Distributed Database Management

# Module 04 – Schema

Jason Kendall – Professor Grinberg



So far, the premise of this course has been to demonstrate and introduce complex database systems designed to support businesses and organizations of varying sizes. Database systems have enabled these business entities to collect, manage, and analyze all the information that they take in during day to day operations. This is critical as in today’s world, businesses can use this stored data to make decisions regarding products or services for the betterment of the company. This would give any organization a competitive advantage over anyone not utilizing these technologies.

Distributed systems come into play adding a whole new dimension to data collection, storage, and delivery. Now, any data collected can be stored and then replicated in the cloud. This data can then be deployed to various regions ensuring the bulk of data is closer to the user. In the end, this means speed. A distributed database system like MongoDB can give a company like Ebay a critical speed advantage when it comes to enabling predictive search for its users. Because data is cached, response times are close to 500 milliseconds, becoming almost unnoticeable to the user. As the user searches for something they cannot find, MongoDB is able to use a cached prefix to provide a search example to the user. This one little action can mean the difference between the user finding the product or not. When a potential sale is on the line, Ebay wants to ensure its users can find what they are looking for. Without the technologies’ that MongoDB and other distributed databases bring to the table, Ebay could lose out on millions of dollars in sales per year.

Now that we understand the true power of the databases we are dealing with; we can attempt to shed some light on a “real world” exercise. The sample restaurants database is growing as the business thrives. More and more users are visiting the restaurant and the company needs to increase its access to a list of active orders and the ingredients they contain. The company is also requiring live inventory reporting to include monthly analytics regarding the frequency of use ingredients.



As this data comes in, we can see that the information stored in the data warehouse can provide reporting. Using JavaScript allows us to take action against that data. With MongoDB allowing us to scale the company’s data to greater than petabyte size we can ensure that the restaurant does not outgrow its technology stack for the foreseeable future. This takes care of one problem for us but leaves us with another.

To gain access to the live orders and the ingredients in them we can write code that allows us to return this information in key value order. The JavaScript function that would return this data for us looks something like this:

db.restaurants.find({“name”: “Chilis”}, {“orders.active.ingredients”: “steak”}).limit(5)

This query would target the specific restaurant, and then look in the active orders for a specific ingredient. In this case we used the keyword steak. Since we have constant access to our inventory, it is considered live and the reports on the ingredients are constantly updated as needed.

To expand the area of the search we would create a 2d sphere based around the ingredients. To build the 2d index we would go through the following code to setup our database and define a location for our inventory search for the different restaurants in our region.

db.<restaurants>.createIndex( { <location field> : "2d" ,

<additional field> : <value> } ,

{ <index-specification options> } )

We can even implement code designed to find the top selling orders per month and then return this information to the analytics team for the restaurant. They can then use this information to see what is selling the best and what is not. With this information they could make cuts to items that are not selling at all. The code to aggregate orders might look something like this:

db.restaurants.orders.aggregate(

{ $match : { "order\_line.ol\_i\_id" : 531 } } } },

{ $project : { "order\_line" : 1, \_id:0 } },

{ $unwind : "$order\_line"},

{ $match : { "order\_line.ol\_i\_id":{"$ne": 531 } } },

{ $group : { \_id : "$order\_line.ol\_i\_id",

totalSales : { $sum : "$order\_line.ol\_qty" } } },

{ $sort : { totalSales : -1 } },

{ $limit : 5 }

);

Once the database and system are setup, and now that we can find all the active orders and their ingredients, we can utilize MongoDB cloud manager to gain a bird’s eye view of the whole operation. After we have coded our database, we can then use our variables such as “ingredients” with the cloud management features enabled by MongoDB.



The cloud manager can allow us to run analytics against the entire database system at all times. This will provide our restaurant the ability to make predictions about products and sales. This would also give us the ability to ensure our database is running smoothly and can handle the amount of information we are taking in and pushing out. This type of technology is providing an advantage that has never been seen before. The tech industry was considered second class and isolated a few years ago but slowly, as the technology evolves, more and more businesses are able to leverage and utilize the assets tech can provide. Now technology pervades almost every sector, from finance, to hospitality, to the medical field. Data is growing at an alarming rate, so much so that I would consider it the economic resource of the future. Understanding database technologies is something this course has taught me, and I am very thankful for it.

Resources:

<https://docs.mongodb.com/manual/indexes/>

<https://docs.mongodb.com/manual/core/2dsphere/>

<https://docs.mongodb.com/manual/aggregation/>

<https://en.wikipedia.org/wiki/Analytics>

<https://www.geeksforgeeks.org/distributed-database-system/>