

Übung Meteo Service Spring Boot

Aufgabe Meteo Daten

Meteo Daten sollen über die Datenbank als Temperaturen vom Typ Double gespeichert werden. Diese Daten werden über den Meteo REST Endpoint zur Verfügung gestellt via die folgende URL `http://localhost:8080/meteo/rest/data` Der Meteo REST Endpoint gibt die MeteoDaten gemäss dem folgenden JSON Listing

zurück: `[{"t":4.5}, {"t":6.2}, {"t":10.3}, {"t":14.2}, {"t":16.1}, {"t":13.3}, {"t":9.8}, {"t":7.2}]` Das Datenbank Schema soll die

MeteoDaten wie folgt ablegen: `create table meteodata (
 id bigint not null`

`auto_increment,
 temperature double precision,
 primary key (id)
)` Sie können das Schema über die JPA Klasse und dem Property `spring.jpa.hibernate.ddl-auto=create-drop` erstellen (default bisher so gemacht in den Beispielen) oder via Datei `schema.sql`. Im Falle `schema.sql` würde dies wie folgt aussehen: `drop table meteodata;
create table meteodata (
 id bigint not null`

`auto_increment,
 temperature double precision,
 primary key (id)
)` Die Daten können initial mit der Datei `data.sql` beim Startup eingelesen werden: `INSERT INTO meteodata (temperature) VALUES (4.5);
INSERT INTO meteodata (temperature) VALUES (6.2);
INSERT INTO meteodata (temperature) VALUES (10.3);
INSERT INTO meteodata (temperature) VALUES (14.2);
INSERT INTO meteodata (temperature) VALUES (16.1);
INSERT INTO meteodata (temperature) VALUES (13.3);
INSERT INTO meteodata (temperature) VALUES (9.8);
INSERT INTO meteodata (temperature) VALUES (7.2);` Damit die `data.sql` Datei im `resources` Folder gelesen wird setzen wir das folgende Property in der Datei `application.properties` `spring.sql.init.mode=always`

Die Daten können initial mit der Datei `data.sql` beim Startup eingelesen werden: `INSERT INTO meteodata (temperature) VALUES (4.5);
INSERT INTO meteodata (temperature) VALUES (6.2);
INSERT INTO meteodata (temperature) VALUES (10.3);
INSERT INTO meteodata (temperature) VALUES (14.2);
INSERT INTO meteodata (temperature) VALUES (16.1);
INSERT INTO meteodata (temperature) VALUES (13.3);
INSERT INTO meteodata (temperature) VALUES (9.8);
INSERT INTO meteodata (temperature) VALUES (7.2);` Damit die `data.sql` Datei im `resources` Folder gelesen wird setzen wir das folgende Property in der Datei `application.properties` `spring.sql.init.mode=always`

Die Daten können initial mit der Datei `data.sql` beim Startup eingelesen werden: `INSERT INTO meteodata (temperature) VALUES (4.5);
INSERT INTO meteodata (temperature) VALUES (6.2);
INSERT INTO meteodata (temperature) VALUES (10.3);
INSERT INTO meteodata (temperature) VALUES (14.2);
INSERT INTO meteodata (temperature) VALUES (16.1);
INSERT INTO meteodata (temperature) VALUES (13.3);
INSERT INTO meteodata (temperature) VALUES (9.8);
INSERT INTO meteodata (temperature) VALUES (7.2);` Damit die `data.sql` Datei im `resources` Folder gelesen wird setzen wir das folgende Property in der Datei `application.properties` `spring.sql.init.mode=always`

Die Daten können initial mit der Datei `data.sql` beim Startup eingelesen werden: `INSERT INTO meteodata (temperature) VALUES (4.5);
INSERT INTO meteodata (temperature) VALUES (6.2);
INSERT INTO meteodata (temperature) VALUES (10.3);
INSERT INTO meteodata (temperature) VALUES (14.2);
INSERT INTO meteodata (temperature) VALUES (16.1);
INSERT INTO meteodata (temperature) VALUES (13.3);
INSERT INTO meteodata (temperature) VALUES (9.8);
INSERT INTO meteodata (temperature) VALUES (7.2);` Damit die `data.sql` Datei im `resources` Folder gelesen wird setzen wir das folgende Property in der Datei `application.properties` `spring.sql.init.mode=always`

Die Daten können initial mit der Datei `data.sql` beim Startup eingelesen werden: `INSERT INTO meteodata (temperature) VALUES (4.5);
INSERT INTO meteodata (temperature) VALUES (6.2);
INSERT INTO meteodata (temperature) VALUES (10.3);
INSERT INTO meteodata (temperature) VALUES (14.2);
INSERT INTO meteodata (temperature) VALUES (16.1);
INSERT INTO meteodata (temperature) VALUES (13.3);
INSERT INTO meteodata (temperature) VALUES (9.8);
INSERT INTO meteodata (temperature) VALUES (7.2);` Damit die `data.sql` Datei im `resources` Folder gelesen wird setzen wir das folgende Property in der Datei `application.properties` `spring.sql.init.mode=always`

Die Daten können initial mit der Datei `data.sql` beim Startup eingelesen werden: `INSERT INTO meteodata (temperature) VALUES (4.5);
INSERT INTO meteodata (temperature) VALUES (6.2);
INSERT INTO meteodata (temperature) VALUES (10.3);
INSERT INTO meteodata (temperature) VALUES (14.2);
INSERT INTO meteodata (temperature) VALUES (16.1);
INSERT INTO meteodata (temperature) VALUES (13.3);
INSERT INTO meteodata (temperature) VALUES (9.8);
INSERT INTO meteodata (temperature) VALUES (7.2);` Damit die `data.sql` Datei im `resources` Folder gelesen wird setzen wir das folgende Property in der Datei `application.properties` `spring.sql.init.mode=always`

Aufgabe Meteo Metrics

Über den Meteo REST Endpoint und der URL `http://localhost:8080/meteo/rest/metrics` sollen die Maxima, Minima und der Mittelwert via JSON wie folgt geliefert werden: `{"min":4.5,"max":16.1,"med":10.200000000000001}`

Die Daten können initial mit der Datei `data.sql` beim Startup eingelesen werden: `INSERT INTO meteodata (temperature) VALUES (4.5);
INSERT INTO meteodata (temperature) VALUES (6.2);
INSERT INTO meteodata (temperature) VALUES (10.3);
INSERT INTO meteodata (temperature) VALUES (14.2);
INSERT INTO meteodata (temperature) VALUES (16.1);
INSERT INTO meteodata (temperature) VALUES (13.3);
INSERT INTO meteodata (temperature) VALUES (9.8);
INSERT INTO meteodata (temperature) VALUES (7.2);` Damit die `data.sql` Datei im `resources` Folder gelesen wird setzen wir das folgende Property in der Datei `application.properties` `spring.sql.init.mode=always`

Lösung

Eine mögliche Lösung finden Sie hier

Kontakt

Simtech AG
Finkenweg 23
3110 Münsingen
Schweiz

Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz. Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsausschluss. Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

<https://www.simtech-ag.ch/key>