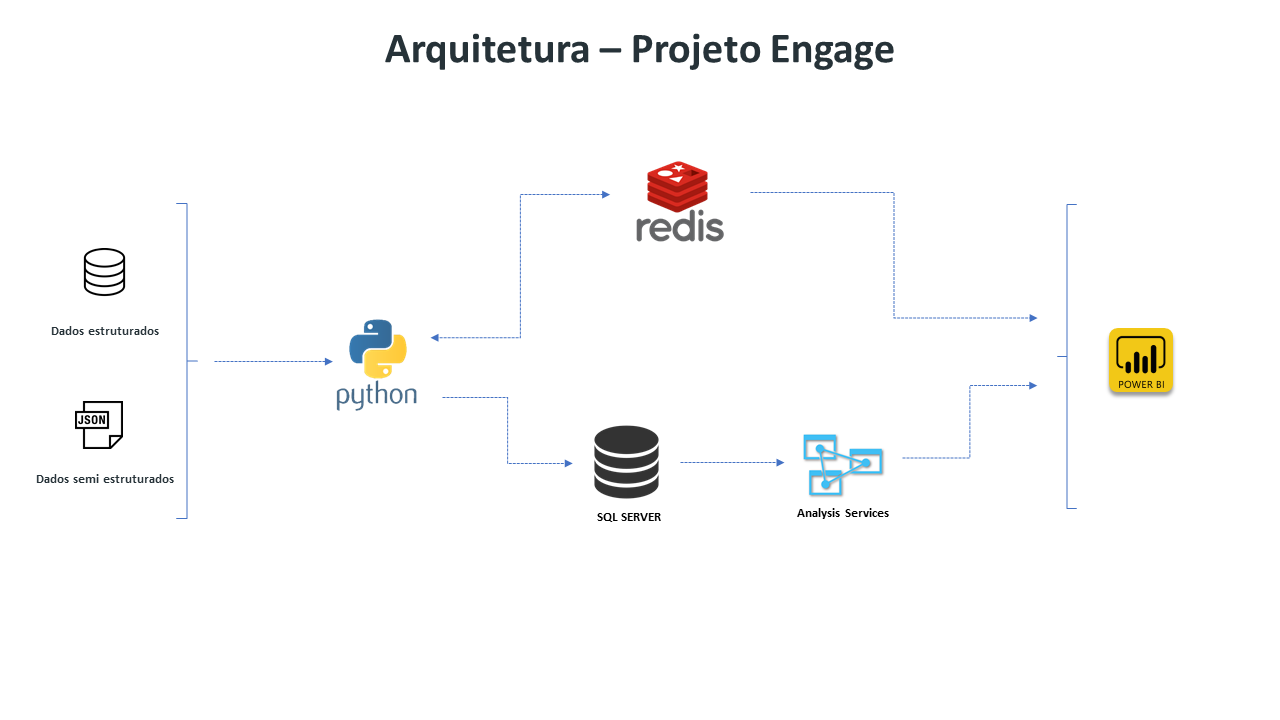
**Entregável 1 - Arquitetura de dados proposta**

1. **O diagrama da arquitetura de dados proposta.**

****

1. **Justificativa das tecnologias escolhidas.**

**Python:** Escolhemos utilizar a linguagem Python devido a sua grande quantidade de bibliotecas disponíveis, com elas conseguimos tratar e inserir os dados no formato escolhido dentro do Redis e se necessário realizar análises mais complexas em cima desses dados.

**Redis:** Optamos por utilizar o Redis devido ao seu alto desempenho, por sua estrutura de dados em memória, versatilidade e compatibilidade com a linguagem Python. Com os nossos dados armazenados em cache conseguimos diminuir a latência de acesso aos dados, aumentar a produtividade e aliviar a sobrecarga do banco de dados.

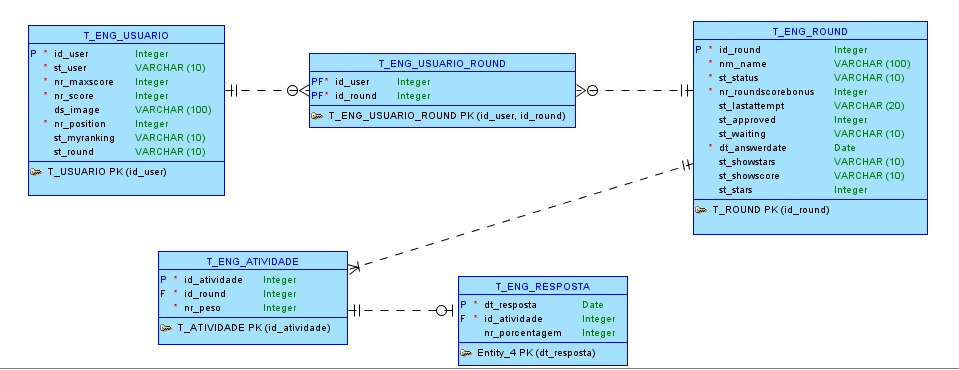
**SQL Server:** Estamos utilizando o banco de dados SQL Server devido ao seu alto desempenho e compatibilidade com outras ferramentas de mercado. Nele vamos armazenar os registros do banco transacional e outras fontes, e criar um Data Warehouse para obter informações e melhorar a tomada de decisões baseada nos nossos dados.

