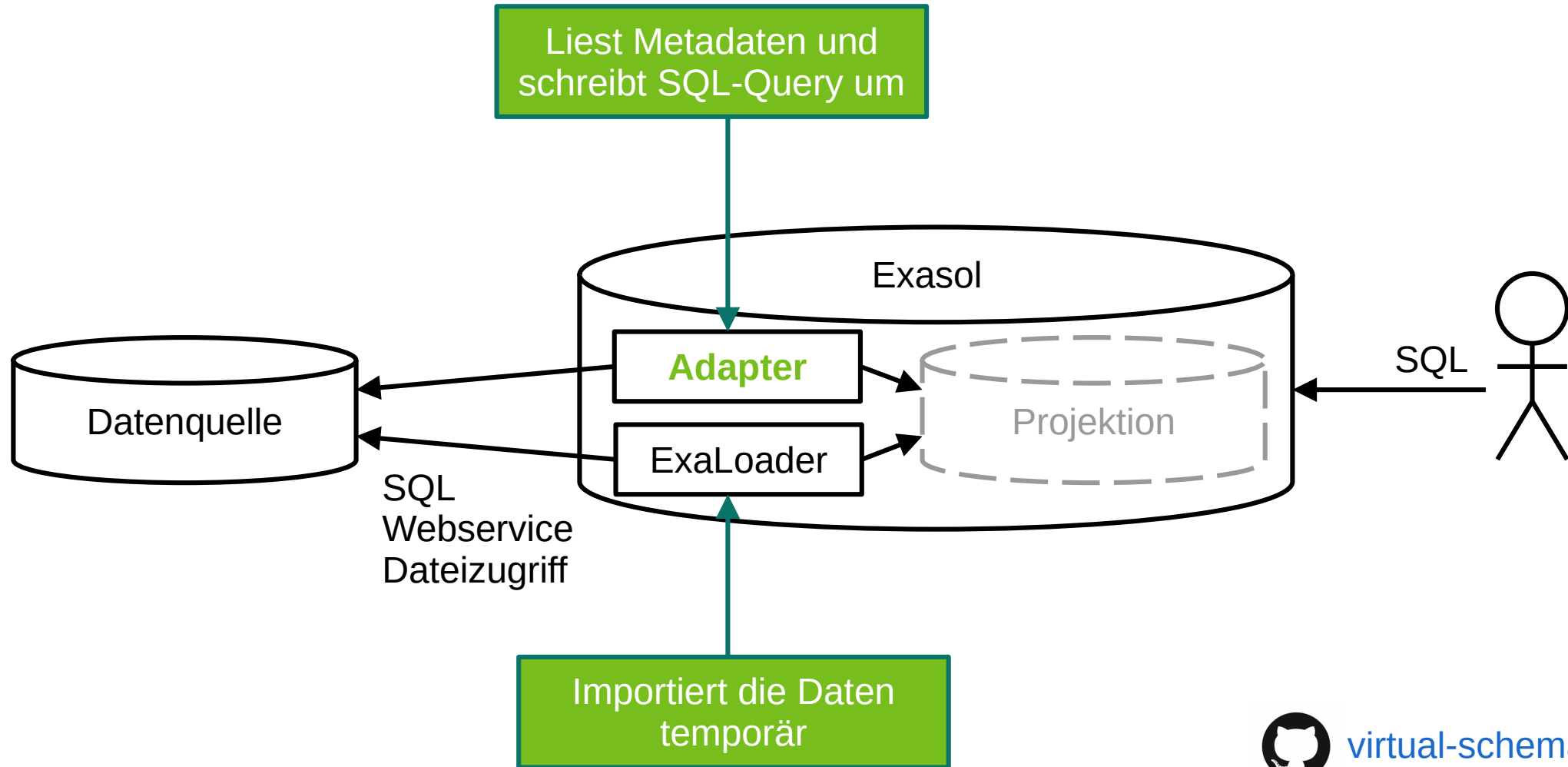
Proble Search Console Termina Git Rep Coverag SonarLi SonarLi SonarLi SonarLi History Call Hie

