Why You Need Real-Time Session Stores to Maximize Engagement

Introduction

Player expectations are sky-high. Gameplay must be fast, entertaining, and tailored to the player. Real-time session stores have become integral in meeting these demands and keeping players engaged.

In the worst case scenario, poor game session management causes players to lose all of their inventory, personalization, and progress each time they log in. This will butcher engagement and trigger a string of events that begin with a drop in DAUs and end with a decrease in revenue. No developer can achieve victory in the gaming arena without real-time session stores.

Session stores have to be in real time to guarantee consistent and seamless responses that all players expect. This is especially important given the global prominence of online multiplayer games.

At any moment, millions of players from around the world could be interacting with your game, requiring your database to process mountains worth of session data with hyper-efficiency. Even being just an inch off the mark is enough to impair performance with lags, poor matchmaking, and forgotten inventories.

Delivering real-time session stores is easier said than done. But with the right database, your game sessions can be processed at lightning speed that will supercharge all other elements of your gaming performance.

What does real-time mean?

Research into human response indicates applications have roughly 100 milliseconds (ms)—one third of the time it takes to blink—before users feel like they're waiting for a response. To be considered real-time, the session store needs to send a request, have it processed, receive the response, and return it to the player in less than 100ms.

Session stores are crucial to skyrocketing engagement levels

A personalized experience is essential

A 'session state' is data that captures the current status of a user's interactions within a game. It's used to keep a record of all player interactions, including personalization information, recent actions, player level, login credentials, user identity, themes, and more.

During this period, all session-related data will be stored in either the main memory or in a database that won't lose the data when the game is finished. Session stores are crucial for a whole variety of reasons.

For one, when a player logs into their game again, they expect to pick up from where they left off—nobody wants to invest more time jumping through the same hoops. Session stores share all unique player information with databases to provide chronological progression through gameplay.

This draws a parallel with game inventories. Session stores are used to help ensure that players don't ever have to log in to find that they have items missing in their inventory, which can be seen as virtual robbery.

Players will feel cheated, tensions will arise, and they'll view the entire inventory building process as a waste of time. Few will be willing to gamble on building everything back up, knowing that they may lose it all again due to a technical deficiency.

We also can't ignore the impact that session stores have on personalization. Game-led personalization uses machine learning algorithms to analyze player actions and make adjustments to the experience.

Player actions, in-play habits, and interactions are monitored and saved in a session store, providing databases with the information to make the personalization process easier, faster, and more reactive.

Allowing the game to mold itself to fit the tastes and preferences of the player will boost engagement because we all love personalization. The more personal something is, the more attached to it we feel. This is no different in gaming.

And then we've also got to factor in the importance of leaderboards. Competition can make or break a game.

Having no leaderboard to measure progress stifles competition. Players need a goal to aim at and a hierarchy to climb since it's all about differentiating the winners from the losers. All of these elements make up a core pillar in gaming that keeps players entertained, builds communities, and maximizes engagement.

Yet without session stores, games would not have the data required to pull off these elements together and ramp up the playing experience. A game would become unplayable based on modern standards since no player wants to invest hours of their time to only discover that all of their progress has been wiped out as soon as they log back in.

Matchmaking can't happen without sessions stores

The massively multiplayer online games (MMOG) industry was worth around $10 billion in 2020 and it continues to grow in leaps and bounds. As MMOG grows in size, competition for the consumer's attention will become more fierce, requiring game developers to enhance all areas of the multiplayer experience.

Matchmaking is the process of matching players with the most suitable opponents to get the most out of the playing experience. Algorithms use session stores to extract more information about the level of each player, measuring a range of factors including experience, skill, and playing frequency.

Session stores allow the video game to gauge the skill level of each player to create teams and pit them against opposition of a similar level. Poor matchmaking poses immense ramifications on engagement.

Novices who are mismatched against elite performers are likely to suffer consecutive heavy defeats that'll discourage them from future participation. After all, there are only so many times a player is willing to lose before walking away.

And for the veterans, a lack of competitiveness creates a less stimulating experience that leads to boredom. For the adrenaline rush to kick in, the fraught likelihood of losing needs to creep in from time to time.

Players expect real-time sessions and nothing less

Player expectations are at an all-time high. From the second they jump into a game to the moment they put down the controller, they expect gameplay to be fast and hyperresponsive to commands. This means no lags, no freezes, and no errors whatsoever.

They also expect to pick up from where they left off and continue to expand on what they've already built. A mere blip in performance is enough to frustrate players, which over time, will accumulate to a point where they will pull the plug and walk away.