**Analysis Services (SSAS):** Escolhemos essa ferramenta porque permite construir estruturas multidimensionais (Cubos) para pré-calcular e armazenar os dados, com isso ganhamos em alta disponibilidade e escalabilidade.

**Power BI:** Decidimos escolher o Power BI como ferramenta de visualização por sua alta performance, por se conectar com diversas fontes de dados e a sua facilidade de implementação.

**Entregável 2 – Representação do modelo de dados para a solução**

1. **Modelo E/R com as principais entidades, relacionamentos e atributos existentes – representado na ferramenta Oracle Developer.**

****

1. **Metadados dos armazenamentos em NoSQL.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Campo Lógico | Campo Físico | Tipo de Dados | PK | FK | Restrição | Descrição |
| user | id\_user | Integer | Id\_user | Id\_user | UNIQUE | Item responsável por armazenar a identificação do usuário. |
| situação usuario | st\_user | VARCHAR(10) |  |  | NOT NULL | Item responsável por armazenar a situação do usuário. |
| número maxcore | nr\_maxscore | integer |  |  | NOT NULL | Item responsável por armazenar o número máximo do score. |
| número score | nr\_score | integer |  |  | NOT NULL | Item responsável por armazenar o número do score. |
| descrição Imagem | ds\_image | VARCHAR(100) |  |  | NOT NULL | Item responsável por armazenar a descrição da imagem. |
| número position | nr\_positiom | integer |  |  | NOT NULL | Item responsável por armazenar o número da posição. |
| situação ranking | st\_myranking | VARCHAR(10) |  |  | NOT NULL | Item responsável por armazenar a situação do ranking. |
| situação round | st\_round | VARCHAR(10) |  |  | NOT NULL | Item responsável por armazenar a situação do round. |

**Entregável 3 – Implementação da solução**

1. **Detalhes da instalação e da configuração realizada para a criação da arquitetura sugerida para a solução.**

**Instalação do programa:**

* Baixar o script: $ git clone <https://github.com/lucascoimbr/SS2.git>
* Abrir um terminal na pasta do programa
* Instalar a biblioteca Pandas
  + $ pip install pandas
* Na pasta redis\_server, clicar no arquivo redis-server.exe para ativar o servidor redis
* Clique em redis-cli.exe para abrir a CLI do redis.
* Para rodar o script, basta utilizar a linha de comando:
  + $ python main.py

