CS209A Project Report

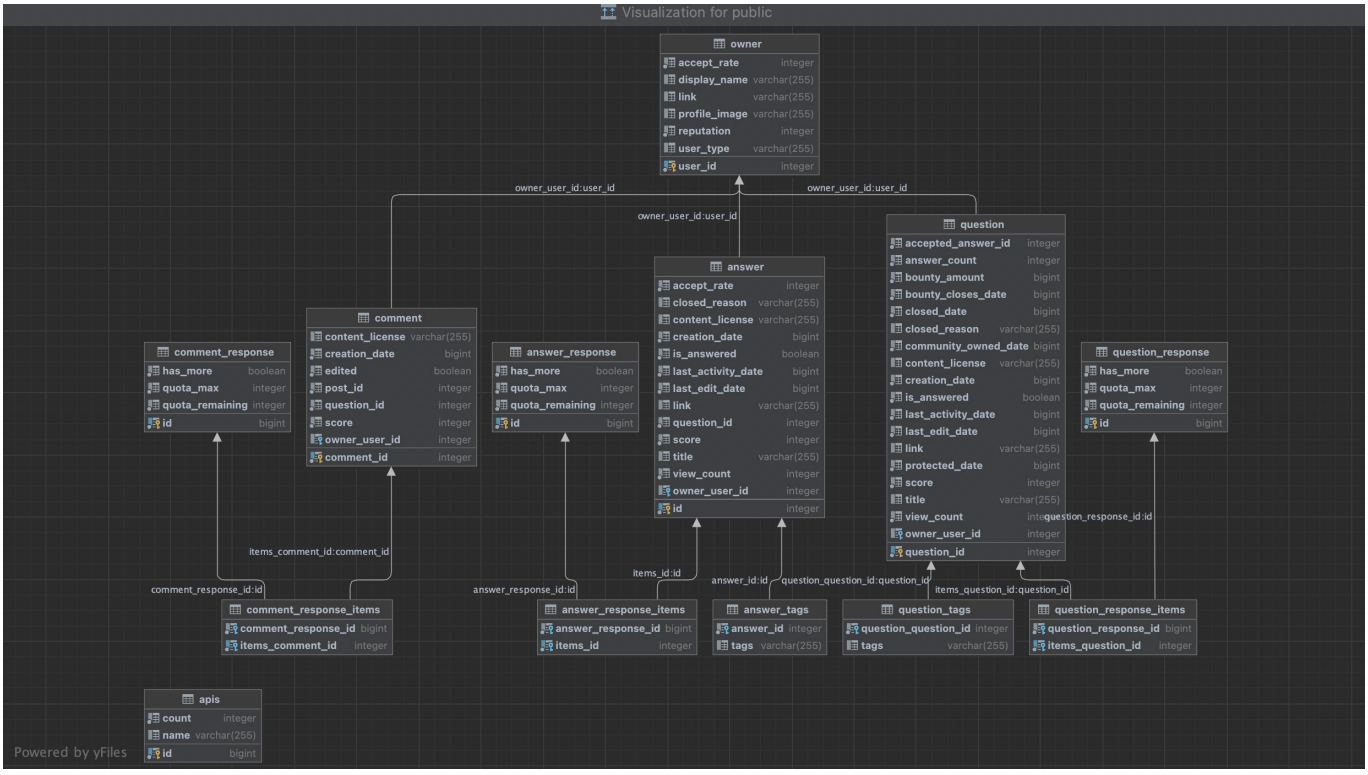
1. *architecture design:*

The architecture of this project mainly consists of two parts, with the front-end to accept data, show the visualization result and the back_end to process the data and respond to the request of the front-end.

1.1. *technology stack*

- front end: React Axios echarts
- back end: SpringBoot JPA
- database: postgreSQL

1.2. *Design of database*



Thread Count：549

1.3. ***Front End***

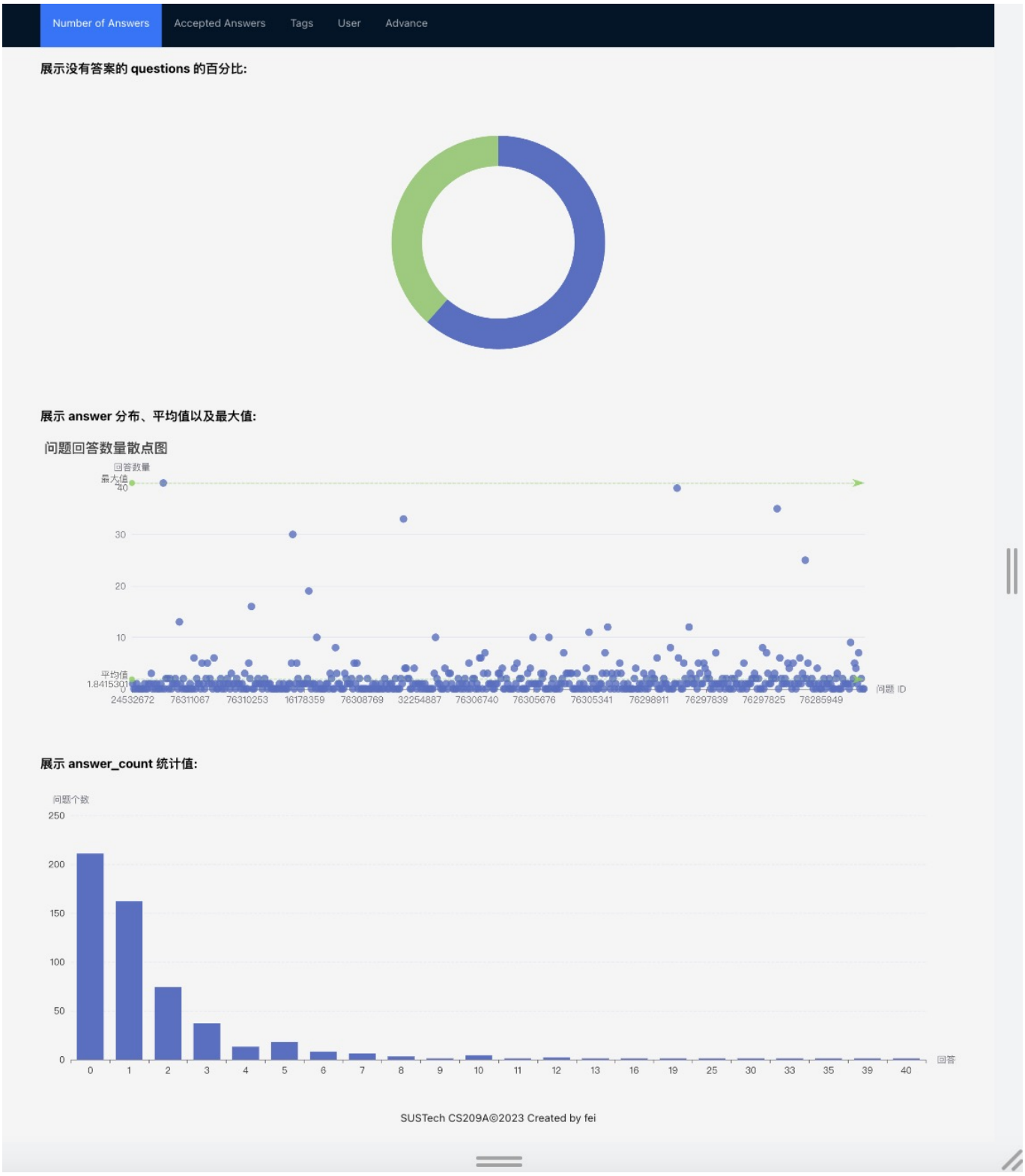
The front-end and back-end interaction was implemented using ***axios***, a popular JavaScript library for making HTTP requests.

Axios provided a convenient and efficient way to send requests from the front-end to the back-end API endpoints and handle the responses.

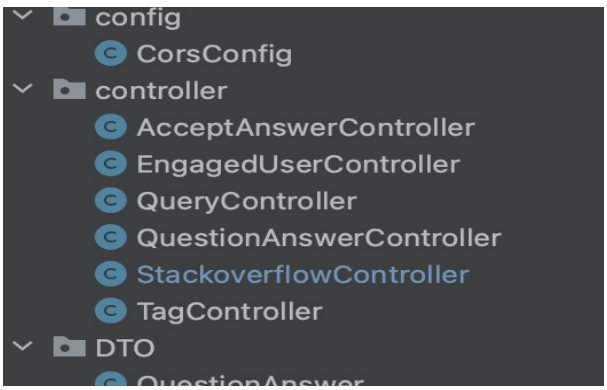
After the front-end sends a request to the endpoint, it gets the data and draws a suitable chart using the ***echarts*** library.

