ANALYSIS OF MIGRATION TO NOSQL DATABASES IN WEATHER CHANNEL

Venu Dodda

KENT STATE UNIVERSITY

"The Weather Channel turned to MongoDB to get killer features out to users quickly. Changes that used to take weeks can now be pushed out in hours"

Scenario:

The main goal of the project is to conceive of a hypothetical situation when a non-relational database would be the most practical choice. To justify this, I have taken the real-time example of the company which faced multiple issues using relational databases and later switched to a No-SQL database (MongoDB) to resolve the issue. Details of the company are listed below:

- Industry: Information Technology of Weather Forecasting.
- Firm: Weather Channel.
- Information Requirements: To solve this problem I have collected the weather forecast dataset from the Kaggle website and used this data for further analysis.
- Data requirements: My project needed data related to every data weather forecast from different countries. I have considered factors like Mean temperature, maximum temperature minimum temperature, mean station pressure, and Mean Sea level pressure, etc. With this information, the company switched to the no-SQL database to achieve the industry's needs.

1. **A discussion describing the form that the NoSQL DB should take (including choosing a specific DB). You should consider other options in your discussion.**

   Problem:

   Rapid weather change is common. In its most extreme forms, it is erratic, occasionally hazardous, and frequently exciting. People are eager to get their hands on the most recent knowledge because of how it affects their way of life. They demand to know the situation immediately.

   To satisfy the need for continuous, real-time weather information, The Weather Channel launched a 24x7 television network in 1982. A few years later, they took the next logical step online with weather.com.

   However, creating mobile apps was challenging due to the site's clumsy relational database backend. With responsive apps and a scalable framework, the Weather Channel team needed to iterate more quickly.

   The Weather Channel brand needs to upgrade from a traditional relational database architecture because of its 40 million-strong and rapidly expanding smartphone user base.

To quickly release revolutionary features to users, The Weather Channel changed to the No-SQL database MongoDB. Changes that once required weeks to implement can now be made in a matter of hours.

The main reason to choose MongoDB is that relational databases use tables to store the data. The analysis was almost done on the tables. Some of the analysis was based on multiple tables so the retrieval of the data is difficult. It takes time to process. In MongoDB, the data was stored in the form of documents (JSON). So that access to the data and analysis of the data would be easier with JSON objects.

2. **A discussion of why your non-relational solution would work best – a key grading point is justifying your choice (business issues such as cost may also contribute to your decision in addition to technical requirements). Describe the specific characteristics of your scenario that suggests your NoSQL solution is the best choice. Describing why it can be utilized is important but insufficient; convince me that it should be the solution – not just that it can be the solution.**

The weather channel company had addressed the following issues and below factors were considered:

- **"High costs and complexity with simplified scale and speed"**

  High expenses and complexity have been substituted by streamlined size and speed. They are moving news, lifestyle, and some weather content from their web properties to MongoDB now that they have updated on a cloud platform.

  Users may customize their experiences across mobile devices, tablets, and the website thanks to a fleet of apps built on MongoDB. They have access to extremely quick radar maps and immediate severe weather alerts. The Weather Channel is prepared to provide whatever its user need.

- **"Severe weather alerts, faster than the storm"**

  The Weather Channel's severe weather alert system is used by five million viewers. For many users, it's a necessary feature and a competitive differentiation for the company.
  The Weather Channel is required to immediately inform those 125,000 local customers of any storm warnings issued for Cook County, Illinois, by the National Weather Service (NWS).

- **"With MongoDB, The Weather Channel can quickly distribute those weather alerts to subscribers in affected geographic locations in real-time"**

  Kolin asserts that MongoDB is the only product that, thanks to its secondary indexes and quick ad hoc querying, can dependably carry out that kind of lookup on such a massive user base in a matter of seconds.

- **“Simplified scale in the cloud”**

  Weather can be unpredictable. The internet usage of weather applications is also increasing.
  The Weather Channel can relax about app performance even during erratic peak periods. In a normal minute, the apps can handle two million requests, including those for weather information and social sign-ins. MongoDB will expand along with the user base. Thousands of nodes, petabytes of data, and hundreds of thousands of operations per second may all be supported by MongoDB thanks to its natural scale-out features.

  At first, The Weather Channel intended to develop its own management tools for the brand-new cloud infrastructure. Instead, they utilized Cloud Manager, a management tool for MongoDB, and they saved a considerable amount of time and money. Cloud Manager is a cloud service that makes it simple to run MongoDB at any size. It was created by the same people who create MongoDB. The Weather Channel can prevent problems from happening by using features like performance analysis, customized alarms, and point-in-time recovery. This allows them to optimize its MongoDB implementation.

- **“Fast apps, without the wait”**

  The Weather Channel brand's requirements were immediately satisfied by MongoDB, necessitating no substantial optimization. MongoDB was designed with this task in mind. The Weather Channel crew can iterate quickly nowadays without being concerned about schema modifications. They are adaptable. They can deliver updates to people in a much shorter amount of time. And at a much lesser price.
  New technologies, new features, and new expectations. Users demand amazing apps that are always improving. And now, MongoDB supports The Weather Channel in its delivery.

Conclusion

The project aims to Imagine a hypothetical situation when a non-relational database would be the most practical choice. I have taken the real-time example of an Information Technology of Weather Forecasting (Weather Channel) company that faced multiple issues like Rapid changes in weather and wanted to provide Real Time Updates to People. They demand to know the situation immediately. To satisfy the need for continuous, real-time weather information, The Weather Channel launched a 24x7 television network in 1982. A few years later, they took the next logical step online with weather.com. Creating mobile apps was challenging due to the site's clumsy relational database backend. With responsive apps and a scalable framework, the Weather Channel team needed to iterate more quickly. MongoDB, a No-SQL database, was adopted by The Weather Channel in order to deliver ground-breaking innovations to users more quickly. Previously taking weeks to complete, changes can now be completed in a few hours.

References:

1. **https://www.kaggle.com/datasets/twtdata/weather-dataset.**
2. **https://hevodata.com/learn/mongodb-use-case/#t7**
3. **https://www.mongodb.com/use-cases/analytics**
4. **https://www.researchgate.net/publication/278302676_A_Comparative_Study_MongoDB_vs_MySQL.**