Scenario

Your company has developed a popular mobile app that offers coupons for registered local businesses within the user's vicinity. Discounts are **valid only for a limited time.** Users of this app can rate the coupon and provide comments. The application then aggregates this feedback along with usage data for each business.

A major sporting event is coming to the downtown area, and you need to ensure that the service can scale in a cost-efficient manner to deal with the increased volume in traffic during the event.

1. Ensure that the service will scale in a cost-efficient manner?
2. Prevent throttling?
3. Collect data (such as user feedback and app usage) in real time?
4. Let the local businesses view the feedback and usage data?

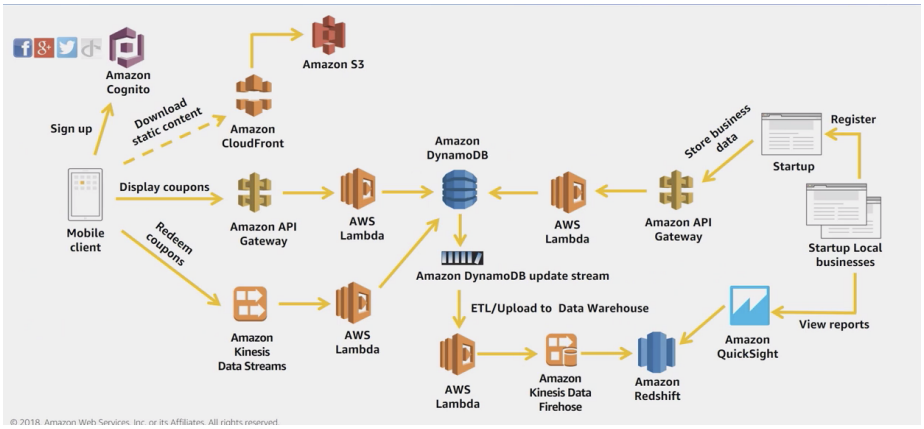

Figure 1: image

Registration process

Display Coupons

Why did we use DynamoDB?

- Is efficient for high-volume, high-velocity data
- Is fast and predictable
- Scales seamlessly
- Has no administrative overhead

Consistent Single-digit latency

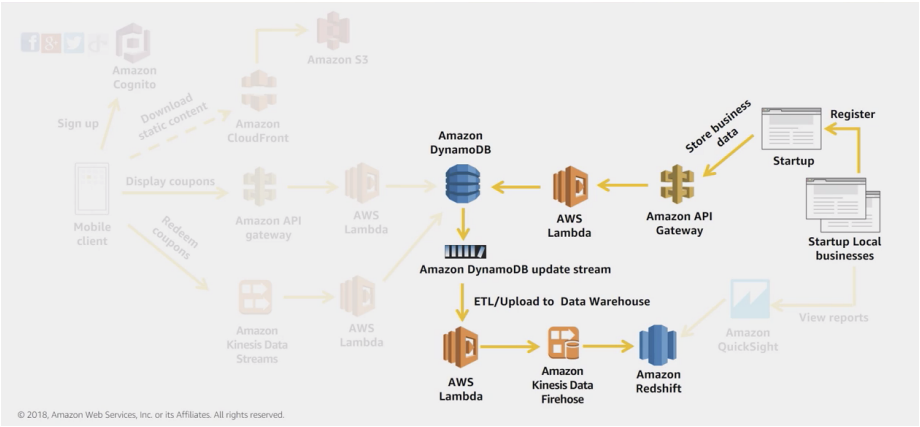

Figure 2: image

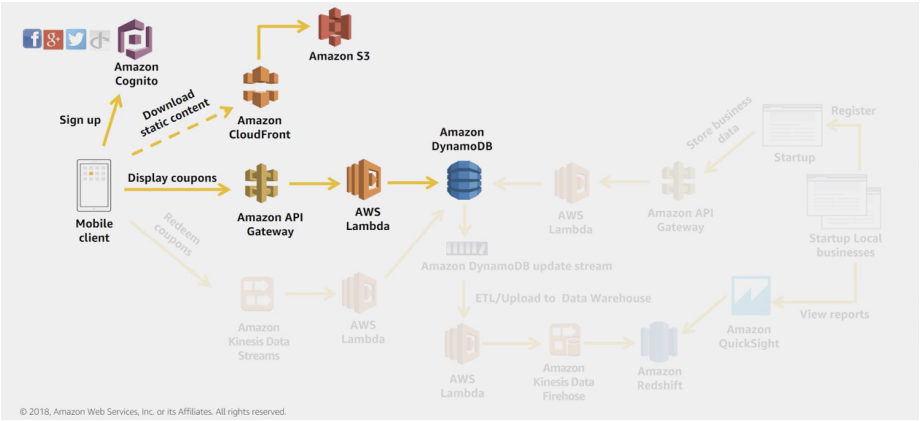

Figure 3: image

Relational vs. Non-relational databases

Performance of Relational Database depends on the hardware spec. To improve performance, you might move to a better server. => Scale up

Non-relational database is developed to scale-out!