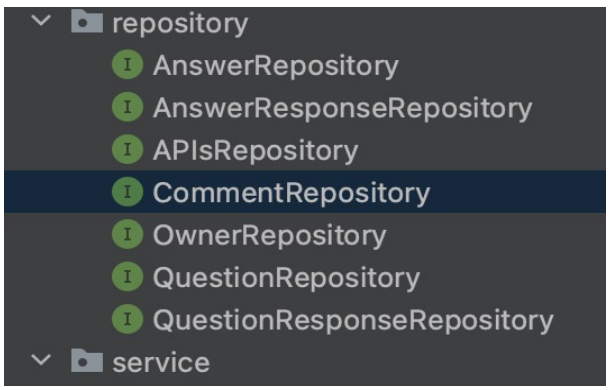
1.3.1. Exhibition:

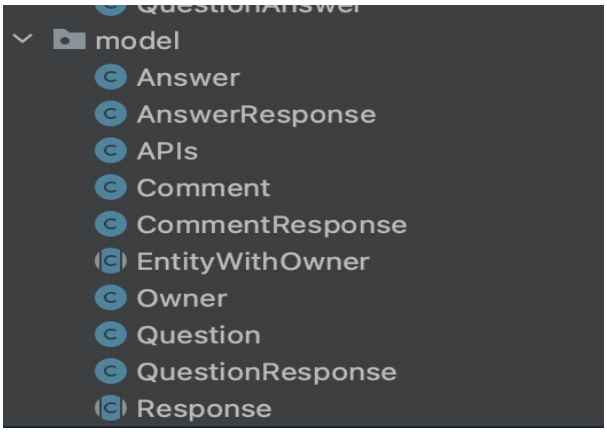
The top navigation bar allows you to select different categories of statistics to view.

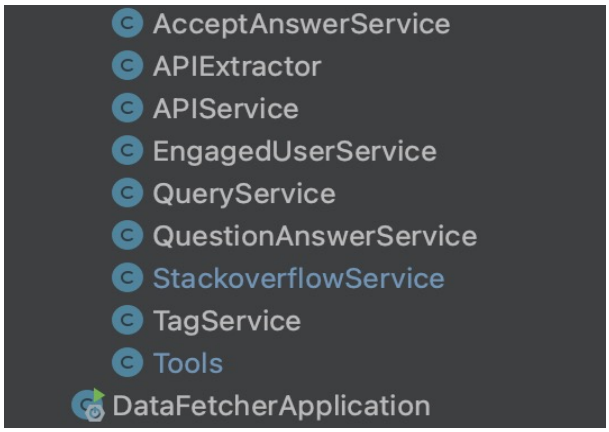

Number of Answers
Accepted Answers
Tags
User
Advance
展示没有答案的 questions 的百分比:
展示 answer 分布、平均值以及最大值:
问题回答数量散点图
回答数量
最大值
平均值
1.8415301
问题 ID
24532672
76311067
76310253
16178359
76308769
32254887
76306740
76305676
76305341
76298911
76297839
76297825
76285949
展示 answer_count 统计值:
问题个数
回答
SUSTech CS209A©2023 Created by fei

1.4. *Back End*


config
CorsConfig
controller
AcceptAnswerController
EngagedUserController
QueryController
QuestionAnswerController
StackoverflowController
TagController
DTO


repository
AnswerRepository
AnswerResponseRepository
APIsRepository
CommentRepository
OwnerRepository
QuestionRepository
QuestionResponseRepository
service





- ***CorsConfig***:Solve the cross-domain problem of front and back-end communication
- ***StackoverflowService***:Data Crawling
- ***AcceptAnswerController***:Provide API related to ***Accepted Answers***
- ***QuestionAnswerContrller***： Provide API related to ***Number of Answers***
- ***TagContrller***： Provide API related to ***Tags***
- ***EngagedUserContrller***： Provide API related to ***Users***

1.5. *Data Crawling*

StackoverflowService is a service class for getting data from the Stack Overflow API. It contains the following main methods:

Apache HttpClient: The CloseableHttpClient and HttpClients classes from the Apache HttpClient library are used to create an HTTP client for making requests to the Stack Overflow API.

Data Source：

official Stack Overflow REST API documentation

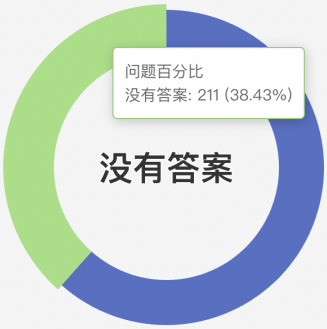
The basic idea of the crawler is as follows:

A StackoverflowService class is defined to encapsulate the methods and logic for crawling data. A RestTemplate is used to initiate HTTP requests and the Stack Overflow API is called to get the data. The GET request is mainly used and the corresponding parameters are passed to specify the type of data to be obtained, sorting method, tags and other information. Based on the return result of the API, the JSON data is parsed using ObjectMapper and converted to Java objects. The parsed data is saved to the corresponding database table through the methods provided by the JpaRepository interface. For multi-page data, the data is paged using a loop traversal and saved to the database. Page crawling is achieved by controlling the parameters of page number and amount of data per page. Process the returned data if needed, such as extracting the APIs mentioned in the question and counting them. Use the Apache HttpClient library to create and manage HTTP clients for HTTP requests.

1.5.1. *Result*

What percentage of questions don't have any answer?

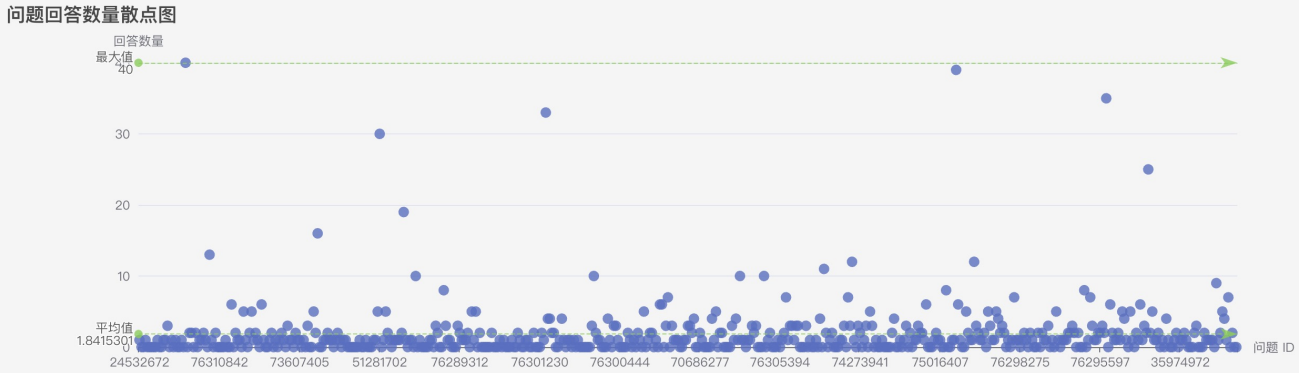
展示没有答案的 questions 的百分比:



- 61.57% questions have answer.

What is the average and maximum number of answers?

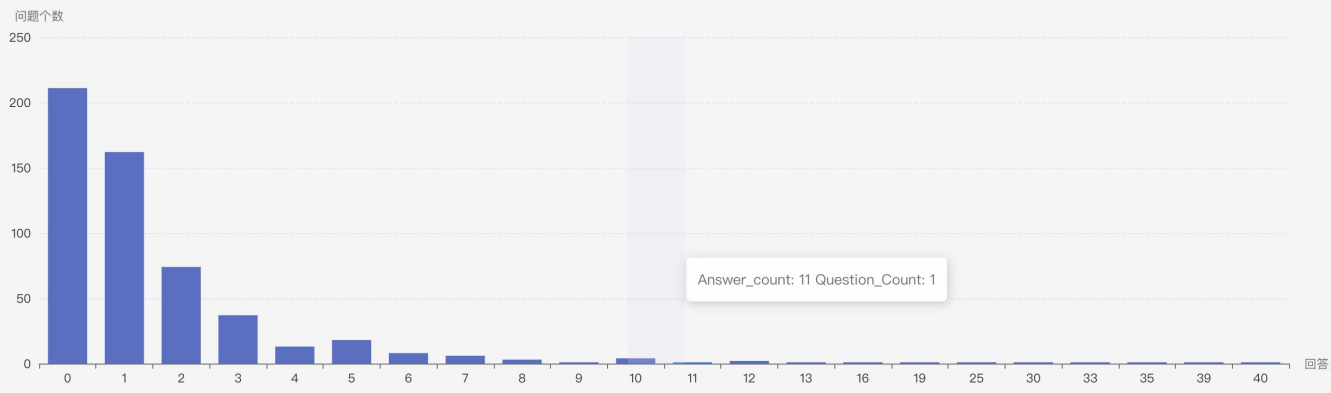
展示 answer 分布、平均值以及最大值:



- average:1.8415301
- maximum:40

What is the distribution of the number of answers?