Users will experience lags when accessing inventories, freezes when customizing avatars, delays in the matchmaking process, and out-of-tune leaderboards that fail to keep track of performance.

Players demand a seamless playing experience and won't settle for anything less. And if they don't like it, they'll talk to their friends. These negative performance reviews can spread fast and will deflect potential players away from your game.

For game developers, the outcome can be catastrophic: a drop in DAUs or MAUs, a decrease in brand value, damaged brand reputation, plummeting profits, and dwindling market share. The gaming arena is cutthroat and players won't tolerate any shortcomings when it comes to real-time responsiveness.

This places great importance for game sessions to be processed in real time since they're at the heart of everything. Failing to do so will create a ripple effect that spreads out and causes havoc in all areas of performance.

So yes, real-time session stores are crucial to ramping up engagement levels and maximizing the playing experience in all areas. But achieving real-time session stores can be complicated because it requires a database to meet a long list of technical requirements.

The data technology requirements to pull off real-time game session stores

As a quick recap, players expect to be able to pick up from where they left off in each game and build upon their playing experience. Session stores are directly responsible for this since they provide databases with unique player data to carry out key game elements.

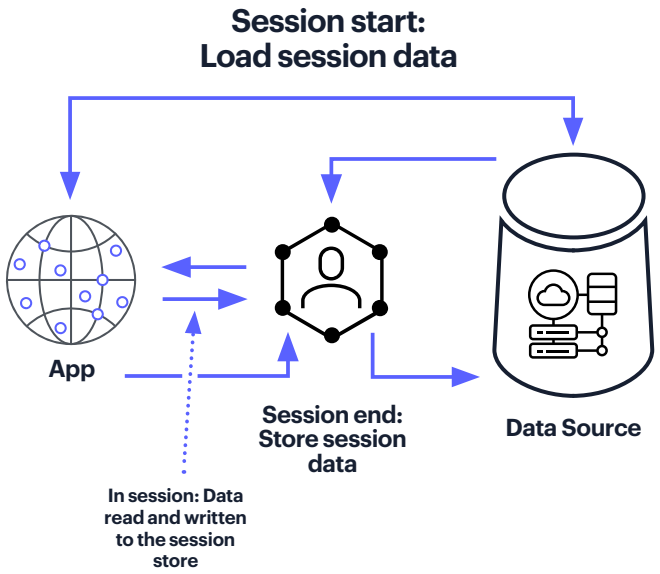
If session stores are inefficient, tensions will arise and players will be frustrated through long lobby waiting times, lags when accessing inventory, freezes in the customization process, stale leaderboard data, and more.

For a database to have the horsepower to serve all game features and have the latency for that 100ms speed, it needs to meet a specific list of technical requirements:

- **High database concurrency:** With potentially millions of players logged in at the same time, the database needs to be able to handle concurrent operations without hindering performance.
- **Low latency:** There will always be a certain amount of time taken up by data transfer and processing, which means your database needs to take up as little latency as possible for sessions to be processed in real time.
- **Persistent player stores:** Player data needs to be stored in persistent session stores that can scale with servers so that they're always accessible for personalizing, accessing inventory, matchmaking, and keeping track of player performances with leaderboards.
- **High write throughput:** Methods like batch processing create periods of stale session data that will affect in-game features. The latest player data (identity, credentials, inventory items, recent actions, etc.) need to be stored as they happen, which requires a high level of write throughput.
- **Scalability:** Whether it's thousands or millions of players playing online, each user needs to have the same experience, which means that your database needs to be able to scale whilst maintaining the performance necessary to make in-game features seamless.
- **Global synchronization:** Millions of players will simultaneously be accessing their session stores from around the globe, requiring the database to process this data consistently on a global level.

Real-time session stores need a real-time database

If your session stores are going to respond fast enough to feel no latency, then the entire process from request to processing to serving the data to players has to take place in less than 100ms.



