aws

Architecture Diagrams

Paginated Search with Purpose-Built Databases

Paginated Search with Purpose-Built Databases: Architecture Diagrams

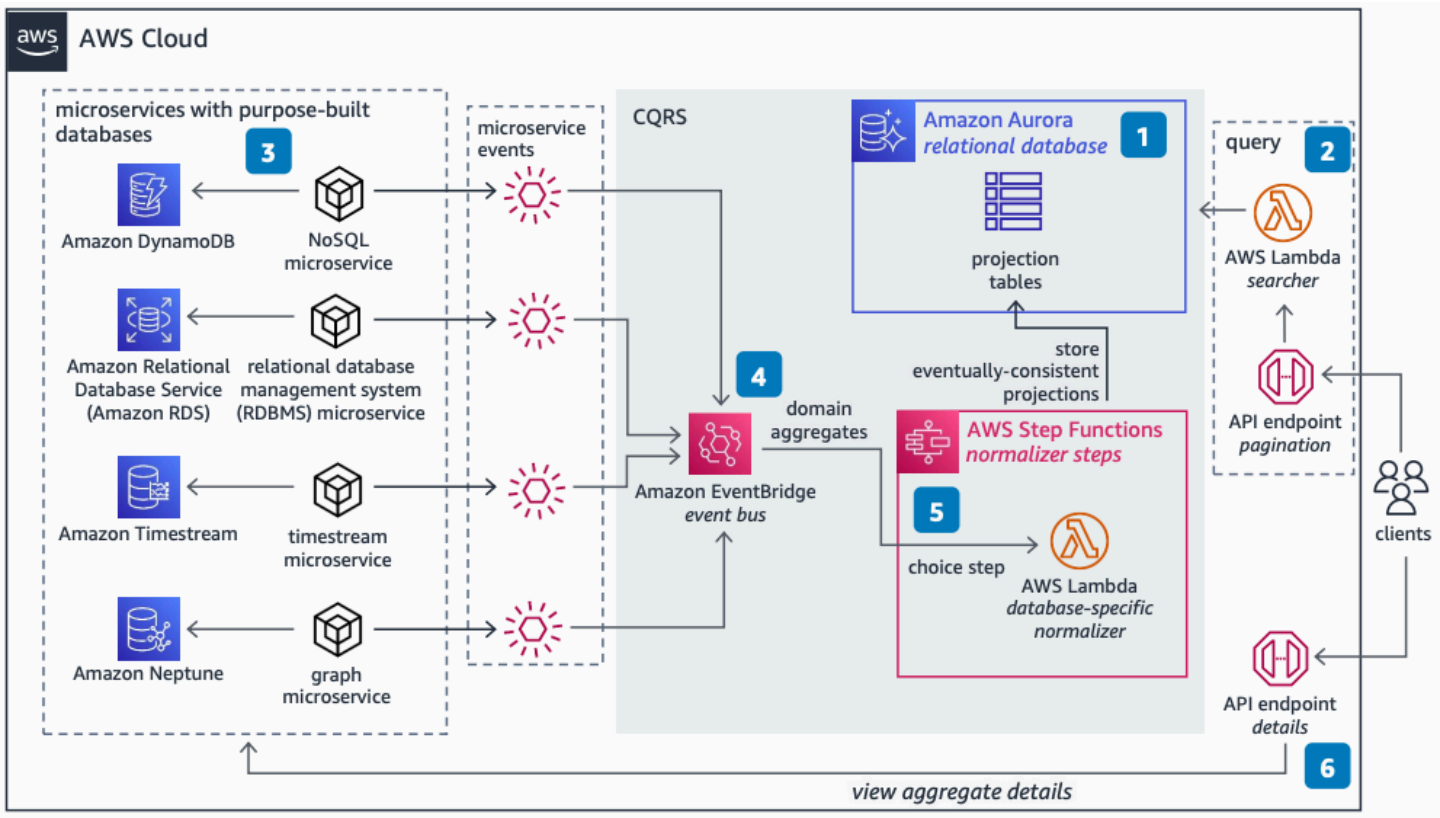
Table of Contents

Paginated Search with Purpose-Built Databases

Publication date: **August 25, 2022 (Diagram history)**

This architecture shows how to create a searchable, paginated list on domain aggregates from purpose-built databases with a step transformation and SQL queries. You can apply the command query responsibility segregation (CQRS) pattern to an event-driven microservice architecture. This enables searching, pagination, and sorting by querying eventually consistent data projections in Amazon Aurora.

Paginated Search with Purpose-Built Databases



The following steps describe the architecture:

1. Create a relational database using Amazon Aurora to hold projections of the domain aggregates you need to search, paginate, and sort. Work backwards from the user interfaces to design the tables and columns.

2. Create an AWS Lambda function to invoke a SQL query against the relational database, then connect it to your pagination API endpoint to handle search requests. The function uses a `Where` clause for searching, `OrderBy` for sorting, and `Limit`, `Offset`, and `Count` for pagination.
3. Decoupled microservices raise events for changes in the domain aggregates they own. Each microservice selects the purpose-built database that suits their use case, while eventually keeping the projection database up to date consistently.
4. Use Amazon EventBridge to create an event bus to collect the domain events from your microservices.
5. Normalize domain aggregates with database-specific normalizers to match the projection tables in the database and calculate additional analytics. To manage the workflow and direct domain aggregates to the right normalizer, create a choice step within AWS Step Functions. For complex normalizations, the state machine can integrate and wait for long-running containers on Amazon Elastic Container Service.
6. Clients can request details on a returned aggregate, referring to it by a unique key. The relevant microservice manages this through the API endpoint.

Further reading

For additional information, refer to the following resources:

- AWS Architecture Icons
- AWS Architecture Center
- AWS Well-Architected

Diagram history

To be notified about updates to this reference architecture diagram, subscribe to the RSS feed.

Change	Description	Date
Initial publication	Reference architecture diagram first published.	August 25, 2022

Note

To subscribe to RSS updates, you must have an RSS plugin enabled for the browser you are using.