展示 answer_count 统计值:



- Most of the data had only 5 and less answers. Overall, the number of questions decreases as the number of answers increases.

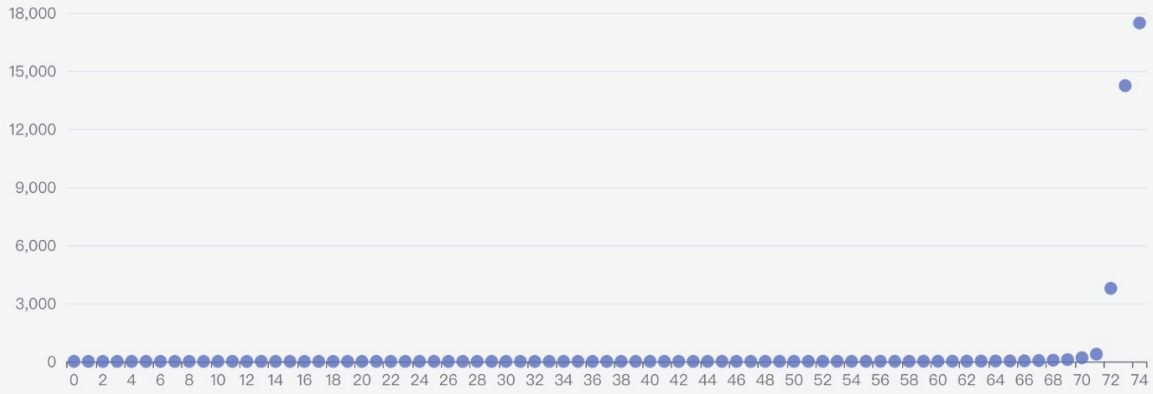
What percentage of questions have accepted answers (one question could only have one accepted answer)?

展示有 accepted answer 的问题的百分比:

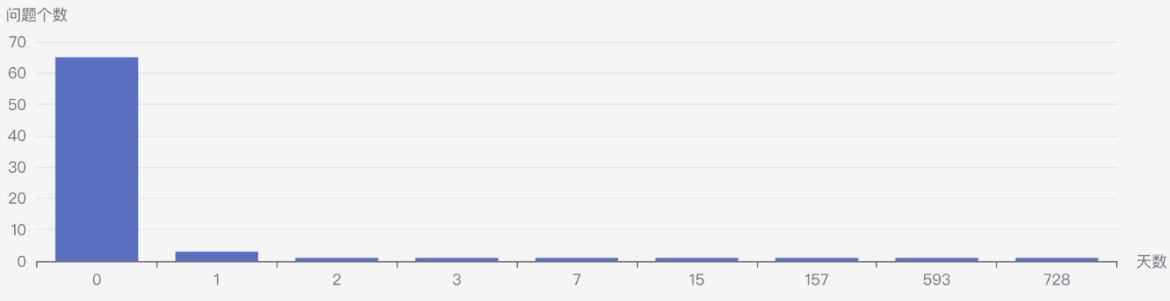
- 20.77% questions have answer.

What is the distribution of question resolution time (i.e., the duration between the question posting time and the posting time of the accepted answer)?

展示问题从提出到解决 (answer accepted time – question post time) 的时间间隔分布(单位:h):



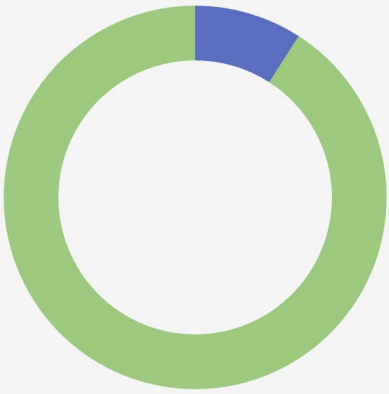
展示问题从提出到解决 (answer accepted time – question post time) 的时间间隔分布(单位:天):



- most question can be solved within one day.

What percentage of questions have non-accepted answers (i.e., answers that are not marked as accepted) that have received more upvotes than the accepted answers?

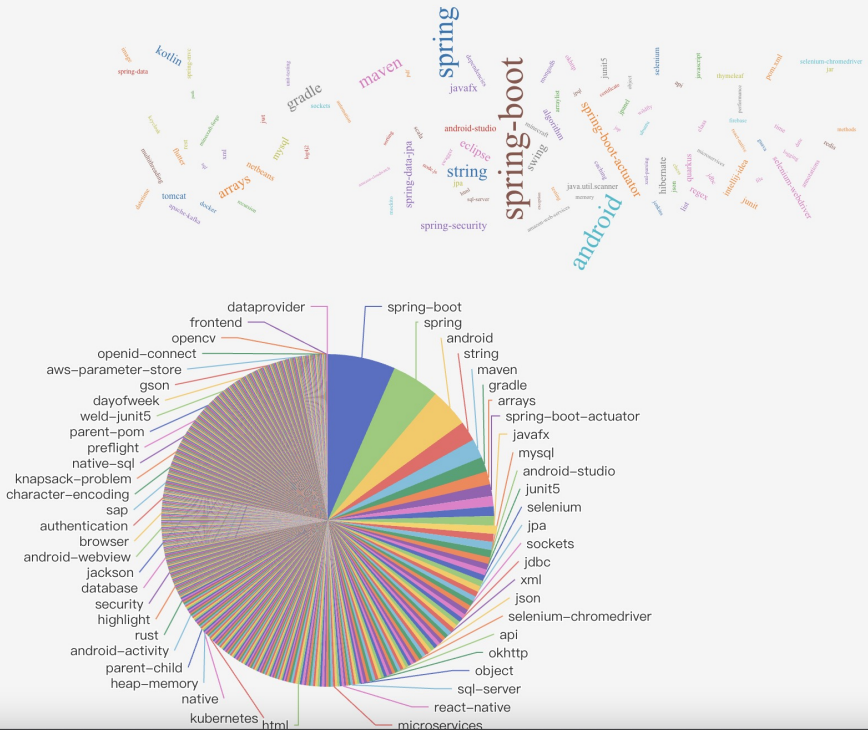
展示含有 non-accepted answer 的 upvote 数高于 accepted answer 的问题的百分比:



- 9.09%
- Consistent with intuitive expectations, answers with more votes are more likely to be adopted

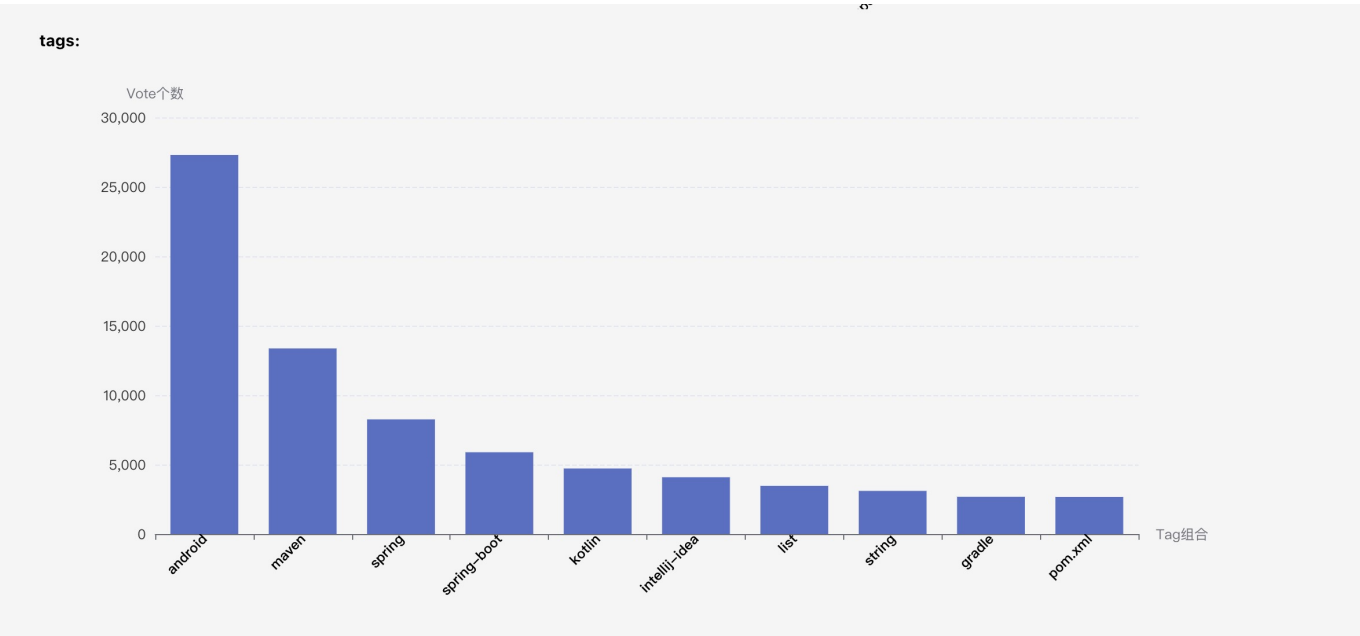
Which tags frequently appear together with the java tag?