Texto

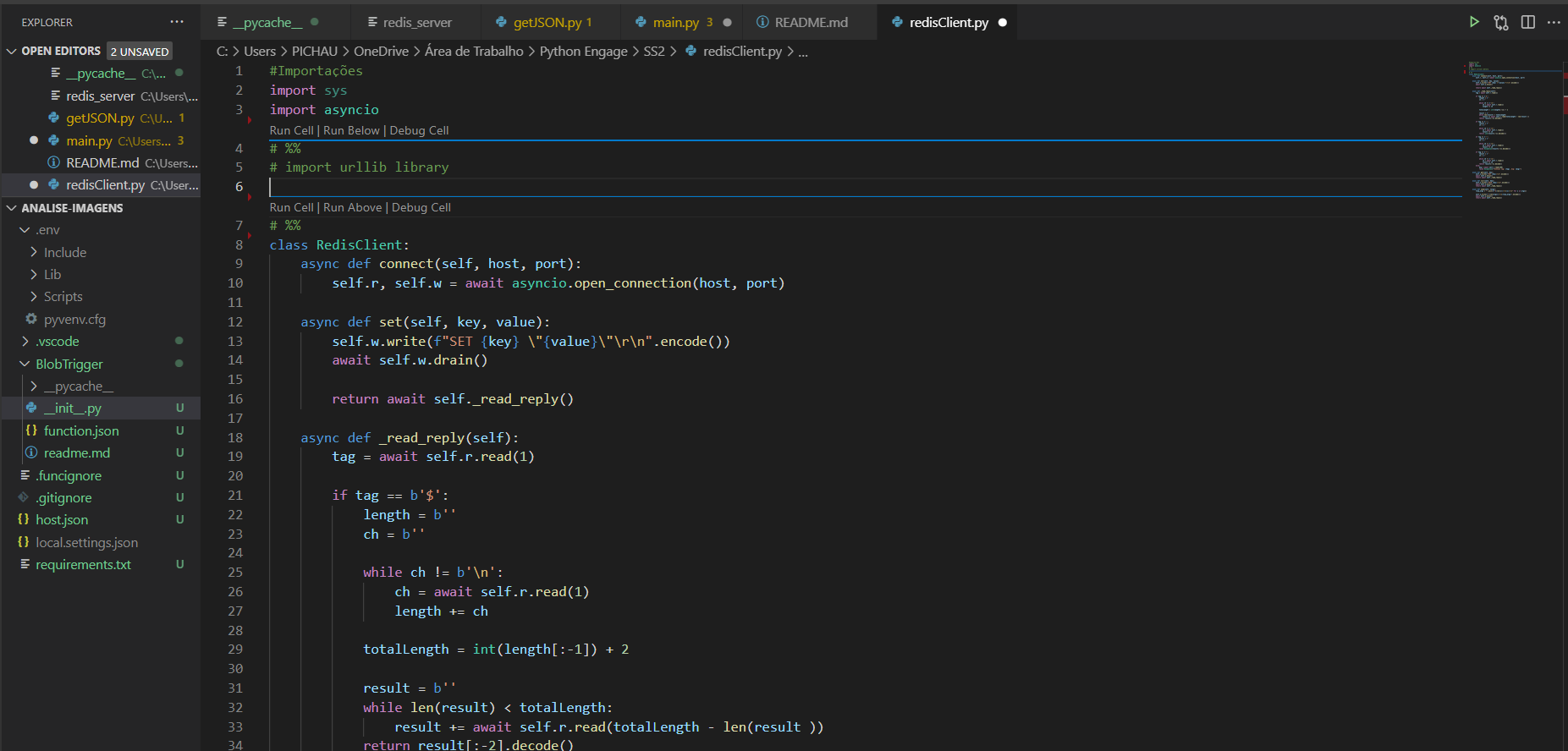
Descrição gerada automaticamente

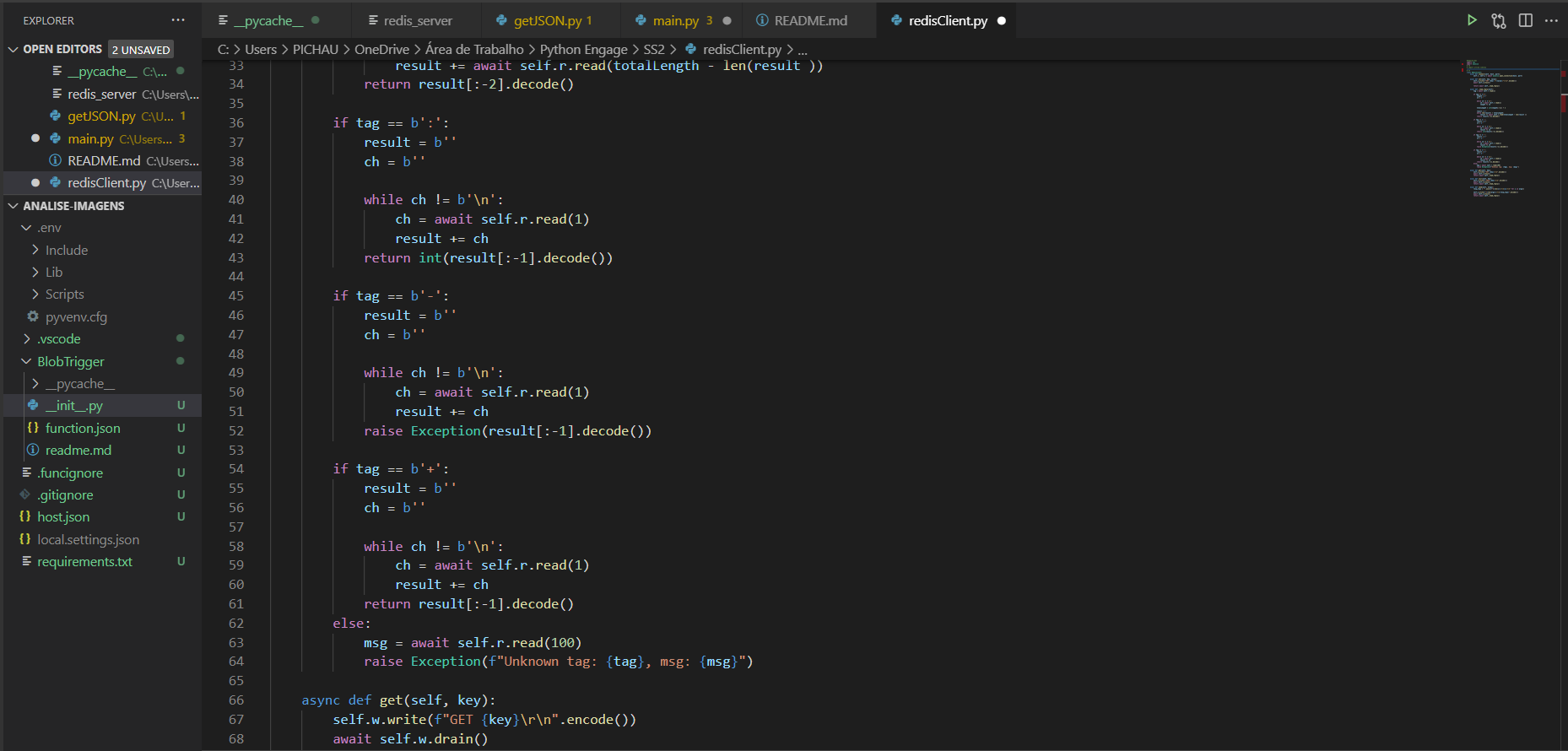