<terminated> MarkdownStatisticsScalarScriptIT [JUnit] /usr/lib/jvm/java-1.11.0-openjdk-amd64/bin/java (Oct 26, 2021, 12:27:15 PM – 12:27:19 PM)

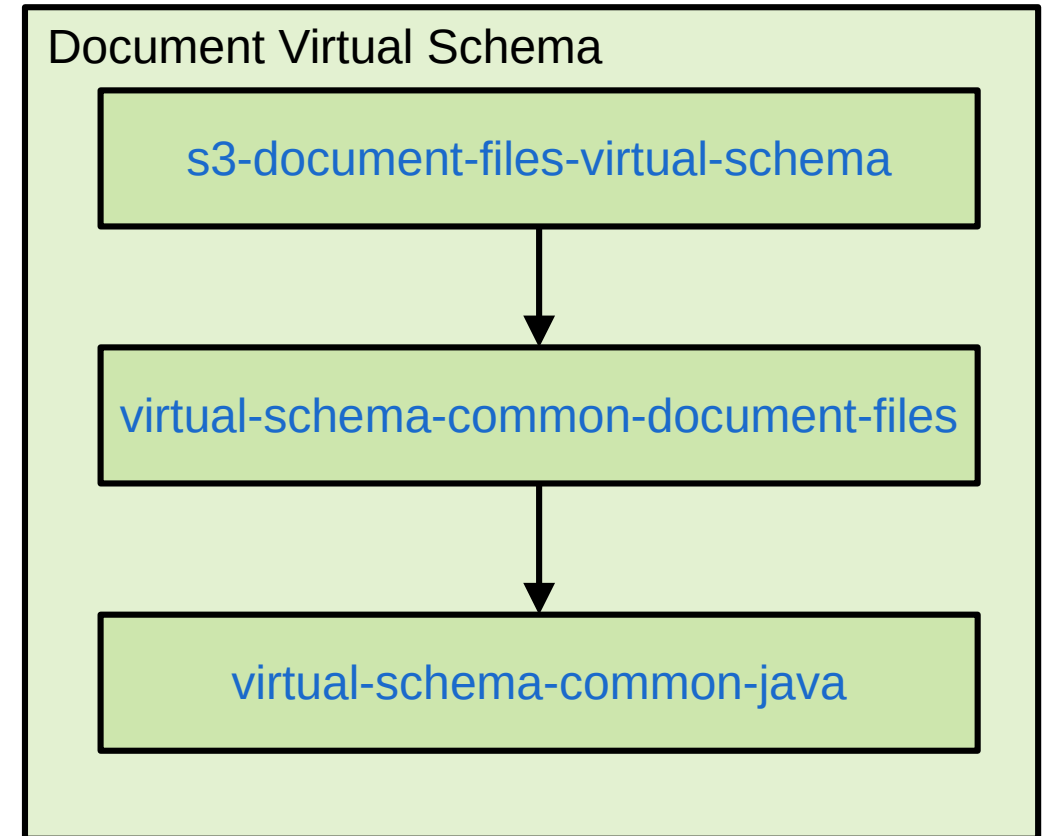
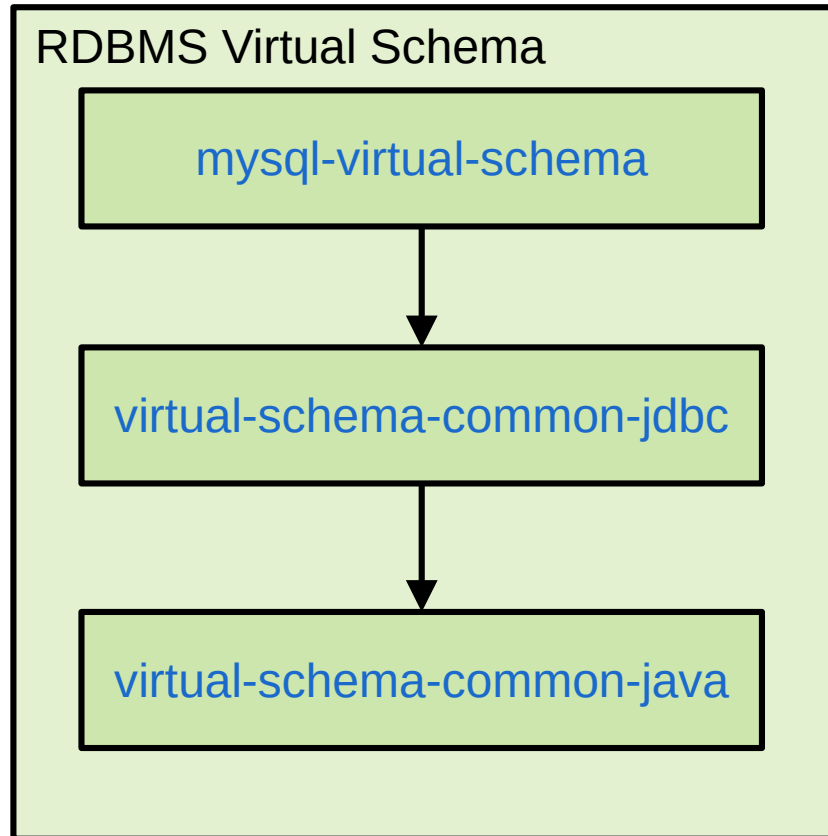
2021-10-26 12:27:16.919 [FINE] Exposing ports: [8563, 2580, 443]
2021-10-26 12:27:16.934 [INFO] Creating container for image: exasol/docker-db:7.1.1
2021-10-26 12:27:16.962 [INFO] Reusing container with ID: 13c954c644d88eb899718e1ac223f9176e717b8e1d366f5f3fc209937f8
2021-10-26 12:27:16.963 [INFO] Container exasol/docker-db:7.1.1 is starting: 13c954c644d88eb899718e1ac223f9176e717b8e
2021-10-26 12:27:16.969 [FINE] Reading container state from cache file "/tmp/exasol_testcontainers/13c954c644d88eb8997
2021-10-26 12:27:16.980 [FINE] Reading cluster configuration from "/exa/etc/EXAConf"
2021-10-26 12:27:17.860 [FINE] BucketFS marked running in container status cache. Skipping startup monitoring.
2021-10-26 12:27:17.861 [FINE] UDF Containter marked running in container status cache. Skipping startup monitoring.
2021-10-26 12:27:17.861 [INFO] Exasol container started after waiting for the following services to become available:
2021-10-26 12:27:17.861 [INFO] Container exasol/docker-db:7.1.1 started in PT0.941856S