展示哪些 tags 经常和 Java tag 一起出现:



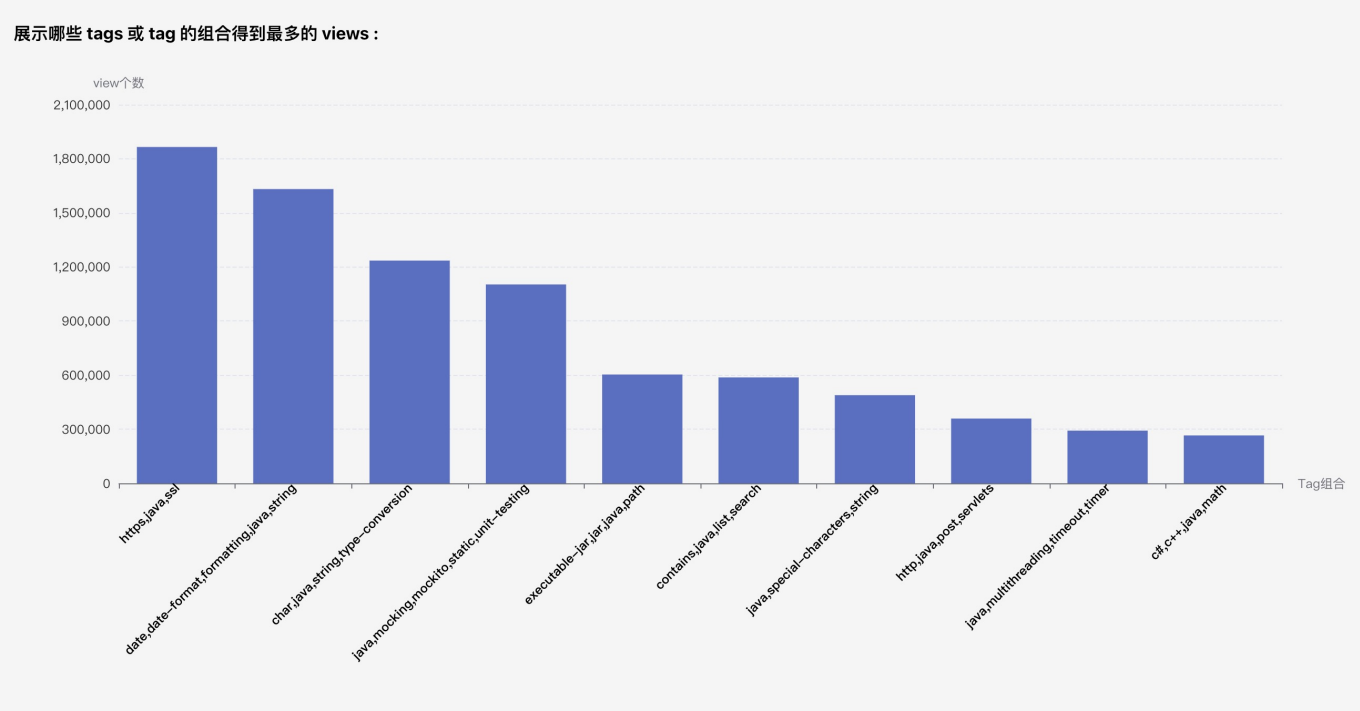
- Spring Boot
- spring
- andorid
- It looks like the Springboot framework is very popular at the moment

Which tags or tag combinations receive the most upvotes?



- tag combinations : executable-jar,jar,java,path

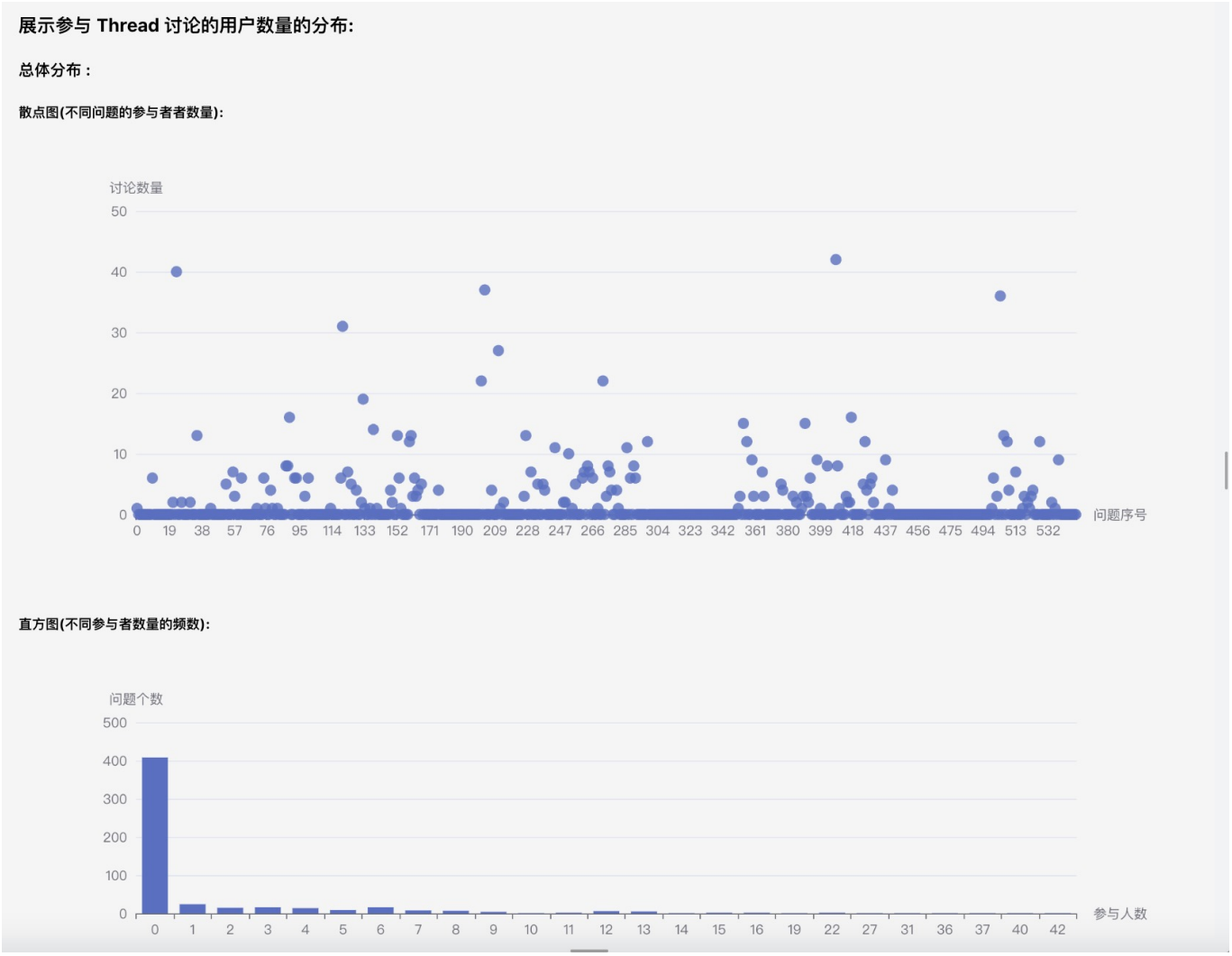
Which tags or tag combinations receive the most views?