**Consulta do ranking:**

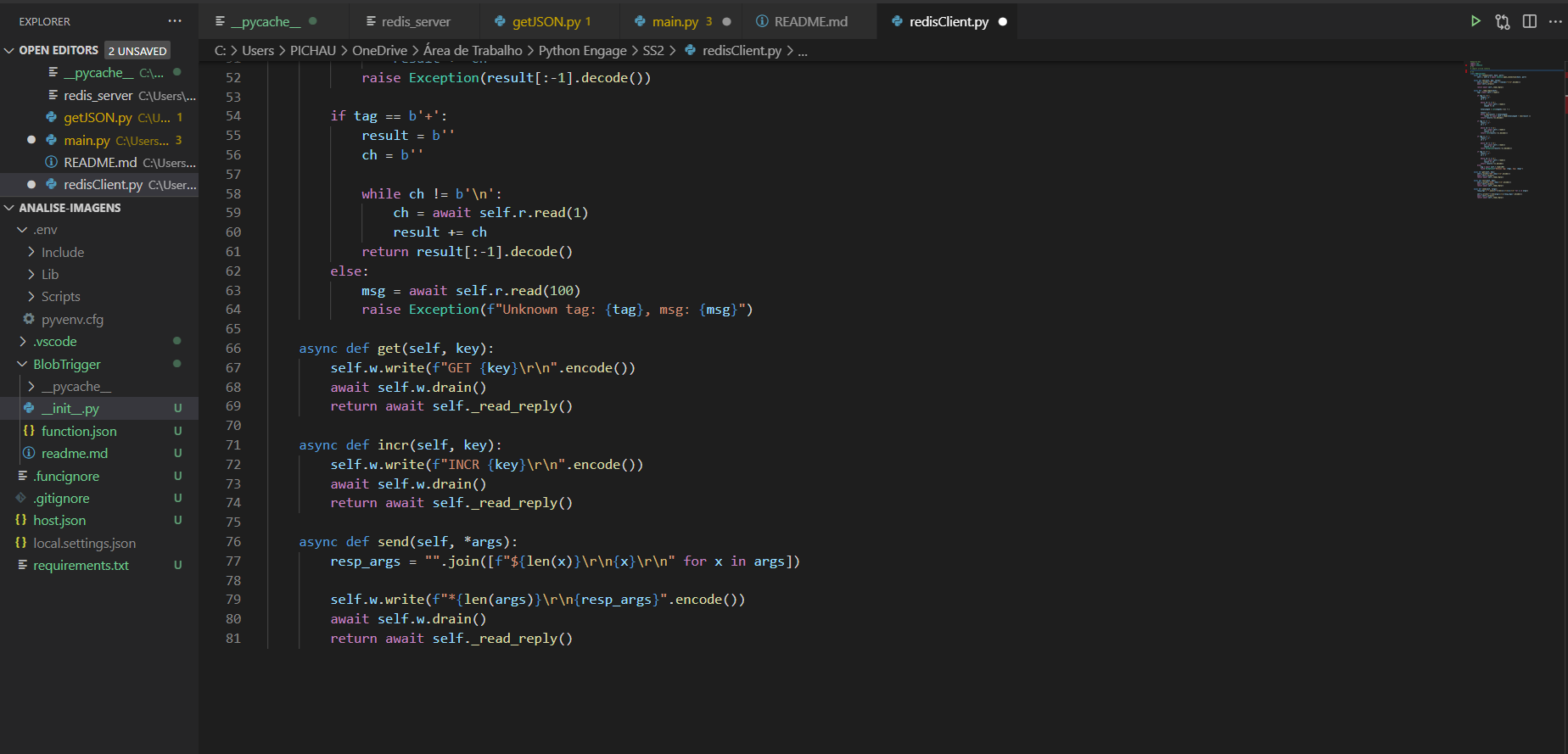
No redis-cli, basta utilizar o comando get rank, que o JSON será retornado.