2021-10-26 12:27:17.862 [FINEST] Log rotation workaround unnecessary, since container is being reused.
2021-10-26 12:27:17.863 [INFO] Purging database for a clean setup
2021-10-26 12:27:18.009 [FINE] DROP SCHEMA IF EXISTS "SCHEMA_FOR_MARKDOWN_STATISTICS" CASCADE
2021-10-26 12:27:18.060 [FINE] Writing container status to cache file "/tmp/exasol_testcontainers/13c954c644d88eb8997
2021-10-26 12:27:18.364 [FINE] No previous uploads to 'exasol-java-tutorial.jar' recorded in upload history. No upload
2021-10-26 12:27:18.371 [FINE] Uploading file 'target/exasol-java-tutorial.jar' to bucket 'bfsdefault/default' at 'ht
2021-10-26 12:27:18.570 [FINE] Successfully uploaded file 'target/exasol-java-tutorial.jar' to 'http://localhost:4915
2021-10-26 12:27:18.572 [FINE] Recorded upload to 'exasol-java-tutorial.jar' at 2021-10-26T10:27:18.571164Z in upload
2021-10-26 12:27:18.574 [FINE] Created log detector that scans for "exasol-java-tutorial.jar.*linked" in "/exa/logs/c
2021-10-26 12:27:18.637 [FINE] Found matching log entry 2021-10-26T12:27:18 (after 2021-10-26T12:27:18): [I 211026 12
2021-10-26 12:27:18.638 [FINE] Recorded upload to 'exasol-java-tutorial.jar' at 2021-10-26T10:27:18.638089Z in upload