- tag combinations : arrylist,collections,duplicates,java,list
- Probably because the list is really used very often

Many users could participate in a thread discussion. What is the distribution of such participation (i.e., the number of distinct users who post the question, answers, or comments in a thread)?

merge:

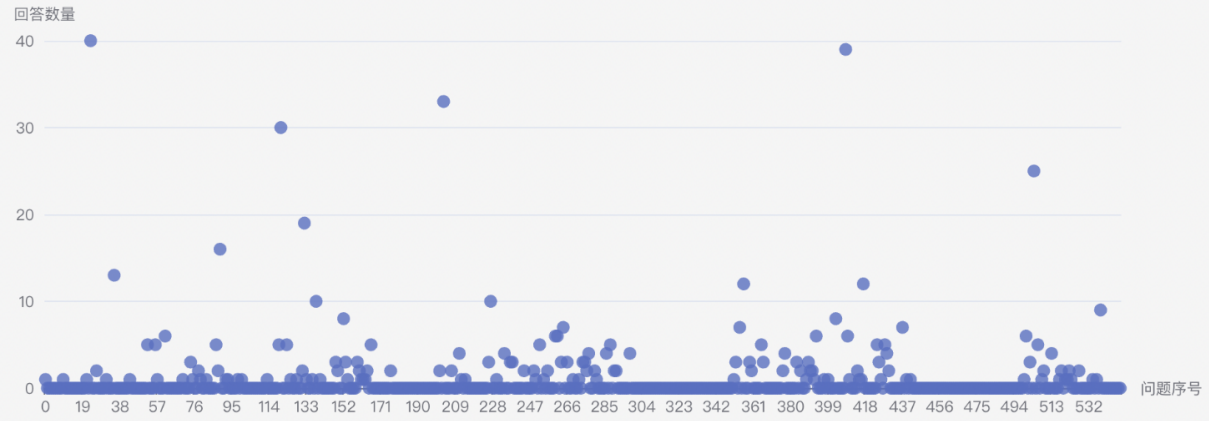


- There are many issues that no one is involved in discussing. Most of the questions with people participating in the discussion had less than 10 participants.

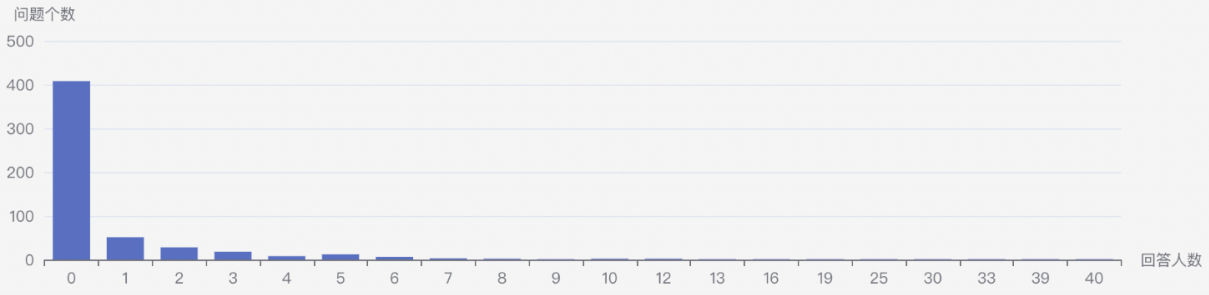
answer:

回答者分布：

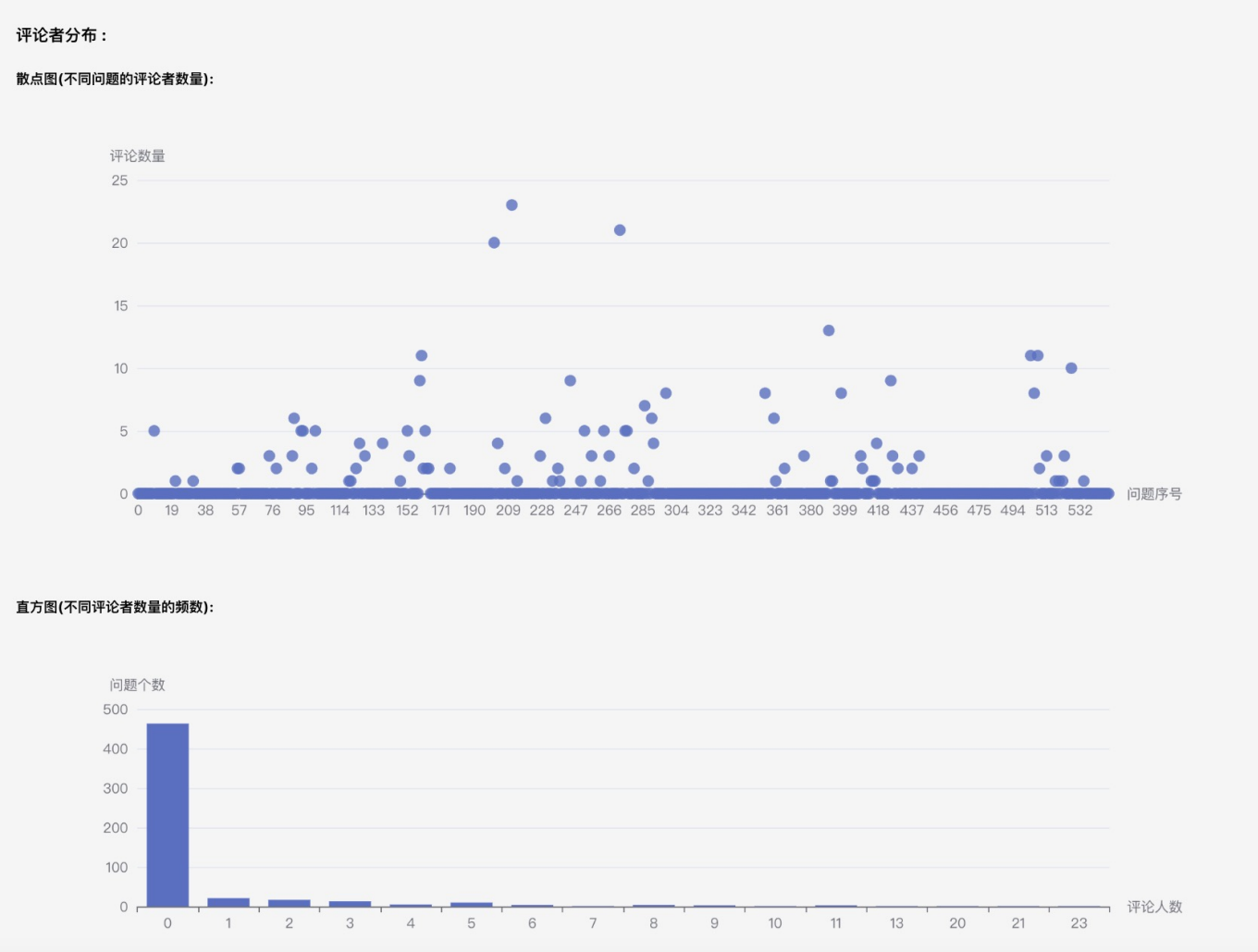
散点图(不同问题的回答者数量)：



直方图(不同回答者数量的频数)：



comment:

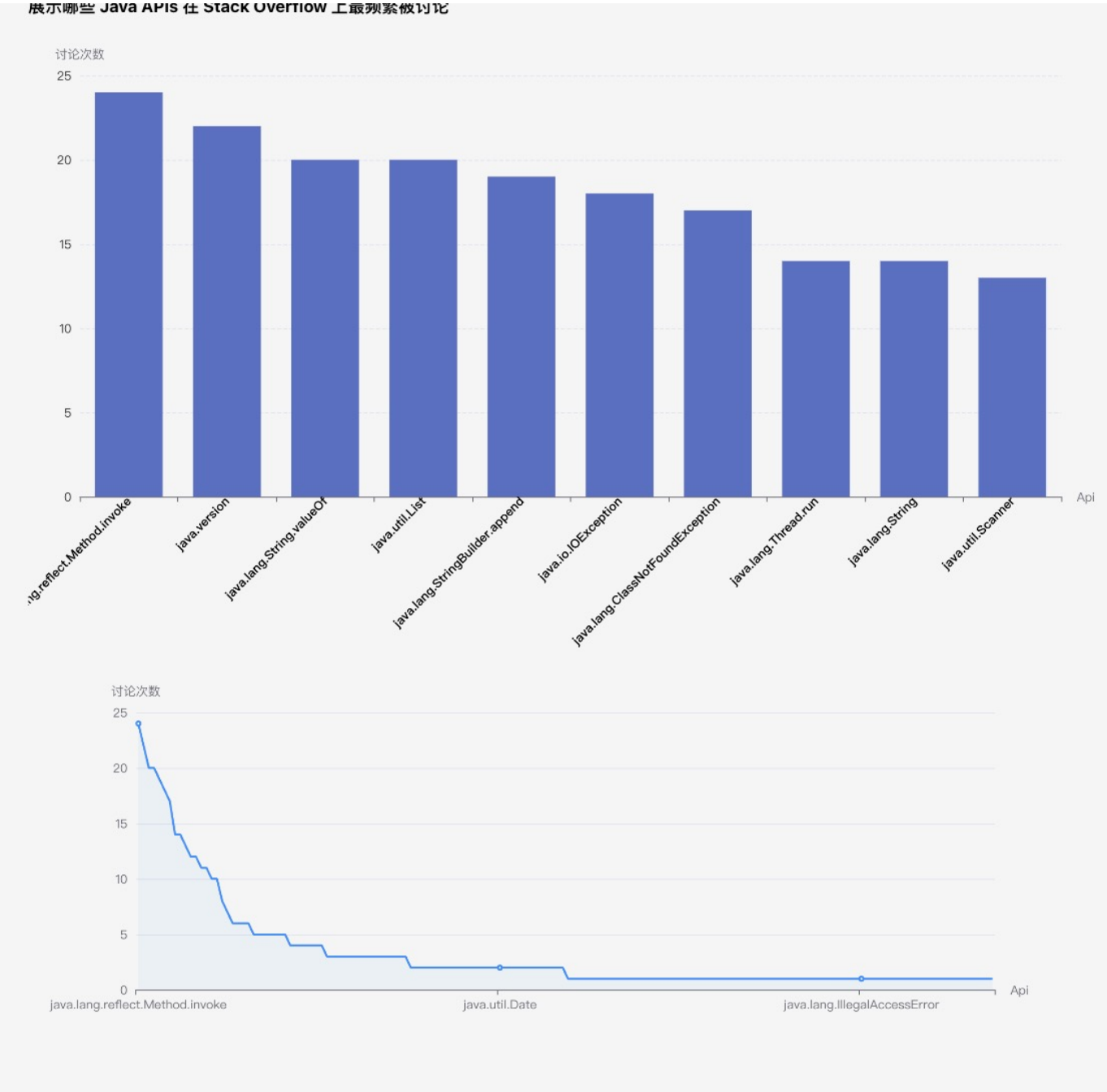


- Questions with a higher number of responses tend to have a higher number of comments. The reverse is also true.

Which are the most active users who frequently participate in thread discussions?

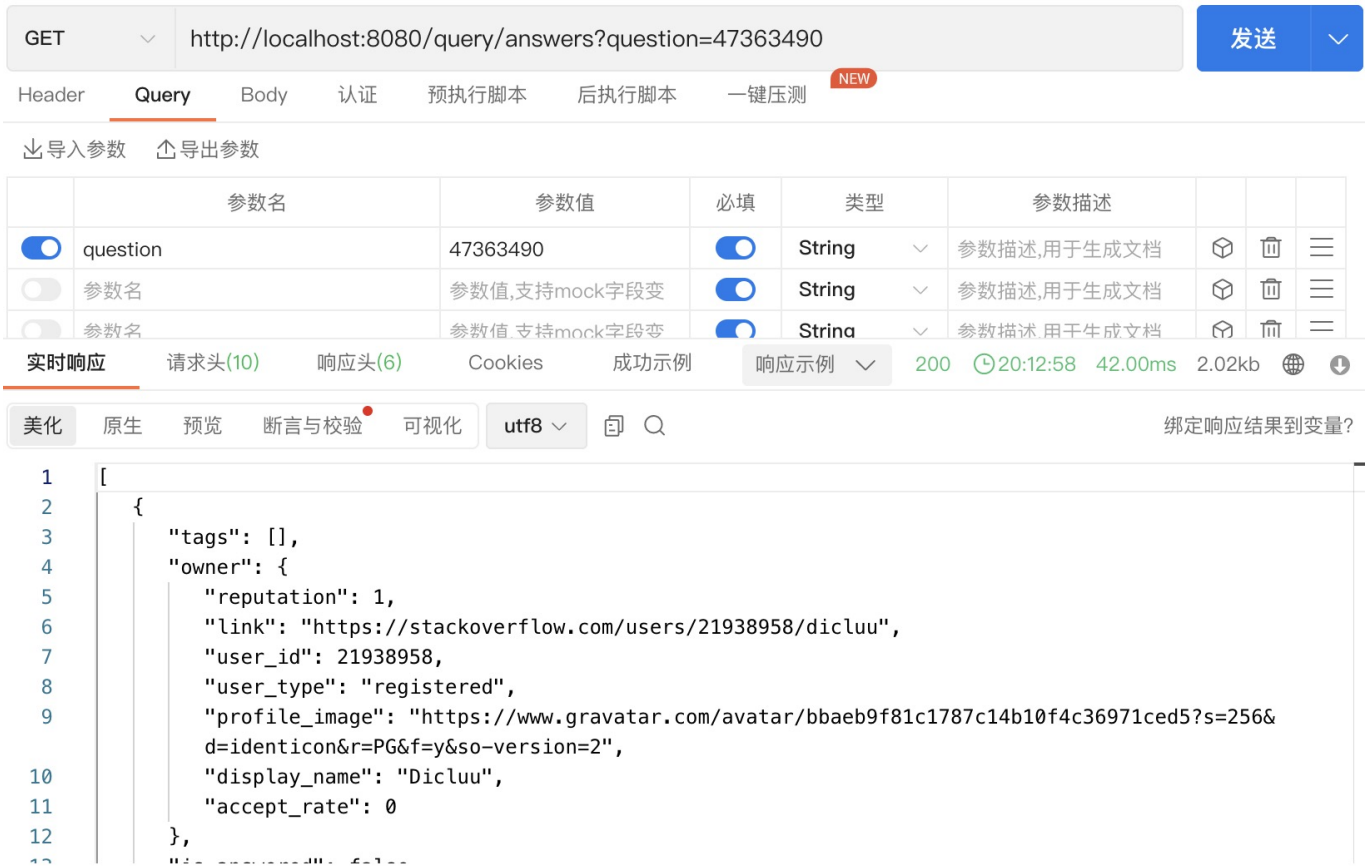
- Among the users with active records, most of them have only 1 active count.

