In this essay, we are going to talk about the advantage and the disadvantage about choosing different kinds of databases which has considered about the project requirements.

First of all, we have to consider the problem from the aspect of data storage.

(1) All Redis data is stored in memory, and persistence uses RDB or aof.

(2) All the data of MongoDB are actually stored on the hard disk, and all the data to be operated are mapped to a certain area of the memory by mmap. Then, MongoDB performs data modification in this area, avoiding fragmented hard disk operations. As for the flushing of the content on the mmap to the hard disk, that is the operating system's business. Therefore, if MongoDB modifies the data in memory, and then, before the mmap data is flushed to the hard disk, the system goes down and the data will be lost.

(3) MySQL stores both data and indexes in the hard disk. It is only swapped into memory when it is needed. And it can handle data that far exceeds the total amount of memory.

So from this point of view, MongoDB seems to be a relatively good choice, but pay attention to the above mentioned, if the system crashes before the mmap data is flushed to the hard disk, the data will be lost, which will be a potential risk , but I think the probability of downtime can be artificially reduced.

Secondly, as mentioned in the requirements, the amount of data in this database may be very large in the future, so data storage capacity is also a topic that has to be mentioned.

When physical memory is not enough, both Redis and mongodb will use virtual memory.

In fact, if Redis wants to start virtual memory, it is obvious that you must either add a memory stick or change the database.

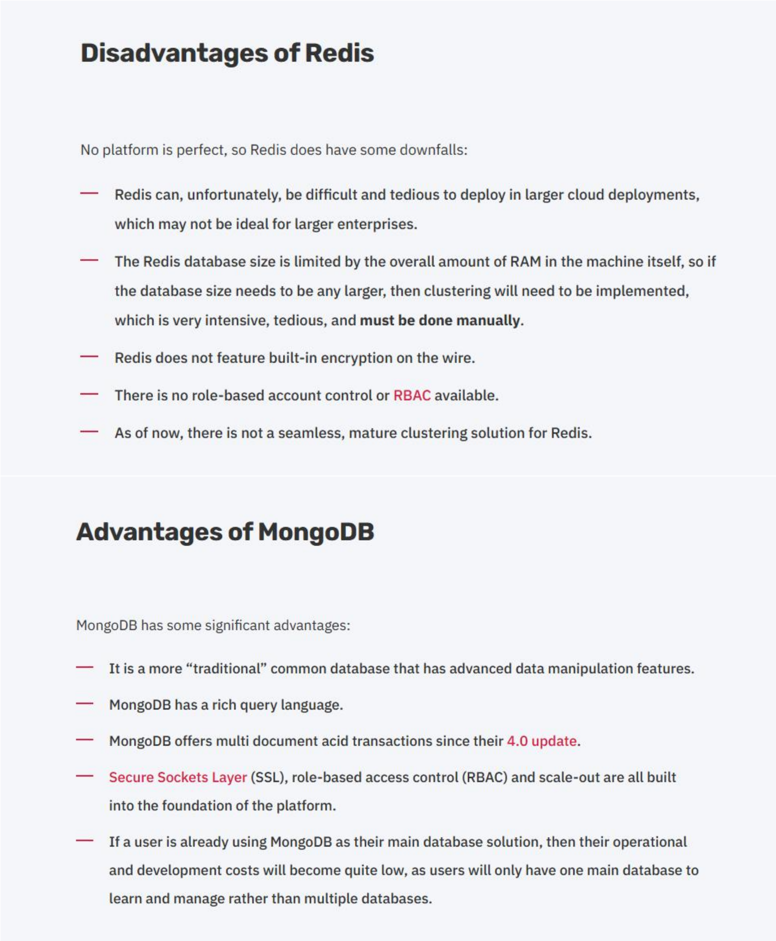
However, MongoDB is different, as long as the business can guarantee that the read-write ratio of hot and cold data makes the hot data in the physical memory, and the exchange of mmap is less. Thus MongoDB can still guarantee performance.

Redis transaction support is relatively weak, and can only guarantee the continuous execution of each operation in the transaction

MongoDB does not support transactions

Additionally, as mentioned in the requirements, the most important operation is to read data extremely fast for players to query. By comparing the reading speed of Redis and MongoDB, we found that Redis is the faster one, but one thing we cannot ignore is MongoDB is also very fast, just not as fast as Redis, so I think MongoDB's speed should be enough. In this part I want to talk about cosmos, which is also faster than MongoDB, but cosmos still have a lot of disadvantages. From the internet, I learned that ***Cosmos DB is positioned as providing a globally scalable, multi-model database supporting operational applications. It borrows many concepts from MongoDB, but falls short in compatibility and functionality. For example, you can only query data using a single model. If you provision a container with the SQL API, you cannot query it with the Cosmos DB emulation API. Cosmos DB also does not support time-series data. MongoDB, on the other hand, lets you run key-value, graph, and SQL queries against the same data. And, with MongoDB 5.0, you can build and run applications with support for specific time-series data storage and query patterns*** (resource from Comparing Microsoft Cosmos DB And MongoDB | MongoDB)**.** There is a fatal flaw here, cosmos does not support time series data, which is why I do not choose this database. We cannot predict what type of data will appear in this project in the future, so MongoDB has a wider range of applications.

Then it comes to the most important reason of why i will choose MongoDB, by comparing the advantages and disadvantages of MongoDB and Redis, we can find that as the amount of data grows, the processing of Redis will become more and more complicated so that it will have a great impact on the management of developers, even though Redis has many Advantages, we also need to consider the long-term goal. Even if the reading speed of MongoDB is not as fast as Redis, we should choose MongoDB, otherwise the cost of managing the database will be particularly high in the future.



Disadvantages of Redis

No platform is perfect, so Redis does have some downfalls:

- Redis can, unfortunately, be difficult and tedious to deploy in larger cloud deployments, which may not be ideal for larger enterprises.
- The Redis database size is limited by the overall amount of RAM in the machine itself, so if the database size needs to be any larger, then clustering will need to be implemented, which is very intensive, tedious, and **must be done manually**.
- Redis does not feature built-in encryption on the wire.
- There is no role-based account control or RBAC available.
- As of now, there is not a seamless, mature clustering solution for Redis.

Advantages of MongoDB

MongoDB has some significant advantages:

- It is a more "traditional" common database that has advanced data manipulation features.
- MongoDB has a rich query language.
- MongoDB offers multi document acid transactions since their 4.0 update.
- Secure Sockets Layer (SSL), role-based access control (RBAC) and scale-out are all built into the foundation of the platform.
- If a user is already using MongoDB as their main database solution, then their operational and development costs will become quite low, as users will only have one main database to learn and manage rather than multiple databases.

(the images resource are from Redis vs MongoDB: Which One Should You Choose in 2022? (sfappworks.com))

Finally, from the picture above we can get a information about MongoDB:
if a user is already using MongoDB as their main database solution, then their operational and development costs will become quite low, as users will only have one main database to learn and manage rather than multiple databases. Considered about the cost of developing a game and the cost of post-maintenance, the lower when we spend in developing a game, the better.

In conclusion, those are all the reasons why I choose MongoDB and I had compared it with other databases from several aspects. It can be absorbed that MongoDB has no obvious fatal flaw in this project, maybe MongoDB in some particular functions is not perfect, but that's enough.