1. **Scripts de criação e inserção de ocorrências nos bancos NoSQL e/ou tecnologias utilizadas.**

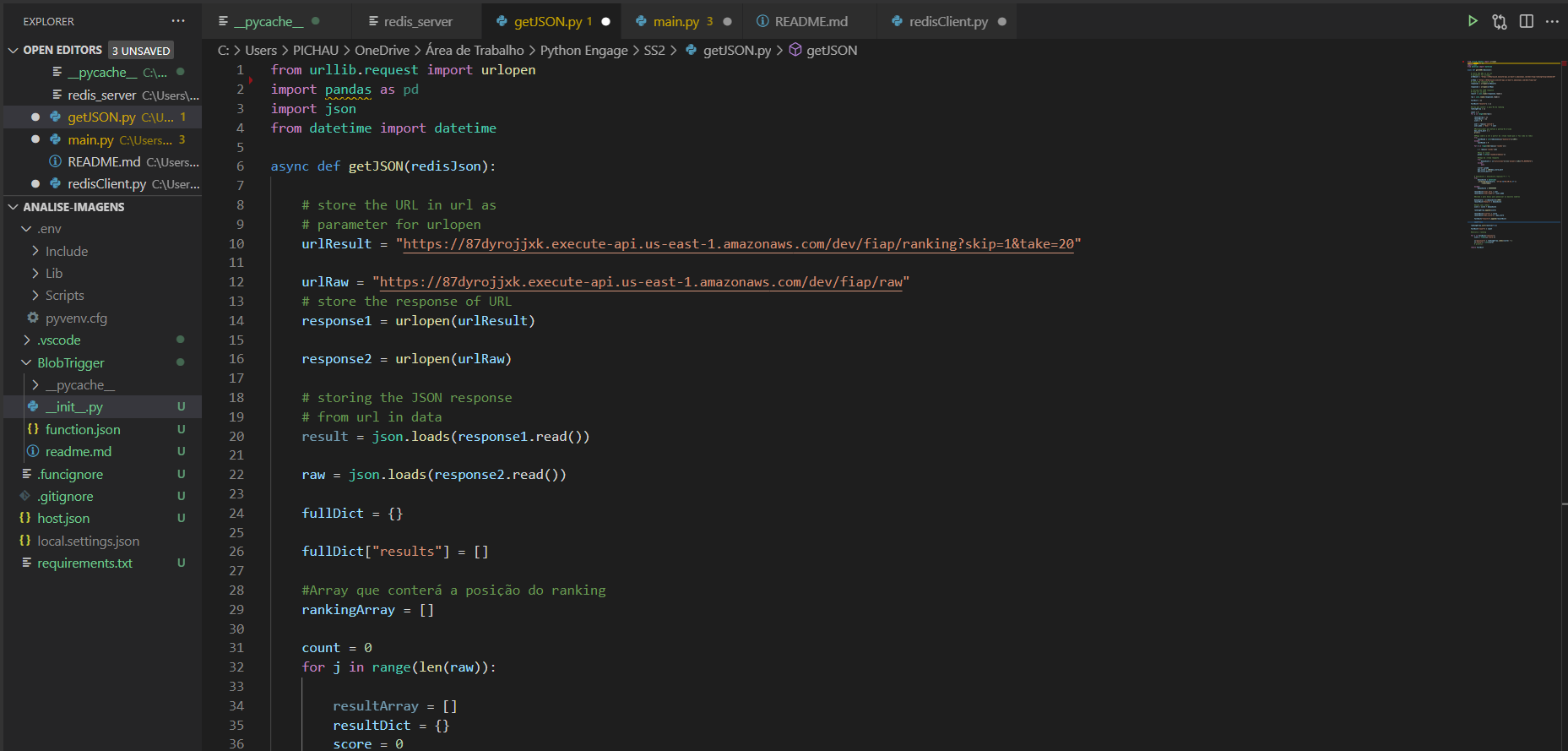
**Scripts de criação:**

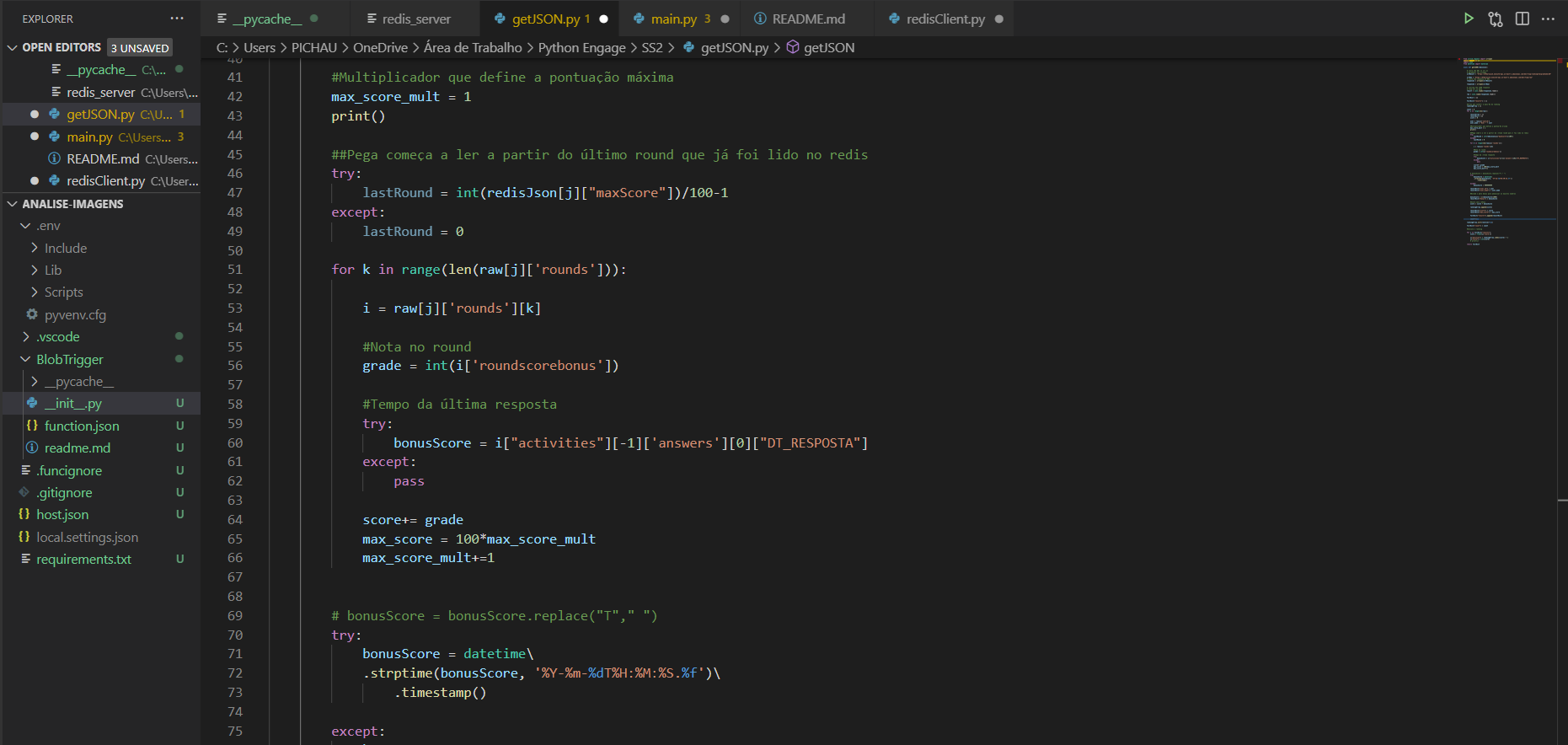


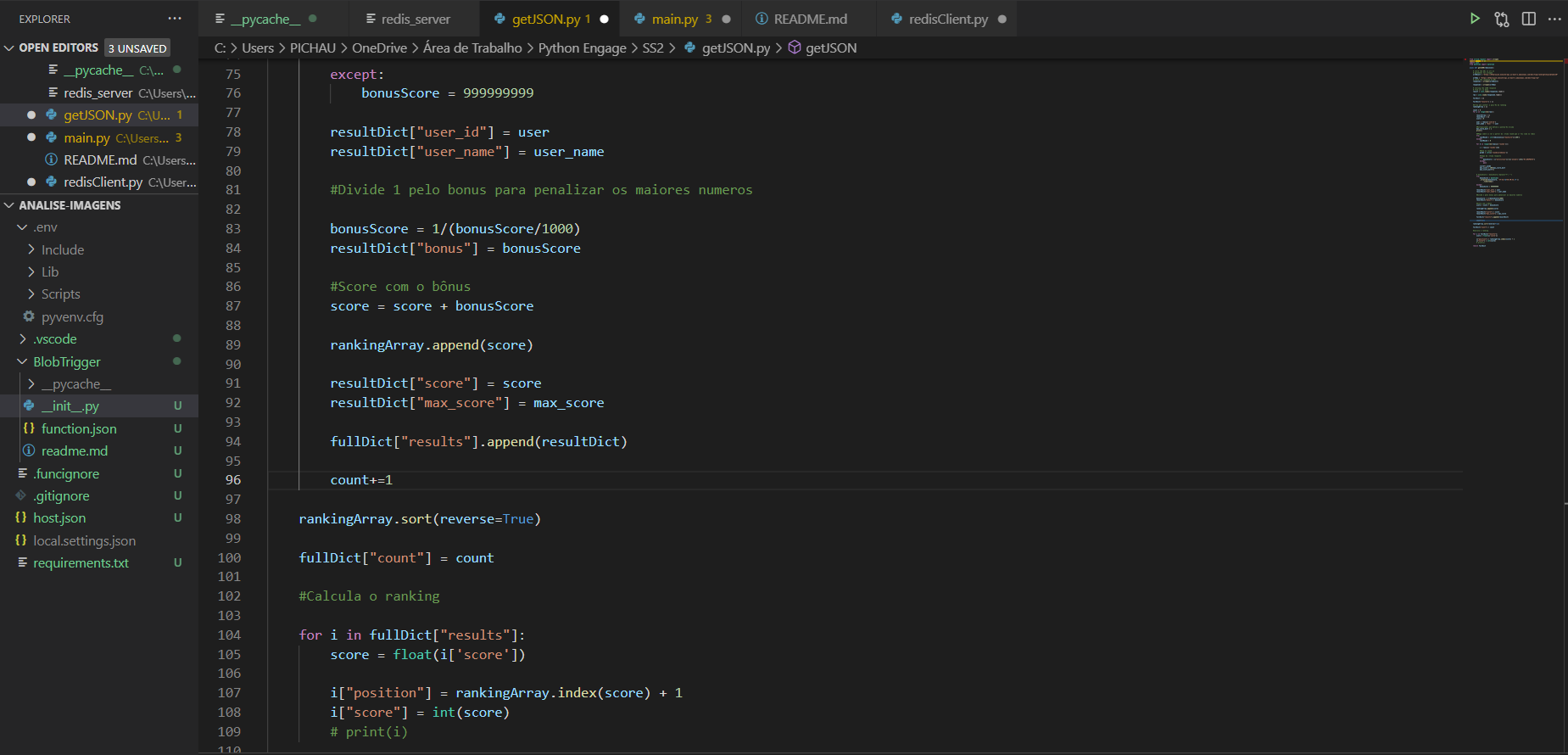




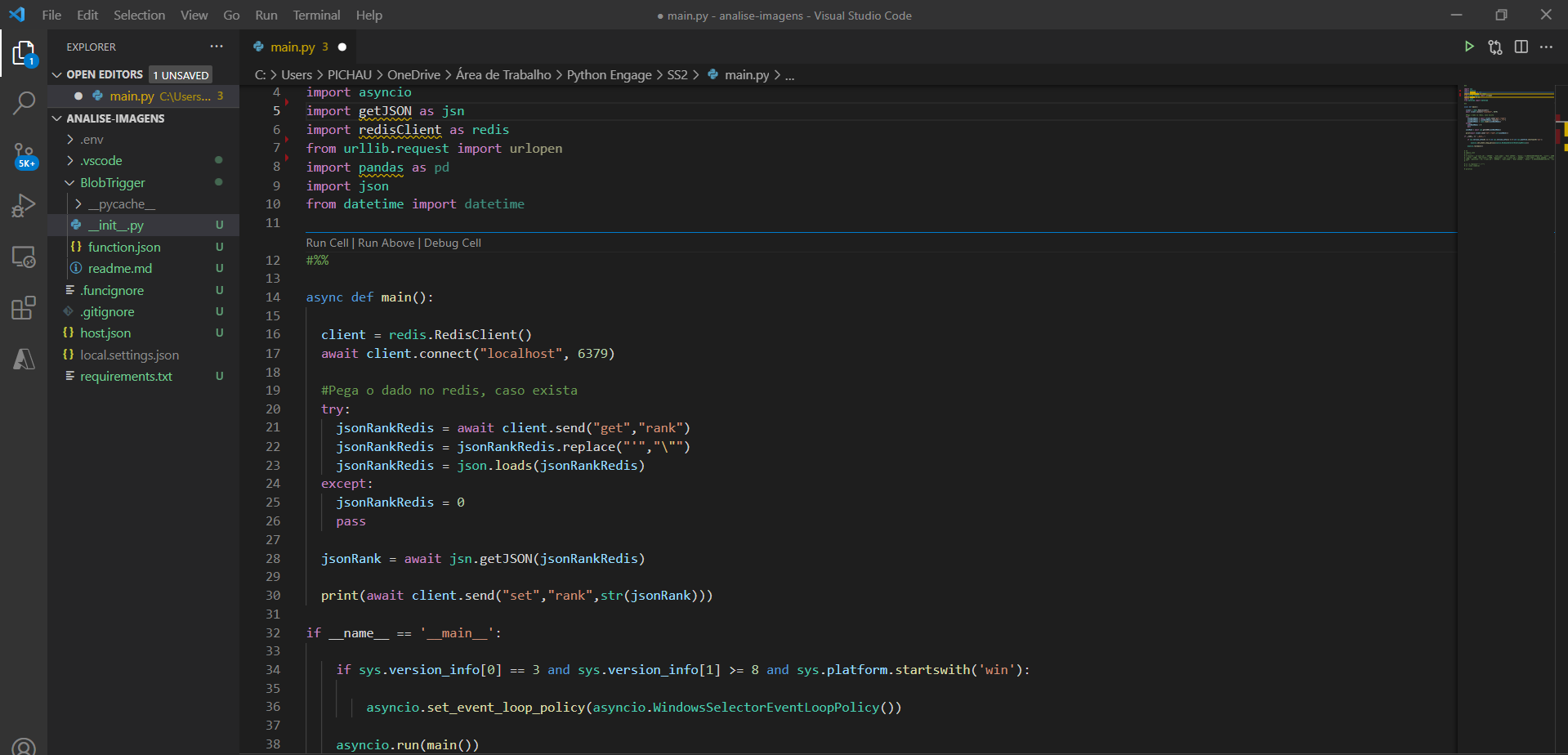
**Scripts de cálculos:**

****

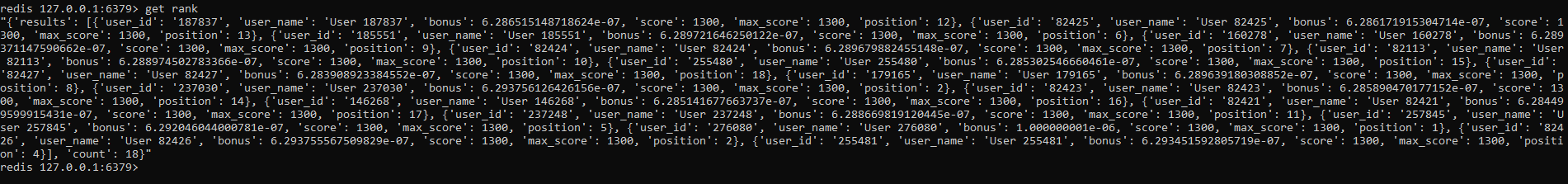
****

****

**Script de inserção:**



**C) Consultas (queries) para obtenção dos rankings.**



Comando executado (**get rank**) e o retorno:

redis [127.0.0.1:6379](http://127.0.0.1:6379/)>get rank  
"{'results': [{'user\_id': '187837', 'user\_name': 'User 187837', 'bonus': 6.286515148718624e-07, 'score': 1300, 'max\_score': 1300, 'position': 12}, {'user\_id': '82425', 'user\_name': 