Advance: Frequently discussed Java APIs

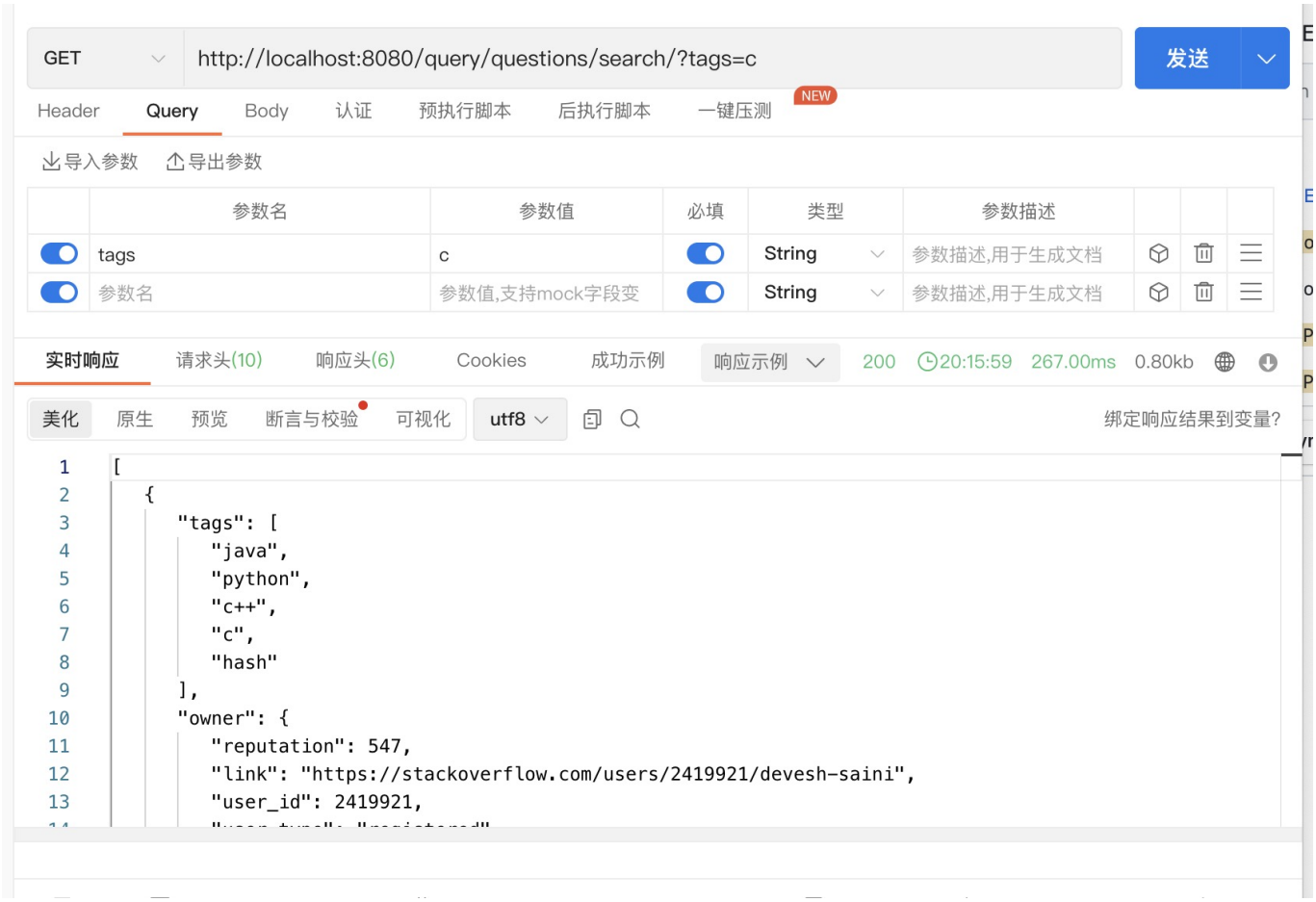


- java.lang.reflect.invoke

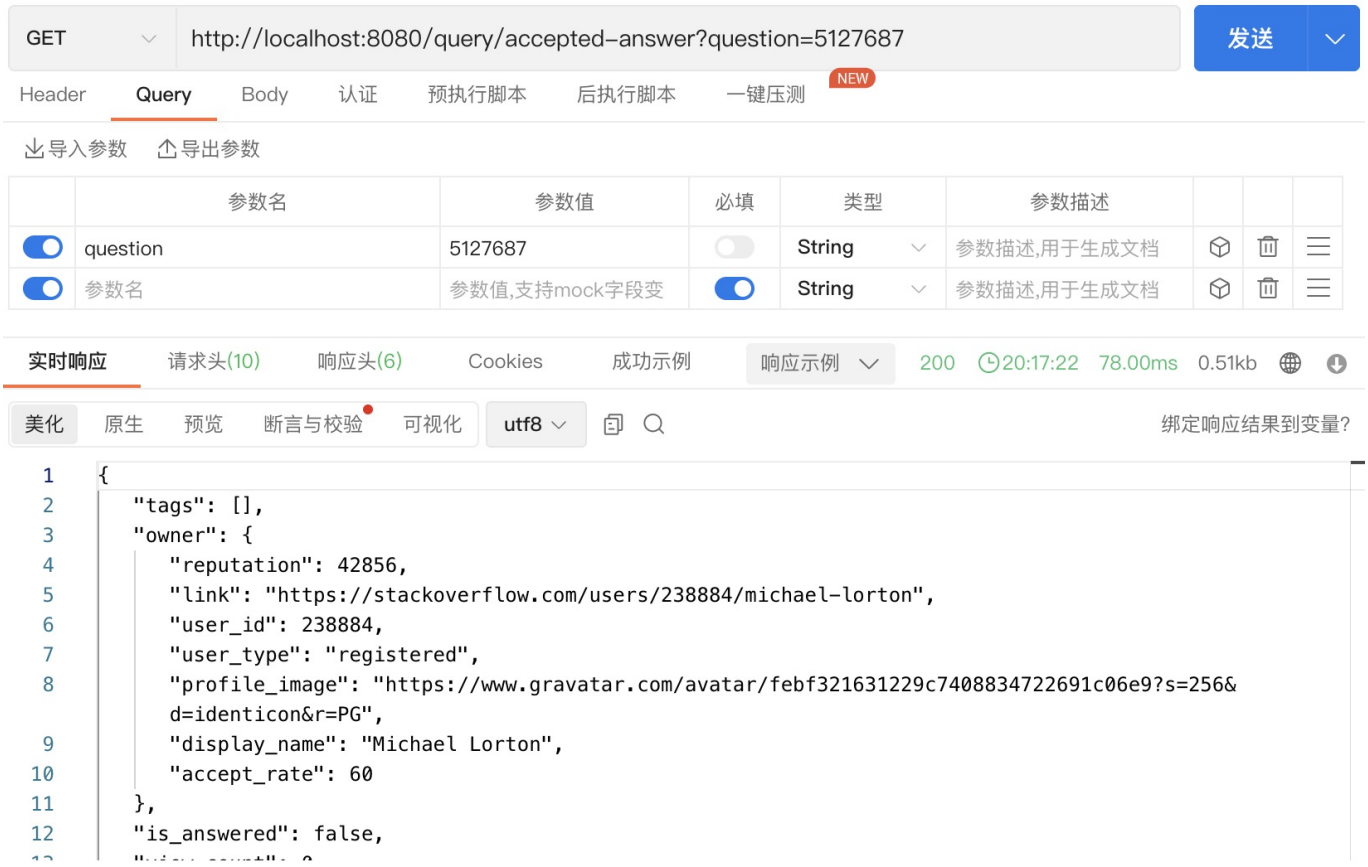
2. Restful API



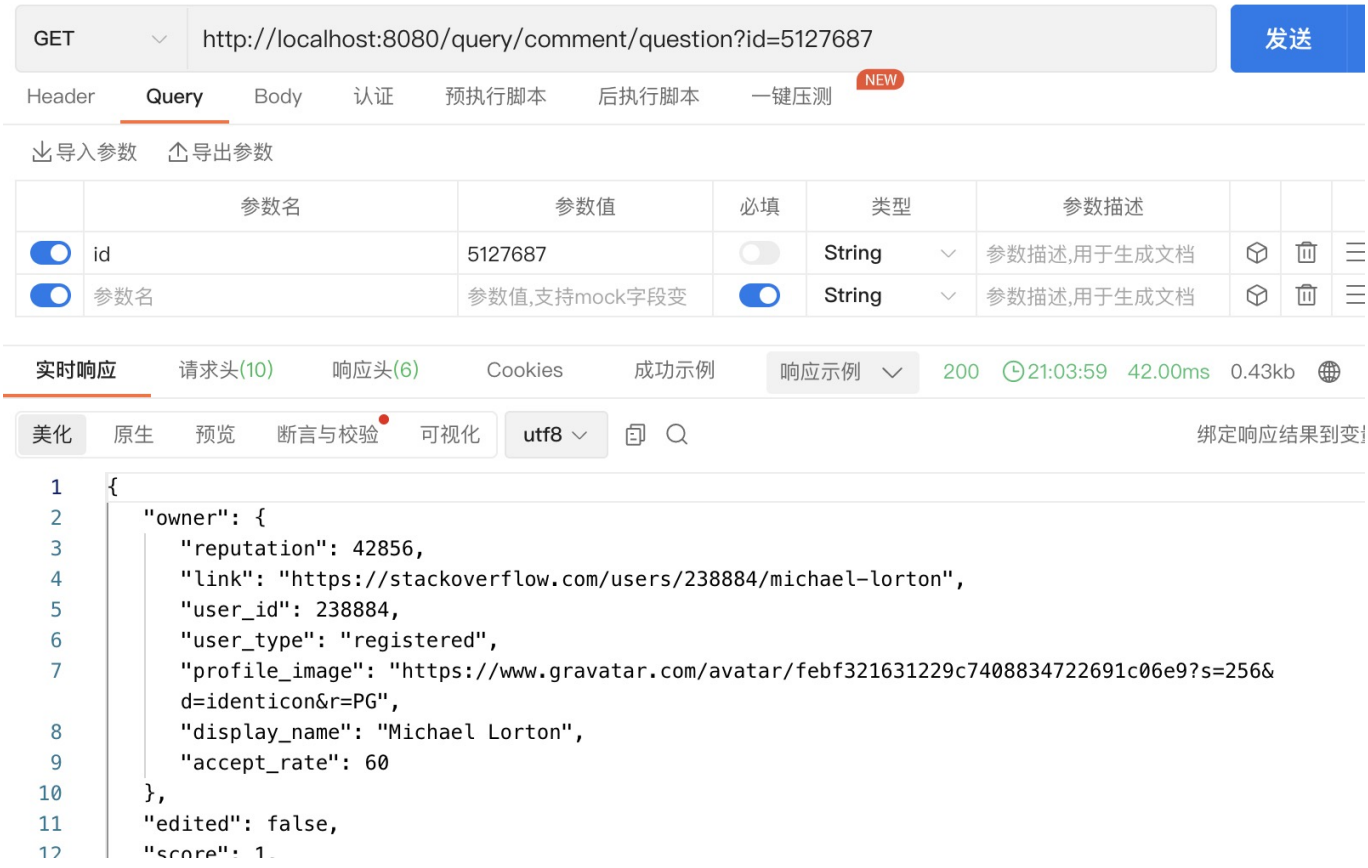
- find the answer according to the question id