Session store requires durability in addition to high availability

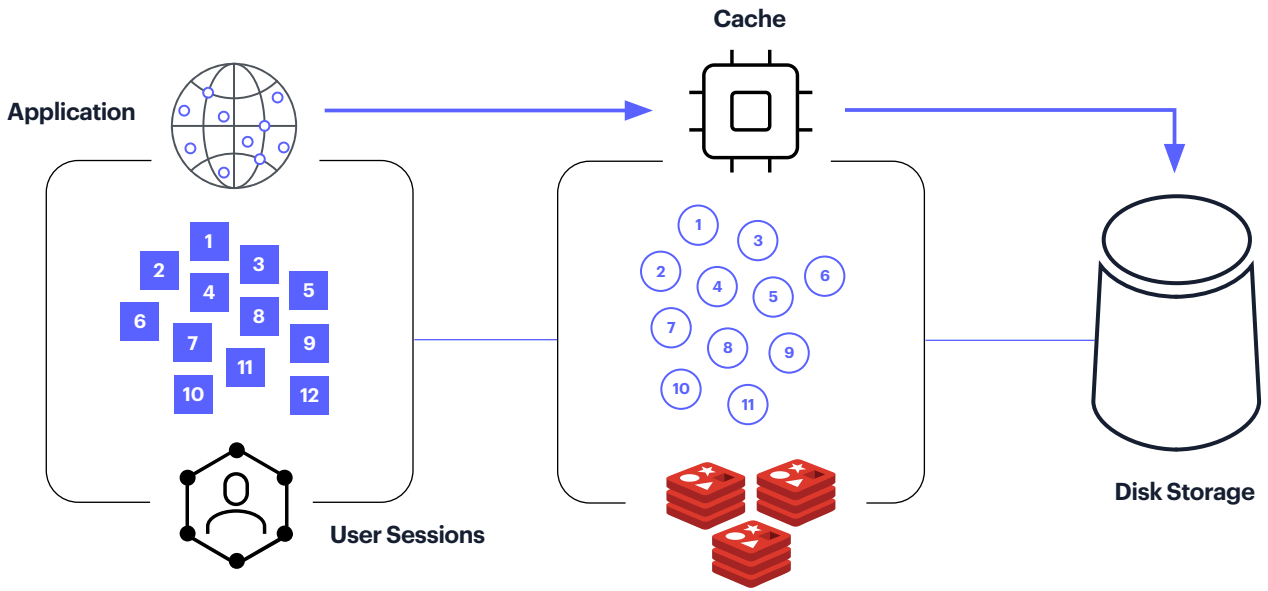
To maximize engagement, reading and writing session data at each user interaction must be done without damaging the user experience. This means that from the moment players log into your game, session requests must be sent, processed, and received within 100ms.

Unfortunately, this is quite a task for many databases since it can easily take 50ms for the game session request to move through the network, then another 50ms for server and infrastructure time, leaving somewhere between 0 and 1ms for databases to process sessions in real time.

Traditional databases simply aren't capable of latency that low which will inevitably impair performance across all levels. Only real-time databases are capable of having a <1ms latency and that's where Redis comes in.

With Redis being able to act as a cache as well as a session store, the architecture takes a unique shape, delivering both the high availability required for caching and session store scenarios as well as the durability needed for session store with in-memory application.

So by placing caching and session store requirements into one umbrella, Redis creates a more interconnected architecture where data transmission between different components becomes more seamless and in real time.



Here's how Redis Enterprise makes sessions fast:

- Redis's architecture allows dataset sizes to grow linearly and seamlessly without having to make changes to the application code.
- Redis provides multiple models of high availability and geographic distribution that allow local latencies for your users when needed.
- Multiple persistence options (AOF per write or per second and snapshots) that don't impact performance guarantee that you don't have to rebuild your database servers after failures.
- Support for extremely large datasets with the use of intelligent tiered access to memory (DRAM, persistent memory, or Flash) ensures that you can scale your data sets with the demands of your users without significantly impacting performance.

It's crucial that databases are able to provide real time speed for gamers across the globe and at any time. This must be carried out regardless of the number of sessions being processed simultaneously, whether that's thousands, hundreds of thousands or millions.

So in order to truly be a real-time database, the database needs the scalability and availability to provide low latency around the globe. To accomplish this, it should include features like native support for data models ideally suited to session stores, geo-duplication, automated resharding for scaling, 99.999% availability, and the ability to deploy in any environment where you need it.

Conclusion

Maximizing player engagement cannot happen without real-time session stores. Having them in real time allows developers to meet player expectations and create a more seamless playing experience that keeps users glued to their game.

This places developers in a tremendous position: higher engagement levels, enhanced brand value, increases in DAUs and MAUs, a boost in revenue, and an increase in market share.

Discover other ways a real-time database can level up your game and increase player engagement. Go to redis.com/gaming or download the *Level Up Your Gametech with a Real-time Database* white paper.

About Redis

Data is the lifeline of every business, and Redis helps organizations reimagine how fast they can process, analyze, make predictions, and take action on the data they generate. Redis provides a competitive edge to any business by delivering **open source** and **enterprise-grade** data platforms to power applications that drive real-time experiences at any scale. Developers rely on Redis to build performance, scalability, reliability, and security into their applications.

Born in the cloud-native era, Redis uniquely enables users to unify data across multi-cloud, hybrid, and global applications to maximize business potential. Learn how Redis can give you this edge at **redis.com**.