'User 82425', 'bonus': 6.286171915304714e-07, 'score': 1300, 'max\_score': 1300, 'position': 13}, {'user\_id': '185551', 'user\_name': 'User 185551', 'bonus': 6.289721646250122e-07, 'score': 1300, 'max\_score': 1300, 'position': 6}, {'user\_id': '160278', 'user\_name': 'User 160278', 'bonus': 6.289371147590662e-07, 'score': 1300, 'max\_score': 1300, 'position': 9}, {'user\_id': '82424', 'user\_name': 'User 82424', 'bonus': 6.289679882455148e-07, 'score': 1300, 'max\_score': 1300, 'position': 7}, {'user\_id': '82113', 'user\_name': 'User 82113', 'bonus': 6.288974502783366e-07, 'score': 1300, 'max\_score': 1300, 'position': 10}, {'user\_id': '255480', 'user\_name': 'User 255480', 'bonus': 6.285302546660461e-07, 'score': 1300, 'max\_score': 1300, 'position': 15}, {'user\_id': '82427', 'user\_name': 'User 82427', 'bonus': 6.283908923384552e-07, 'score': 1300, 'max\_score': 1300, 'position': 18}, {'user\_id': '179165', 'user\_name': 'User 179165', 'bonus': 6.289639180308852e-07, 'score': 1300, 'max\_score': 1300, 'position': 8}, {'user\_id': '237030', 'user\_name': 'User 237030', 'bonus': 6.293756126426156e-07, 'score': 1300, 'max\_score': 1300, 'position': 2}, {'user\_id': '82423', 'user\_name': 'User 82423', 'bonus': 6.285890470177152e-07, 'score': 1300, 'max\_score': 1300, 'position': 14}, {'user\_id': '146268', 'user\_name': 'User 146268', 'bonus': 6.285141677663737e-07, 'score': 1300, 'max\_score': 1300, 'position': 16}, {'user\_id': '82421', 'user\_name': 'User 82421', 'bonus': 6.284499599915431e-07, 'score': 1300, 'max\_score': 1300, 'position': 17}, {'user\_id': '237248', 'user\_name': 'User 237248', 'bonus': 6.288669819120445e-07, 'score': 1300, 'max\_score': 1300, 'position': 11}, {'user\_id': '257845', 'user\_name': 'User 257845', 'bonus': 6.292046044000781e-07, 'score': 1300, 'max\_score': 1300, 'position': 5}, {'user\_id': '276080', 'user\_name': 'User 276080', 'bonus': 1.000000001e-06, 'score': 1300, 'max\_score': 1300, 'position': 1}, {'user\_id': '82426', 'user\_name': 'User 82426', 'bonus': 6.293755567509829e-07, 'score': 1300, 'max\_score': 1300, 'position': 2}, {'user\_id': '255481', 'user\_name': 'User 255481', 'bonus': 6.293451592805719e-07, 'score': 1300, 'max\_score': 1300, 'position': 4}], 'count': 18}"