Virtuelle Schemata

Virtuelle Schemata



Bibliotheken für virtuelle Schemata



YouTube: [Hands on the S3 Virtual Schema](#)

Virtual Schema API in Java

```
public interface VirtualSchemaAdapter {  
    public CreateVirtualSchemaResponse createVirtualSchema(  
        final ExaMetadata metadata,  
        final CreateVirtualSchemaRequest request  
    ) throws AdapterException;  
  
    public DropVirtualSchemaResponse dropVirtualSchema(...) ... ;  
  
    public RefreshResponse refresh(...) ...;  
  
    public SetPropertyResponse setProperties(...) ...;  
  
    public GetCapabilitiesResponse getCapabilities(...) ...;  
  
    public PushDownResponse pushdown(...) ...;  
}
```



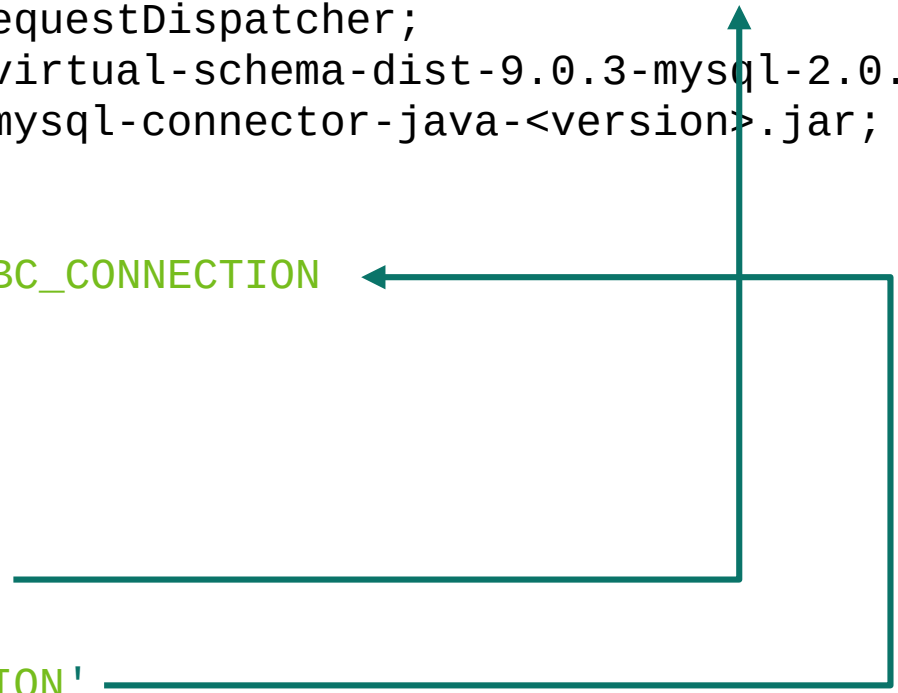
[virtual-schema-common-java](https://github.com/exasol/virtual-schema-common-java)

Virtual Schema anlegen

```
CREATE OR REPLACE JAVA ADAPTER SCRIPT VS_SCRIPTS.ADAPTER_SCRIPT_MYSQL AS
    %scriptclass com.exasol.adapter.RequestDispatcher;
    %jar /buckets/bfsdefault/default/virtual-schema-dist-9.0.3-mysql-2.0.1.jar;
    %jar /buckets/bfsdefault/default/mysql-connector-java-<version>.jar;
/;

CREATE OR REPLACE CONNECTION MYSQL_JDBC_CONNECTION
TO 'jdbc:mysql://<host>:<port>/'
USER '<user>'
IDENTIFIED BY '<password>';

CREATE VIRTUAL SCHEMA VS_MYSQL
USING VS_SCRIPTS.ADAPTER_SCRIPT_MYSQL
WITH
CONNECTION_NAME = 'MYSQL_JDBC_CONNECTION'
CATALOG_NAME = '<database name>';
```



