Oder Cache using Microsoft Azure

IOP Technologies LLP
February 2018

1. Abstract

The e-commerce system using traditional RDBMS with multiples of GBs, TBs of transaction data encounters serious performance problems, latency issues for user or application accesses and to process them in real-time.

This white paper details around the Order data cache mechanism using Microsoft Azure Data Lake to address the availability, latency and better performance of the system.

2. The Problem

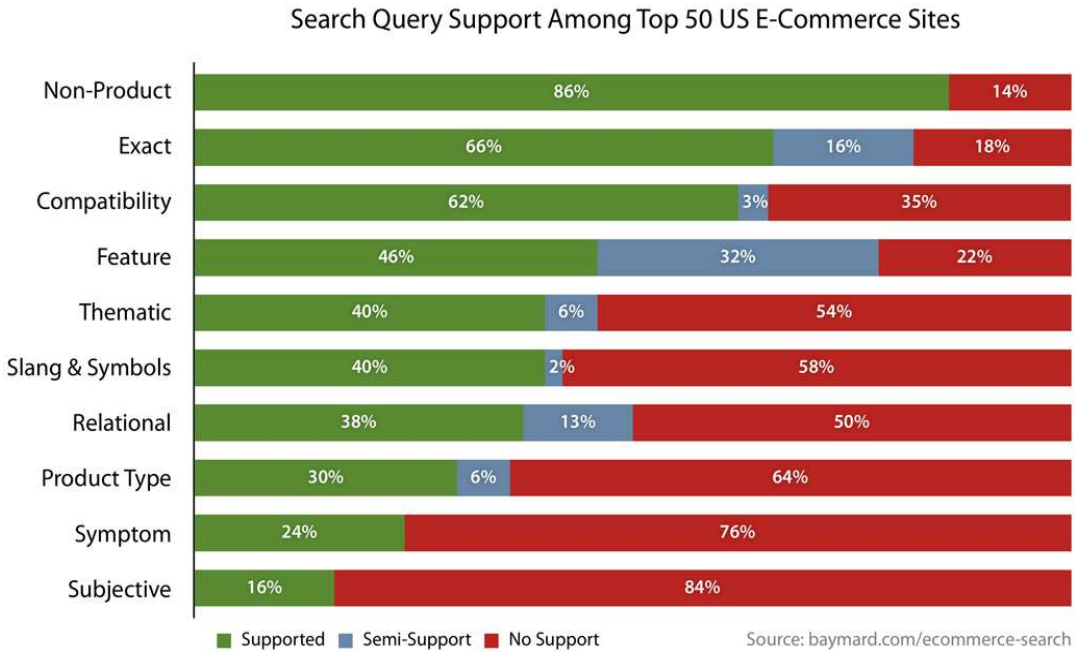
E-commerce systems, for instance an orders processing system, built using traditional RDBMS system does not scale pretty well when the data grows into 10s of GBs. Even commercial RDBMS systems starts breaking apart with data sizes of 100s of GBs. Operating cost and man hours spent in operating the database layers is also very high. We have seen customers observing high read and write latencies and often downtime on the database layer. This is highly undesirable and service downtime often leads to revenue losses and high customer churn.

3. Usability studies

During usability study, it was observed that most of tested sites even big e-commerce giants had poor response time for order queries. Benchmark results reveal surprisingly dismal support for essential e-commerce search query types. For example, among the top grossing US e-commerce sites:

- 16% of the e-commerce sites do not support that users search by product names (which appear on the product page)
- 18% handle phonetic misspellings so poorly that users will have to pass a spelling test to be presented with results (e.g. 0 results for “Kitchen Aid Artysan” when looking for the “Kitchen Aid Artisan” mixer)
- 70% require users to search by the exact same product type jargon the site uses, failing to return relevant products for a search such as “blow dryer” if “hair dryer” is used on the site, or “multifunction printer” vs “all-in-one printer”
- 22% of the sites don’t support search queries for a color variation (despite the product searched for being available in multiple colors)
- 60% don’t support thematic search queries such as “spring jacket” or “office chair”

