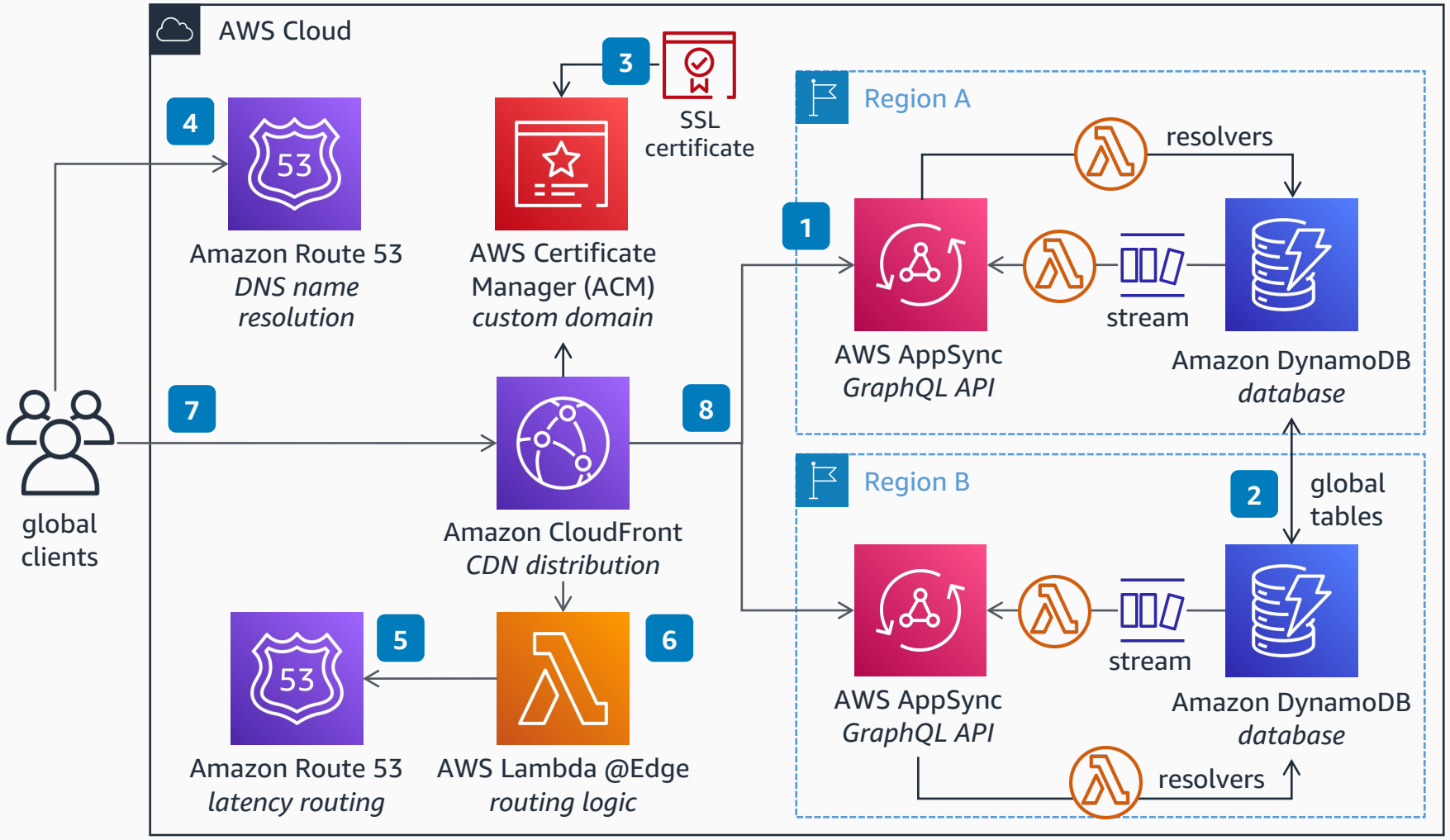
Multi-Region GraphQL API with CloudFront

Reduce latency for end users while increasing your application's availability by providing GraphQL API endpoints in multiple AWS Regions, with active/active real-time data synchronization supported by Amazon DynamoDB global tables.



1. Deploy a GraphQL API in two or more AWS Regions using **AWS AppSync**, then handle the **AppSync** commands and queries using **AWS Lambda** resolvers connected to an **Amazon DynamoDB** database.
2. To notify clients about data changes across all Regions, enable **DynamoDB** global tables to keep data in sync across Regions, then handle **DynamoDB** data streams with a **Lambda** handler, triggering purposely built GraphQL schema subscriptions.
3. To support custom domains, upload the domain's SSL Certificate into **ACM** and attach it to an **Amazon CloudFront** distribution.
4. Point your domain name to **CloudFront** by using **Amazon Route 53** as your DNS name resolution service.
5. Set up a routing rule on **Route 53** to route your global clients to the Region with less latency to their location.
6. So that your clients can authenticate seamlessly to **AppSync** endpoints in any Region, use **AWS Lambda @Edge** to query **Route 53** for the best Region to forward the request to, and to normalize authorization by abstracting the specificities of each Regional **AppSync**.
7. Clients across the globe can then connect to your GraphQL API on a single endpoint available in edge locations.
8. **CloudFront** will seamlessly route client's requests to the API in the Region with the lowest latency to the client's location.

Reviewed for technical accuracy April 25, 2022

AWS Reference Architecture