Zusammenfassung

Exasol und Java

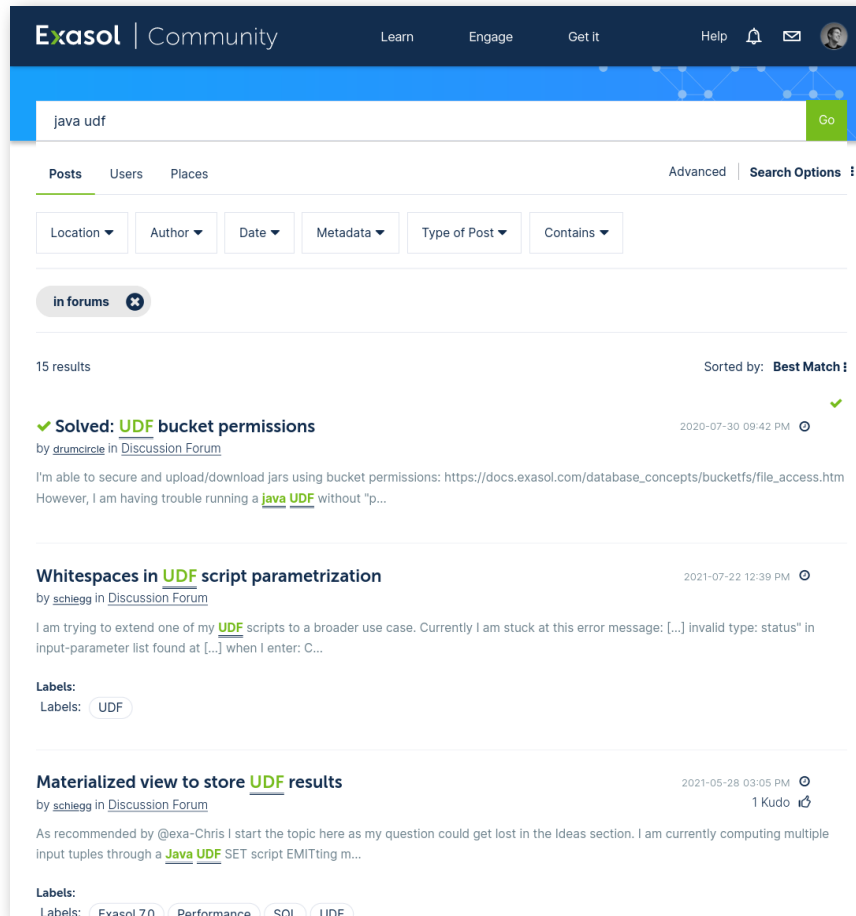
Exasol läßt sich über **UDF** mit Java erweitern.

Virtuelle Schemata erlauben Datenquellen von Exasol aus zu durchsuchen.

Wir stellen viele **Java Bibliotheken** für Implementierung und Test zur Verfügung.

Ein **Java Tutorial** gibt's auf **GitHub**.

Bis bald! In der Exasol Community.



Knowledge base

- Alles über Aktualisierungen
- Eigene Artikel erstellen
- Mit den Autoren kommunizieren

Diskussionsforen

- Fragen stellen, Antworten finden
- Neue Ideen aufnehmen
- Wissen teilen, Expertise ausbauen

User groups & Treffen

- Projekte mit Exasol kennenlernen
- Lokale Verbindungen knüpfen
- Exasol Nutzende und Team treffen

Ideenportal

- Wünsche und Ideen einbringen
- Helfen, Lücken zu finden
- Kommentieren und abstimmen

Fragen?
Antworten!