- 84% don't handle queries that specify a subjective qualifier, such as "cheap" or "high quality"
- 60% don't support symbols and abbreviations, resulting in users missing out on perfectly relevant products if searching for inch when the site has used " or in



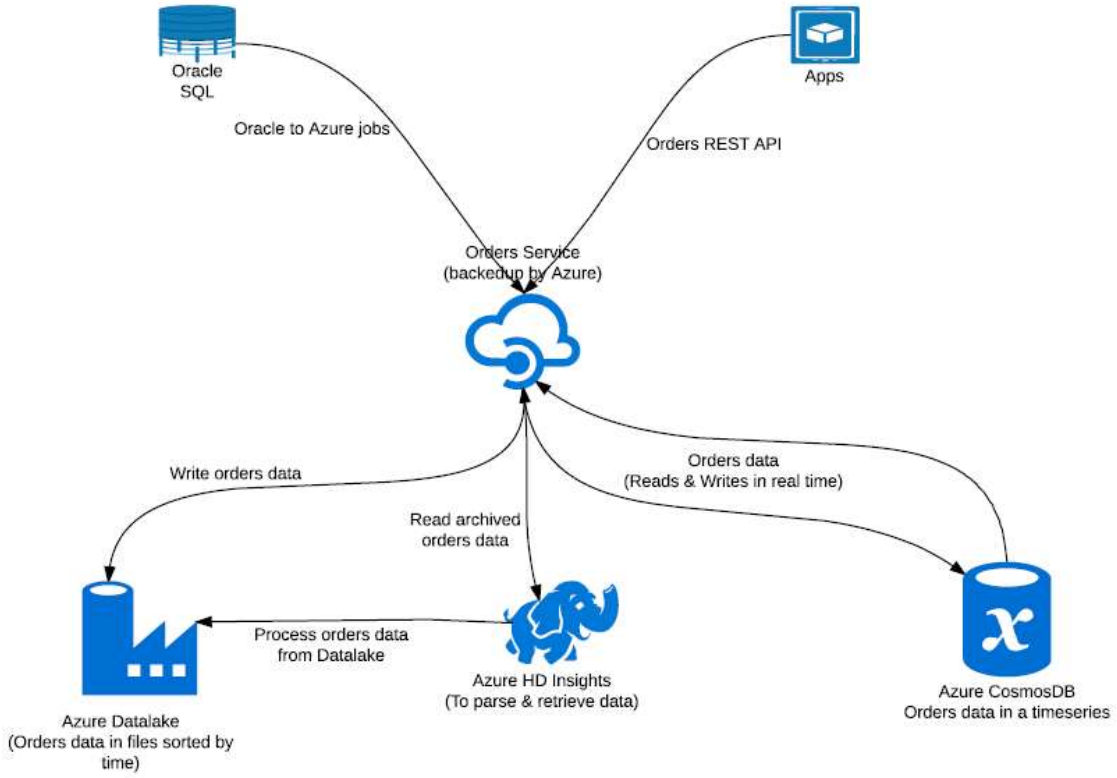
4. The Basic Solution

Above problems can be addressed in a scalable and efficient way using cloud systems. Cloud systems ensure high availability of the solution and help you to scale across the globe very efficiently in quick time. Apart from solving these critical issues, cloud architecture encourages collection of more data and helps you to innovate and deliver awesome customer experiences via AI and Machine Learning.

In this blog, we describe a order processing solution built using the Microsoft Azure cloud infrastructure. Microsoft Azure components can scale up/down automatically on demand across geography depending on the load and usage/throughput. Microsoft Azure Cosmos DB and Data lake store are built for handling high volumes of data. For instance, Data lake store can be designed to handle PBs of data. In specific, we address different types of search queries identified during a large-scale usability study of e-commerce search. While not exhaustive they reflect the main types of queries that users rely on when searching in an e-commerce context. Both live and offline migration of the data is also dealt very effectively on the Azure cloud.

