Live results streaming

Description

You need to create:

- football matches end result generator (*match-end-result-generator)*
- service for sinking results to relational database (*match-end-result-sinking-service*)

match-end-result-generator needs to be developed in python, with a help of python generators.
It can be started through terminal with command like syntax.
Schema of *match-end-result-event* is following:
{matchId : uuidv4,
matchName: string,
endResult: string}

Example would be *{"matchId": "8657ad85-5b62-4892-bbbe-2c56c93d525f", "matchName": "Manchester United – Zrinjski", "endResult": "1 : 2"}*

Match name should be randomly generated from list of football club names. 10 names is enough.
End result should also be generated randomly with max number of goals per team be 7.

Generator should run endlessly and should generate *match-end-result-event* every second.
After every event generation, generator should send *match-end-result-event* on *match-end-result-sinking-service.*

match-end-result-sinking-service needs to accept messages and insert them to database (MySql or Postgress). It should be developed in Java as a Spring Boot application.

Communication between services should be HTTP, *match-end-result-generator* should use HTTP POST method for sending *match-end-result-event* to *match-end-result-sinking-service.*

Given that the db scheme is (MySql types, adjust if using Postgress)

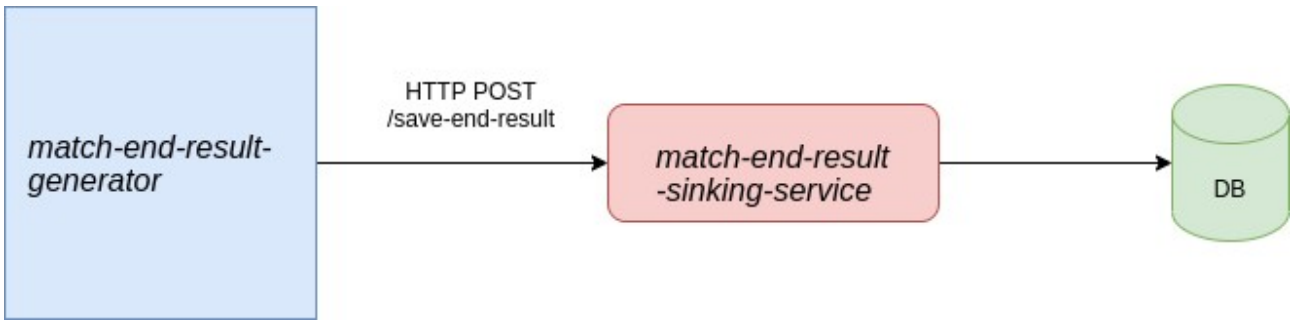
```
create table match_results(
                                    id int not null auto_increment
primary key,
                                    match_id varchar(36),
                                    match_name varchar(128),
                                    end_result varchar(128)
);
```

write sql query that will find a home team with most goals scored.

You should spin up database with help of docker container using *docker-compose.* Docker compose file needs to be in code repository.

Bonus points

In match-end-result-sinking-service save *match-end-result-event* in Java thread safe memory structure like ConcurrentHashMap, spin up a separate thread using java's ExecutorService that will check for *match-end-result-event* in memory structure and insert to database if any exist. After event is inserted in database remove saved event form memory structure.


match-end-result-
generator
HTTP POST
/save-end-result
match-end-result
-sinking-service
DB