Enterprise Computing and Business CS3525

Group Assignment - Data Models and Warehousing

51549217

Introduction

This report outlines the data models and warehousing used for our business case of OrderShare. OrderShare is an innovative application which, allows customers to order food together with others, in order to save on transportation fees, and make the best use of deals. The system includes features such as creating an order, 'subscribing' to a restaurant, receiving notifications for nearby orders and adding friends to a specific order. The business case of the application is to take a share of each transaction, or be funded by restaurants promoting themselves in the home page or subscriptions. The main focus of the application is mobile, so much of the heavy lifting must be done server side, and delivered to the client via API calls.

Purpose

To handle such a complex system, we must design a robust data model. While the codebase can be changed, adjusting data models retroactively is no easy task. As Bloomberg explains in their professional blog (BloomBerg, 2015), the data lifecycle has 7 phases: Data Capture, Data Maintenance, Data Synthesis, Data Usage, Data Publication, Data Archival and Data Purging. At each stage of the data lifecycle, there are multiple occasions when the system communicates with the data model. Therefore, the flow of data must be streamlined to minimize system overhead and maximize efficiency.

Planning

To build a data model one must follow the Data Modelling Development Cycle (Learn Data Modeling, 2015). Initially, some planning must be made to determine the feasibility of the system, including estimating completion times and staffing. Thereafter, actual development begins by gathering business requirements. In the case of OrderShare, we will gain insight from business analysts who are experts in the restaurant and delivery industry. For example, we can see what kind of market research Deliveroo has completed, and mirror their advancements. Then we will interview end users and find out the needs of a variety of customers. The main groups include students, work colleagues, attendees of a party and residents of apartment buildings.

Methodologies

Due to the scope of this report and the advice of professors, the main focus will now be set to Conceptual Data Modelling. Therefore, further topics such as LDM (Logical Data Modelling), PDM (Physical Data Modelling) and building the database will not be discussed.

The purpose of the conceptual model is to "confirm the scope of subject areas and their relationships" (Kendle, 2005). Two approaches of initiating data modelling are strategic data modelling and data modelling during systems analysis. As the Zachman Framework for Enterprise Architecture explains (Finkelstein, 2007), strategic data modelling focuses on creating an overall structure for the data model. This includes creating a framework which, enables addition of new models and sets requirements for them. In the case of OrderShare, these models could include: Restaurant, Customer, Order and Menu. Conversely, data modelling during systems analysis focuses on identifying information that a lone system requires to function inside a framework. Essentially this occurs when a new model is created and implemented into the system.

For OrderShare, we plan on using strategic data modelling to create a minimal framework, and then use modelling during systems analysis to implement refined models. An alternative approach would be to only use modelling during systems analysis which, would result in multiple detailed models that have clearly specified inputs and outputs. In a system with complex or relatively independent components such a methodology would be optimal. However, in the case of OrderShare, the models are tightly coupled with the business logic so the first option remains the most efficient.

There are two ways to approach the creation of data models: Top-down and Bottom-up (Morris, 2016). The Top-down model begins with a general idea and expands by narrowing down into the details of the system. The resulting abstract model is created by experts, and it can be used as a reference point for future development. The Bottom-up method starts by looking at the details of the system like the leaves of a tree, and then it moves 'upwards' into larger concepts. This is useful when existing data models and structures are being re-built to serve new business requirements. For the purposes of OrderShare, we believe that a Top-down method best meets our needs because we are building a system from scratch, with many uncertainties in the specifics of data relationships.

Conceptual Model

Using the previously outlined methodologies, we created a simple Entity-relationship (ER) model for OrderShare. We chose the ER model as our initial 'framework' because of how effectively it translated our business models into data models. A relational approach is easy to understand and implement. Another option would have been to use a NoSQL or "non relational" approach, but its benefits of scalability are not necessary at the conceptual stage.

