Week 4 Notes and Resources

KEY TOPICS

Diving Deep and Best Practices

This week we look at concurrent access to tables using Optimistic Locking, Dive into Global and Local Secondary indexes, and then look at optimizing queries and schema design.

Optimistic Locking

Optimistic locking is a strategy to ensure that the client-side item that you are updating (or deleting) is the same as the item in DynamoDB. If you use this strategy, then your database writes are protected from being overwritten by the writes of others — and vice-versa.

With optimistic locking, each item has an attribute that acts as a version number. If you retrieve an item from a table, the application records the version number of that item. You can update the item, but only if the version number on the server side has not changed. If there is a version mismatch, it means that someone else has modified the item before you did; the update attempt fails, because you have a stale version of the item. If this happens, you simply try again by retrieving the item and then attempting to update it. Optimistic locking prevents you from accidentally overwriting changes that were made by others; it also prevents others from accidentally overwriting your changes.

LSI and GSI

Amazon DynamoDB provides fast access to items in a table by specifying primary key values. However, many applications might benefit from having one or more secondary (or alternate) keys available, to allow efficient access to data with attributes other than the primary key. To address this, you can create one or more secondary indexes on a table, and issue Query or Scan requests against these indexes.

A *secondary index* is a data structure that contains a subset of attributes from a table, along with an alternate key to support Query operations. You can retrieve data from the index using a Query, in much the same way as you use Query with a table. A table can have multiple secondary indexes, which gives your applications access to many different query patterns.

Amazon DynamoDB supports two types of secondary indexes:

- Global secondary index An index with a partition key and a sort key that can be different from those on the base table. A global secondary index is considered "global" because queries on the index can span all of the data in the base table, across all partitions. A global secondary index has no size limitations and has its own provisioned throughput settings for read and write activity that are separate from those of the table.

- Local secondary index An index that has the same partition key as the base table, but a different sort key. A local secondary index is "local" in the sense that every partition of a local secondary index is scoped to a base table partition that has the same partition key value. As a result, the total size of indexed items for any one partition key value can't exceed 10 GB. Also, a local secondary index shares provisioned throughput settings for read and write activity with the table it is indexing.

Writing Better Queries

In this section we discussed writing conditional expressions to improve query performance. You can use comparisons (such as *"a >b"*), functions (such as *"attribute_exists"*) and logical evaluations. Details can be found in the Developer Guide.

Single Table

Modeling data in DynamoDB is different than when modeling data in a relational database. For details on data models see this section of the Developer Guide.

Time Series

General design principles in DynamoDB recommend that you keep the number of tables you use to a minimum. For most applications, a single table is all you need. However, for time-series data, you can often best handle it by using one table per application per period.

Other Topics

There are other database options available in AWS.

- Amazon Aurora is a MySQL and PostgreSQL-compatible relational database built for the cloud, that combines the performance and availability of traditional enterprise databases with the simplicity and cost-effectiveness of open source databases.
- Amazon DocumentDB (with MongoDB compatibility) is a fast, scalable, highly available, and fully managed document database service that supports MongoDB workloads.
- Amazon Neptune is a fast, reliable, fully managed graph database service that makes it easy to build and run applications that work with highly connected datasets.
- Amazon Timestream is a fast, scalable, fully managed time series database service for IoT and operational applications that makes it easy to store and analyze trillions of events per day at 1/10th the cost of relational databases.
-

WHAT YOU ACCOMPLISHED THIS WEEK

- You continued to build out your application using conditional updates and transactions.
- You completed the course!

Please do complete the course survey! We hope you enjoyed the class.

LEARN MORE ABOUT AWS SERVICES

A variety of online learning resources are available. Here are a few recommendations to help you build on the knowledge you gained in this course:

- For unlimited access to foundational AWS training, visit our digital library at aws.amazon.com/training. We currently offer over 400 free online digital courses.
- AWS Training and Certification offers several courses that cover DynamoDB. You can access the courses at aws.amazon.com/training.
- Would you like to get AWS certified? Visit the AWS Certification page at aws.amazon.com/certification to get more information about exams, requirements, and prep courses.

Finally, remember to check back on edX.org for additional courses about AWS services and cloud skills.

Thanks for joining us! We look forward to seeing you in future AWS classes.