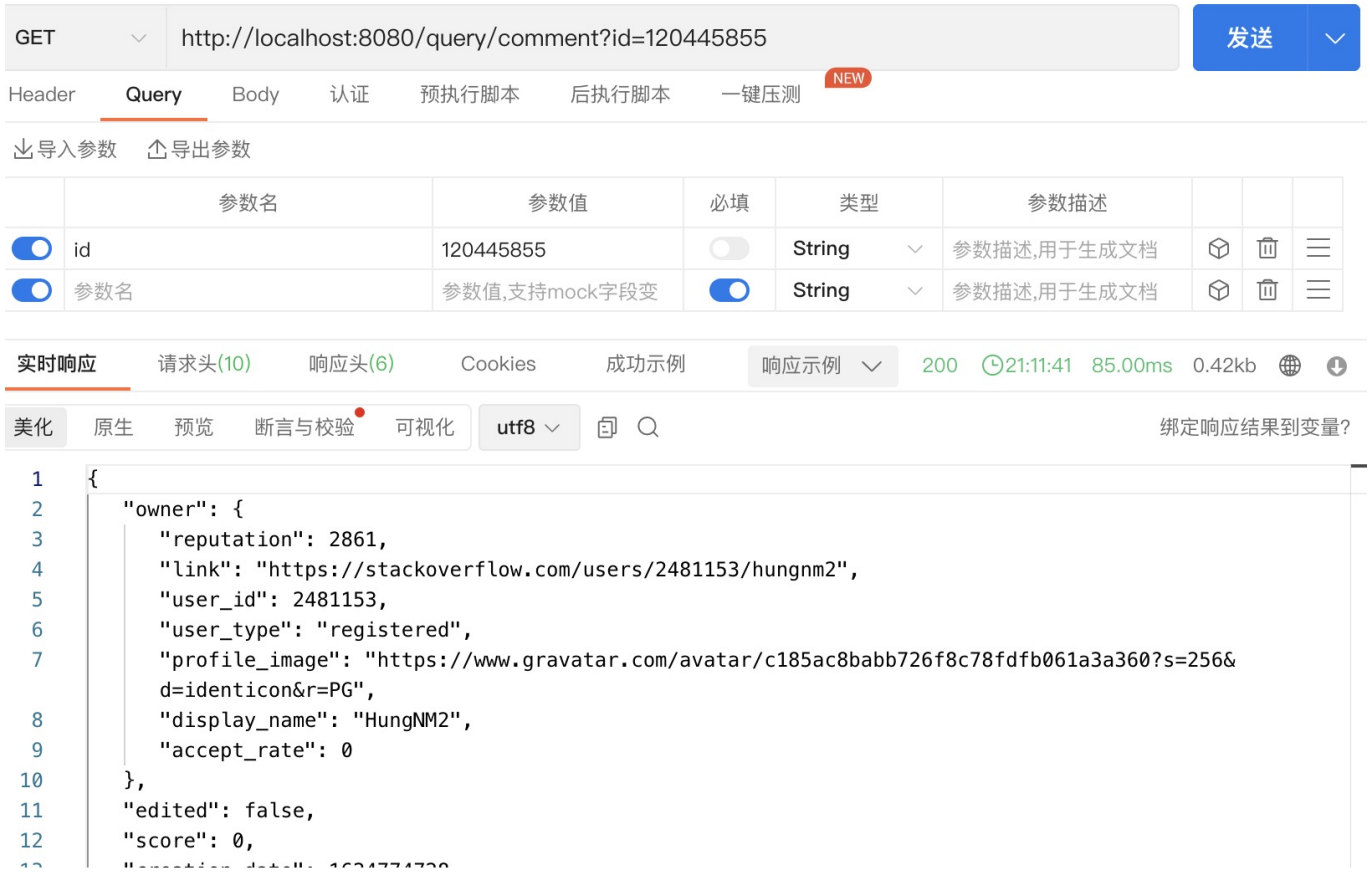
- find the problem according to the tag



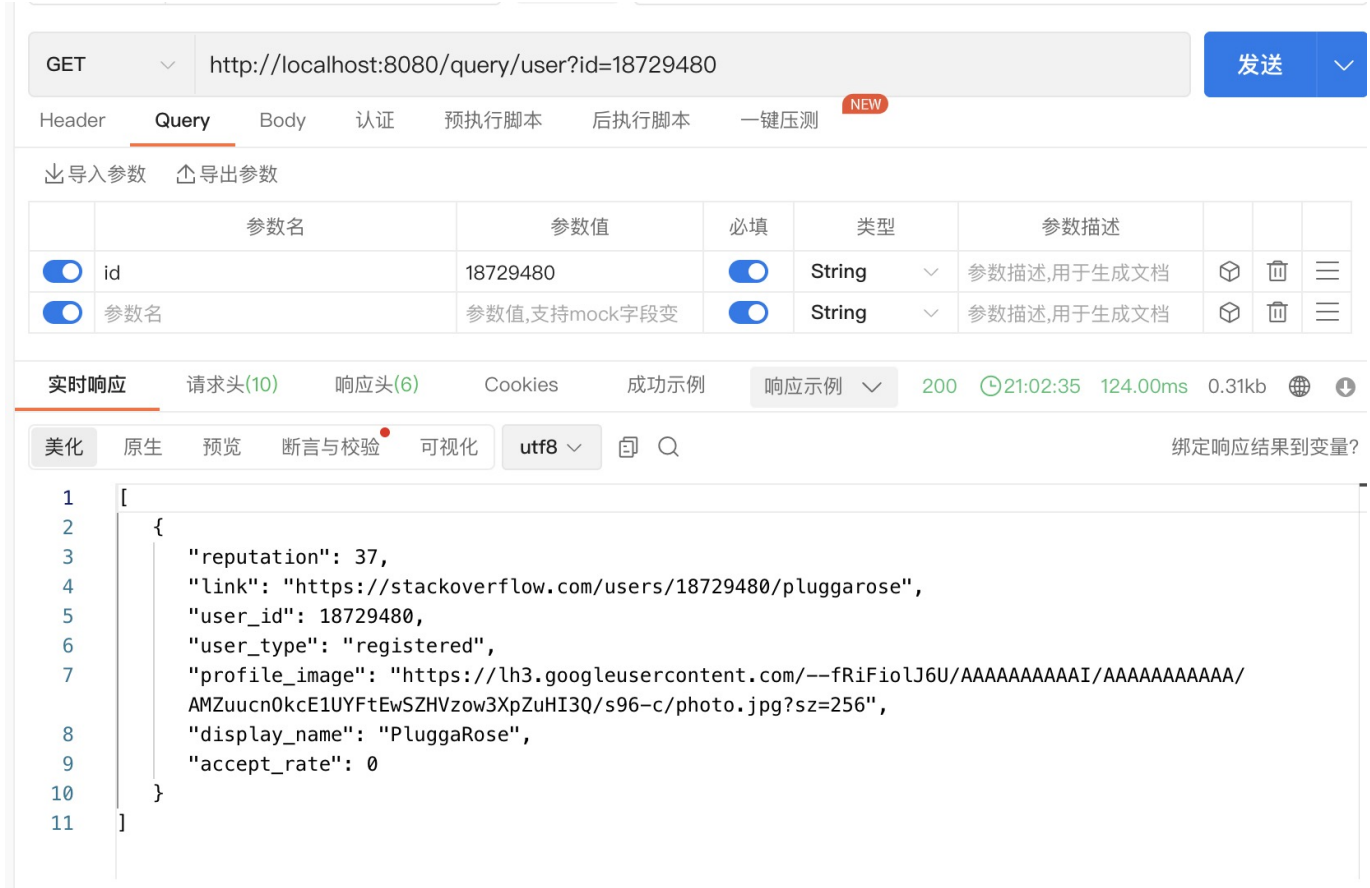
- find the accepted answer according to the question id



- find the comment according to the question id



- find comment by comment id



- find user by userid

3. ***Insight***

The user activity frequency on stackoverflow follows a ***pyramid structure***, which can provide us with the following insights:

1. User Distribution: The pyramid structure indicates an imbalance in user groups. Only a small portion of users are highly active on the platform, while the majority of users are less engaged or passive.

2. Key Users: The top of the pyramid represents highly active users who are often opinion leaders, content creators, or frequent participants in discussions and interactions on social media platforms. These users are crucial for platform growth and community development.

3. Long Tail Effect: The bottom of the pyramid consists of a large number of users with relatively low activity levels, but together, they contribute a significant portion of the overall activity. This reflects the long tail effect, where the cumulative contribution of many low-activity users can surpass that of a few high-activity users.

4. User Engagement: The pyramid structure of user activity suggests that platforms may need to take measures to enhance user engagement and activity levels. By providing engaging content, facilitating user interaction, and implementing reward mechanisms, platforms can stimulate user interest and participation to mitigate a decline in activity.

5. User Segmentation and Personalization: Understanding the distribution of user activity can help platforms segment users and provide personalized recommendations. By analyzing user behavior and interests, tailored content and experiences can be offered to users with different activity levels, thereby increasing their engagement.

The user question answerd frequency and most view tag ,upvote tag and api on stackoverflow follows a ***pyramid structure***, which can provide us with the following insights:

1. These features of Java are probably the most commonly used.eg.list,springboot,java.lang.reflect.invoke.
2. Some api may not be perfect yet or easy to user, so was asked a lot of times.
3. springboot is probably the most popular framework now.