In following the Top-down method we began by creating the core model of the business: the restaurant and continued by adding subsequent models. The Bachman notation allowed us to create relationships without being hindered by cardinality and database constraints.

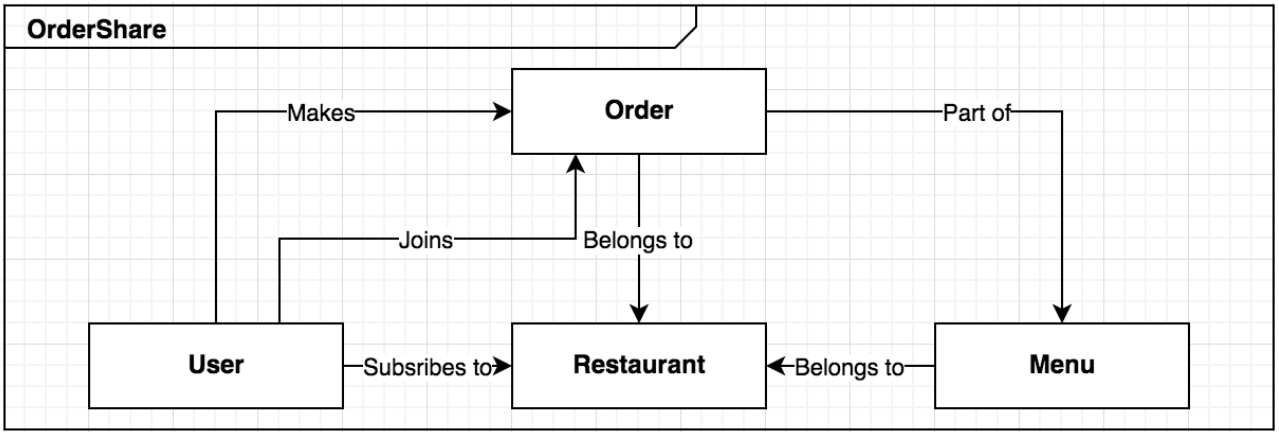

Figure 1. Initial ER model of the OrderShare Data Model

Data Warehousing

Storing large amounts of data is central to OrderShare's business model. The application will be API based for both web and mobile platforms. Thus, data must be stored in a centralised server, preferably in the cloud. While remote servers have their own weaknesses, once properly setup their benefits outweigh the risks. Cloud services have massive benefits in scaling, maintainability and reliability. For example, Amazon Web Services (hereafter, AWS) allows nodes to be added to server clusters instantaneously, in addition to high-quality load balancers to handle high traffic. It also has the advantage of locality – "AWS cloud spans 52 availability zones within 18 geographical regions around the world" (Amazon, n/a). This allows clients around the world to be served with low latency. While services such as Google Cloud and Azure hold their own, AWS is still considered king of the cloud (Finnegan, 2017).

To comply with UK law (Data Protection Act, 1998), the user data stored in OrderShare servers must be under heavy scrutiny in regards to access privileges and security. AWS is adamant in keeping security its number one priority, proven by their ISO 27001 standard. With a magnitude of security tools available, they have managed to keep their leading position in the Cloud market with a market share of over 30% (Miller, 2017). Accordingly, AWS was chosen to power OrderShare.

Other factors that increase security are continuous integration (hereafter, CI) tools. CI tools allow systems to be deployed and setup graphically, without the need of root SSH, which, is unadvisable in a production environment. Moreover, modern CI's offer automated testing, build scripts, AWS, backups and aid in the use of Docker images (Clearscale, 2016). With the three pipelines of development, staging and production, customer data is out of reach of developers, and safe from volatile migrations. The use of Docker images also allows the system to be refreshed into previous states, which, is especially beneficial for recovering the system after a failure.