5. Microsoft Azure Architecture

Solution exposes set of REST APIs to be utilized by the current and any future application tiers for order processing. These REST APIs hides all of dual writes/reads and other data migration complexities from the client applications. Apart from these, it is easy to monitor these APIs and in future, additional features can be implemented very quickly. APIs are served by components running in a micro-service fashion thereby ensuring easy extensibility. Azure API management readily serves above needs.



Below we have described in detail the two APIs - PutOrders and GetOrders required for the serving a orders micro-service. Before we dive into these APIs, a quick data driven thought process put into the architecture design. User always queries for the latest orders (typically few quarters) and it is imperative to serve the latest orders in real time (milli seconds SLA). Older orders are not looked up often and a little bit more time in serving this data is acceptable (few secs SLA).

6. Microsoft Azure Components

All components are deployed in more than 3 regions for high availability.

API management layer

API management layer predominantly serves the following functionality. API management layer also gives the capability to implement any additional security, metering or any complicated caching features in the future. The API management layer is stateless and no data is stored permanently.

a) Authenticating the client apps
b) Caching orders query responses from databases
c) Paginate orders query response
d) Dual read and write capability during migration
e) Routing to nearest region Azure component (Cosmos or HDInsights)

Azure Cosmos DB

Azure cosmos DB stores all of the recent orders data and is used as the first point to read the data. Orders data are stored in a user timeline table on a Cassandra schema. The userId and the month is used as the primary key and within the key, order ids are stored as values across sorted timestamp. Orders data are stored in a separate Cassandra table. An offline purge job is implemented to periodically archive the older data. Time for purging the older data is decided based on the data size and the deployments. The time frame can be modified later depending on the actual traffic.

Azure Data Lake store

Azure Data Lake store holds all of the orders data indefinitely. Data store is built to store PBs of data and the solution will use it as the big data store. The data is organized by date and user id. Azure HD Insights cluster processes the orders data from the given date and optionally user id and outputs the required information to the API layer.

Azure HD Insights

Spark jobs are triggered on the Azure HD Insights cluster on demand to fetch the orders data for a given user and a time. Further HD Insights will be used to do any batch processing across a huge volume of data (say a month, quarter or a year across all users).

7. APIs

Put Orders API

Put orders API does the following sequence:

- Authenticates the client app.
- Validates the incoming http request.
- Data write
- Creates the order id on the orders Cassandra table in the cosmos db.
- Creates/updates the user timeline with the order id on the User-Order Cassandra table in cosmos DB.
- Creates the folder based on the date and user in Azure Data store factory.
- Updates the data onto the required file inside the folder.

Get Orders API

Get orders API does the following sequence:

- Authenticates the client application

- Validates the incoming http request
- Data read
- Query the user-order cosmos table with the date as the range. If data found, query the orders table for each order id returned in the above step
- If no data found, trigger a spark job on the HD Insights cluster with the date and user id. Send back the job id back to the client. HD Insights response will be cached and stored in a separate table temporarily
- An async call-back is done to the client as soon as the HDInsights data is available. Clients can also poll periodically using the job id to get the data

8. Conclusion

The idea here was to describe the building blocks of a core business service on Microsoft Azure cloud ecosystem and seamless migration of the entire data from a SQL running on the premise to the Azure cloud. If you are interested to know more details and on the cost benefit analysis before and after the migration, kindly reach out to us and we would be glad to work with you to take it further.

For further details and enquiries, please contact:

IOP Technologies LLP	www.ioptechnologies.com
516A, 4th Main, 4th Phase	Ph: 080-26581588
JP Nagar, Bangalore INDIA 560078	Email: iop.support@ioptechnologies.com

Copyright © 2018 IOP Technologies LLP

All rights reserved. No part of this document may be reproduced, stored or transmitted in any form without the prior written permission of IOP Technologies LLP. IOP Technologies endeavours to ensure that the information in this document is correct and fairly stated, but does not accept liability for any errors or omissions.