DynamoDB

Table in DynamoDB is collection of items. Each item can have different attributes.

- Partition key: mandatory, key-value access pattern, and determines data distribution (input of hash function, output would be the physical position of the item)
- Sort key: optional, enables rich query capabilities, model 1:N relationships.

Maximum item size in DynamoDB is 400KB.

How to design primary key (based on the query access pattern)

Partition key with high-cardinality attribute or large number of distinct values. => Evenly spread out your data across partitions.

Commonly used sort key is... timestamp! If you query your data ranging from some time interval...

There are LSI, GSI

- GSI (alternate partition and/or sort key)
- GSI is global in sense the queries on the index can span all of the data in the base table across all partition
- Think of a GSI table as an partition table asynchronously populated by DynamoDB.
- If you only want to query your data using the base table's partition key (with alternate sort key), LSI is the best choice.
- LSI is local in a sense that data in the LSI is organized by the same partition key as the base table (but with different sort key)
- There is a size restriction in LSI. Maximum of 10GB per distinct partition key value. This is a restriction imposed on item collection. 10GB includes all of the items in base table + all of the items in the LSI
- You cannot delete the existing LSI table.

Large scale integration can be modeled as GSI

If the data size of item collection is bigger than 10GB, use GSI.

If the eventual consistency is OK, use GSI

If you want to query on a non-key attribute, use GSI.

Class Discussion 1

According to the scenario, discounts are only valid for a limited time. How could you implement a coupon expiration date?

A: DynamoDB TTL (per item)!

Class Discussion 2

You notice that as your DynamoDB tables grow in size, their response times is increasing. How can you prevent this from happening?

A: Scan operation would be slower. Use Query instead of Scan in your application. For query operation, the predicate should be imposed on partition key or sort key only.

Prevent Throttling

Per Table or GSI, 1 RCU -> one 4KB strongly consistent read. or two 4KB eventually consistent read.

RCU and WCU distributes evenly to every partition. There might be a hot partition issue which results in throttling. How to deal with it?

1. adding a prefix or suffix to the partition key that yields more partition key values.
2. But first, without knowing, the table might consume burst capacity
3. When your burst capacity runs out, it makes throttled requests. **Don't rely on burst capacity. Just provision sufficient throughput.**
4. DynamoDB Auto-scaling can be one option. + monitor throttled requests by CloudWatch
5. DAX is another good option when the operation is mainly read. **In-memory, write-through cache.**

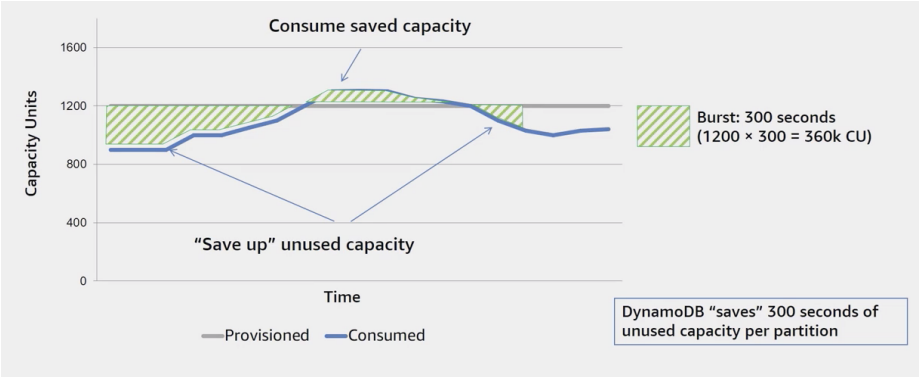

Figure 4: image

DAX (in conjunction with VPC)

DAX cluster consists of one or more nodes. DAX is designed to run within VPC. With new VPC endpoints for DynamoDB, DAX, or even lambda function in that VPC can easily access to your DynamoDB.