Redundancy is a crucial factor in respect to data warehousing. If a company were to host its own servers, it would likely use a RAID or similar mirroring technology that spreads data out to multiple hard drives. The data might be stored in separate server closets, but would likely reside in the same physical room. AWS takes this a step further, as it not only stores data across disks and servers, but across geographical locations (AWS, n/a). With such a zone dependency setup, the system becomes practically fault tolerant.

The scalability of a system becomes essential when users begin to search through extensive amounts of data simultaneously. Fortunately, there are tools to help. One of them is ElasticSearch, a search engine that aids in the retrieval of large data sets quickly. It divides data into distributed shards, which, innately coordinate and delegate the search onwards allowing near real-time search. Another impressive toolset is the Apache software ecosystem, featuring tools such as: Kafka for data feeds (useful for current orders), Hadoop for distributed storage, HBase for non-relational data and ZooKeeper for distributed synchronization. However, one must remember that these tools work best when combined with an efficient data model. A poor data model cannot be saved by excellent tooling.

The next step is to ensure data is cached efficiently. Even with small databases, queries can extend load times and put increase server load. Services such as MemCached and Redis automatically cache popular searches to memory, which queries significantly faster than a database. Companies like Facebook and Google even store substantial amounts of data into local memory to optimise page load times.

Conclusion

In conclusion, this report has outlined the approach OrderShare will take towards enterprise data modelling and data warehousing on a conceptual level. The next steps for data modelling will include creating a logical entity-relationship model with well-thought out attributes. Once Physical Data Modelling is complete, the database can be optimized with normalization, preferably in the third normal form. In regards to data warehousing, specific technologies such as SQL and the Bamboo CI may be considered.

References

Amazon. (n/a, n/a n/a). *Amazon AWS*. Retrieved February 12, 2018, from Amazon AWS: https://aws.amazon.com/about-aws/global-infrastructure/

AWS. (n/a, n/a n/a). *AWS*. Retrieved February 12, 2018, from AWS: https://media.amazonwebservices.com/architecturecenter/AWS_ac_ra_ftha_04.pdf

BloomBerg. (2015, 7 14). *BloomBerg Professional Services*. Retrieved 02 11, 2018, from BloomBerg Professional Services: https://www.bloomberg.com/professional/blog/7-phases-of-a-data-life-cycle/

Clearscale. (2016, September 23). *Clearscale*. Retrieved February 12, 2018, from Clearscale: http://www.clearscale.com/blog/the-importance-of-continuous-development-and-continuous-integration/

Finkelstein, C. (2007). Strategic Models for Rapid Delivery of Enterprise Architecture. *Methods & Tools* (Spring), 13-30.

Finnegan, M. (2017, September 20). *ComputerWorldUk*. Retrieved February 12, 2018, from ComputerWorldUk: https://www.computerworlduk.com/it-vendors/microsoft-azure-vs-amazon-aws-public-cloud-comparison-which-cloud-is-best-for-enterprise-3624848/

Kendle, N. (2005, July 1). *tdan*. Retrieved February 11, 2018, from The Data Administration Newsletter: http://tdan.com/the-enterprise-data-model/5205

Learn Data Modeling. (2015, June 19). *Learn Data Modeling*. Retrieved February 11, 2018, from learndatamodeling.com: http://learndatamodeling.com/blog/data-modeling-development-cycle/

Management, D. (Database Management, Database Management Database Management). *Database Management*. Retrieved Database Management Database Management, Database Management, from Database Management: Database Management

Miller, R. (2017, July 28). *TechCrunch*. Retrieved February 12, 2018, from TechCrunch: https://techcrunch.com/2017/07/28/aws-wont-be-ceding-its-massive-market-share-lead-anytime-soon/

Morris, R. (2016, January 1). *Database Management*. Retrieved February 11, 2018, from Database Management: http://databasemanagement.wikia.com/wiki/Database_Design_Strategies

Parlament, U. K. (1998, July 16). *Legislation*. Retrieved February 11, 2018, from Legislation.gov: http://www.legislation.gov.uk/ukpga/1998/29/contents