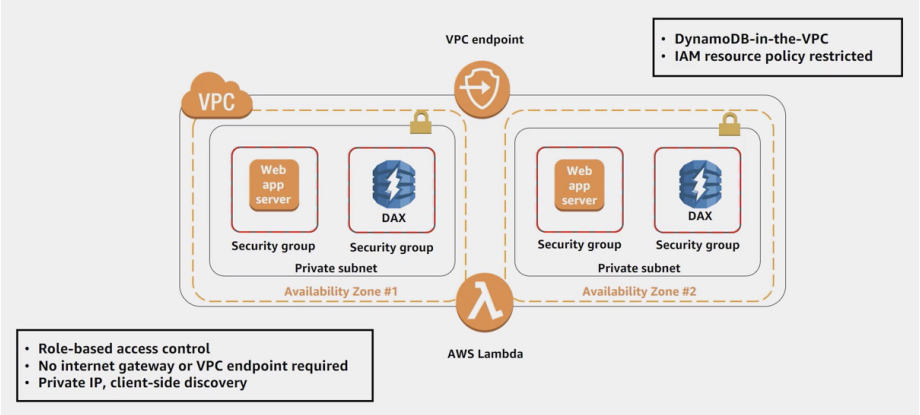

Figure 5: image

DAX and DynamoDB is two different product with different security models.

Class Discussion 3

What is the maximum throughput you can provision for a DynamoDB table?

A: No Theoretical Limit!

Class Discussion 4

What is the default consistency model for DAX?

A: Eventual consistency

Class Discussion 5

Can you utilize multiple DAX for the same DynamoDB table?

A: Yes. You can provision multiple DAX clusters. And they provide different endpoints, which can be used in different queries, enabling efficient query. Multiple DAX are independent, doesn't share anything.

Collecting Data real time (redeeming coupons)

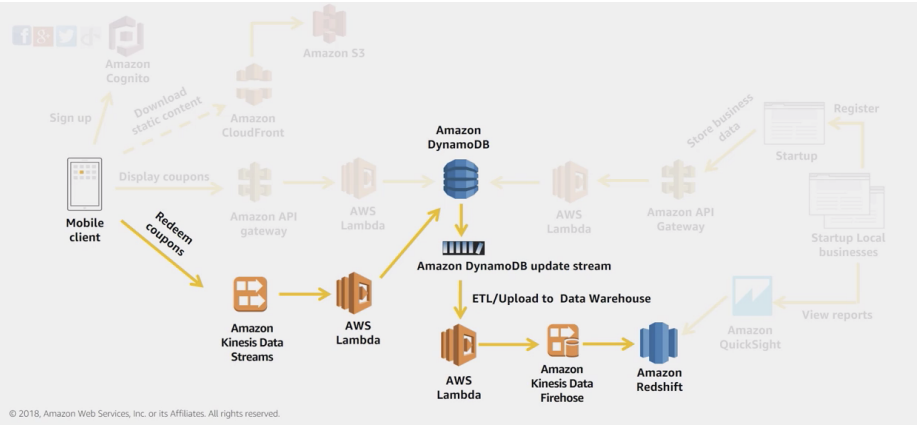

Figure 6: image

If you want to capture item-level modifications in the DynamoDB table, use DynamoDB streams!

- Ordered stream of item changes
- Exactly once, strictly ordered by key
- highly durable, scalable
- 24 hour of retention
- sub-second latency

It is a different service from DynamoDB !

Class Discussion 6

Can you configure a Kinesis Data Stream to be the source of multiple Kinesis Data Firehose delivery streams?

Yes. But note that the GetRecords call from Kinesis Stream can throttle

Class Discussion 7

Will preexisting data be encrypted when you turn on encryption for Amazon Kinesis Data Streams?

No. Preexisting data will not be encrypted.

Class Discussion 8

How can you keep your DynamoDB tables in sync across multiple AWS regions?

A: Global Table.

Class Discussion 9

How much data can you analyze with QuickSight?

A: Don't worry about it! It can scale seamlessly.

Class Discussion 10

Can you connect QuickSight to your Amazon EC2 or on-premises databases?

A: Yes you can! To do this, you need to add the QuickSight's IP address range to the authorized list (white list) in your hosted database

Sample Question 1

A customer is using Amazon Kinesis Data Firehose to put data into an Redshift cluster that is inside a VPC. The customer has made sure that the cluster is publicly accessible but cannot get the Firehose stream to copy the data to the cluster. To fix the problem, **you must unblock the Amazon Kinesis Data Firehose IP in the:**

A: Amazon Redshift cluster security group. Note that Redshift cluster NACL outbound group is a shitty answer, since it's a inbound problem.

Sample Question 2

The company is using DynamoDB to capture votes, location where the votes are coming form, and demographic data. The votes for each contestant are held in DynamoDB table with a partition key of contestant_id. The number of votes more than tripled, and performance degraded because of ProvisionedThroughputExcceededExceptions.

A: Shard write to the table by adding a hash to the contestant_id partition. (by adding some random code, allows multiple partitions for a single contestant)

Sample Question 3

Always put hot-small data to Redshift, cold-bulk of data to S3.