Queries over Heterogeneous Data Stores Applications in Cloud Environments

Diksha P. Kapse
Student, CSE Department, DES'sCOET Dhamangaon Rly, India.

Monika R. Nagoshe
Student, CSE Department, DES'sCOET Dhamangaon Rly, India.

Abstract – In period of time generation of enormous quantity data and also rise of cloud computing have introduced new aspects for data management. Many applications got to move with many heterogeneous data stores betting on the sort of data they need to manage: ancient data types, documents, simple key-value data, graph data, etc. Interacting with heterogeneous data models via different APIs, multiple data store applications imposes difficult tasks to their developers. Indeed, programmers got to be conversant in completely different APIs. Additionally, developers got to master and handle the advanced processes of cloud discovery, and deployment of application and execution. Projected system represents a declarative approach using ODBAPI sanctioning to lighten the burden of the complex and non-standard tasks of discovering relevant cloud surroundings and deploying applica tions on them whereas letting developers to easily concentrate on specifying their storage and computing needs.

Index Terms – ODBAPI, data stores, rational data stores, NoSQL.

1. INTRODUCTION

Cloud computing, a relatively recent term, has become nowadays a buzzword in the Web applications world. Despite the importance of this paradigm, there is no consensus on the cloud computing definition. In this context, experts present a set of twenty one definitions of cloud computing [1]. Based on these definations, we can define cloud computing as a large scale distributed computing paradigm based on virtualized computing and storage resources, and modern Web technologies. Over the internet network, cloud computing provides scalable and abstracted resources as services. All these services are on demand and cloud computing is often presented at three levels: Infrastructure as a Service (IaaS) where clients can deploy their own software, Platform-as-a-Service (PaaS) where clients can program their own applications using the services of the PaaS and Software as a Service (SaaS) where clients use existing applications of the cloud.

Cloud computing adds execution environments for some emerging applications like big data management due to its elasticity property. The variety property of big data is mainly focused and more precisely on multi data store based applications in the cloud. To satisfy variety of storage requirements, cloud applications needs to access and interact with several relational and NoSQL data stores which has heterogeneous APIs of the data stores which induces problems while constructing, deploying and migrating multi data store applications. Main four troubles are:

1. Elephantine workload on the developer: These days data stores have heterogeneous and variety of APIs. Developers of multi data store based applications need to be known all these APIs while coding their applications.

2. No declarative way to execute complex queries: Heterogeneity of the data models has no declarative way to define and execute complex queries over multiple data stores. This is because of the absence of global schema of heterogeneous data stores. NoSQL datastores does not have specific schema. It means developers should have to manage with the implementation of such complex queries.

3. Code acclimation: Application developers need to re-adapt the application source code to deal with new data stores when applications are migrating from one cloud environment to another. Developers should have potential to learn and use new APIs.

4. Tedious and nonstandard processes of discovery of cloud and deployment: Once an application is build or migrated, developers need to spread it into cloud provider. Discovering the most appropriate cloud environment which can provide data stores requirements and deploying the application on it are tedious and meticulous provider-specific process.

To overcome these problems Proposed system represents a clear approach for discovering proper cloud environments using ODBAPI[4] and redistributing applications on them in the time letting developers simply focuses on specifying their storage and computing requirements

In context of this paper, we propose a set of models, couple of algorithms and the tools required which aim to make developers work of developing, deploying and migration for multi DBMS based applications in cloud environments.

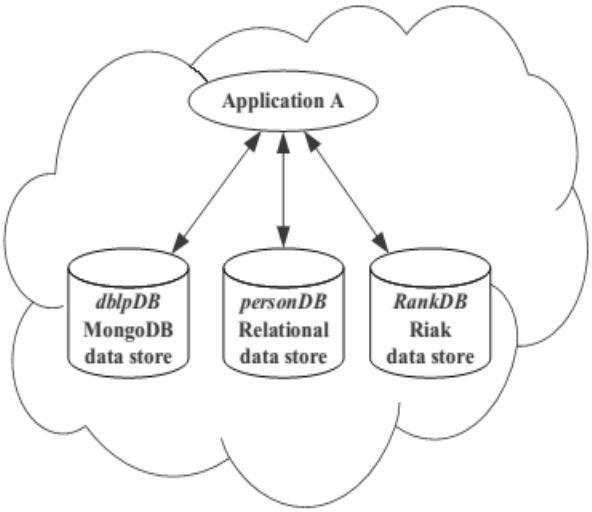

Figure1. Application interacting with three Data Stores in Cloud Computing

In context of this paper, we propose a set of models, couple of algorithms and the tools required which aim to make developers work of developing, deploying and migration for multi DBMS based applications in cloud environments.

Very first, we will define a unique data model which will be used by developers and the programmers to interact and communicate with various data stores. By this model the developers and programmers may write and execute any kind of database queries by using an API called ODBAPI. Then after, we propose virtual data storage system to properly evaluate and execute such queries, more especially complex queries over the different data stores. And finally, here we give a declarative approach for discovering the legitimate cloud environment setting and deploying the applications on those environments while permitting the developers to simply focus on specifying their storage and their computing environments.

In context of this paper, we propose a set of models, couple of algorithms and the tools required which aim to make developers work of developing, deploying and migration for multi DBMS based applications in cloud environments. Very first, we will define a unique data model which will be used by developers and the programmers to interact and communicate with various data stores. By this model the developers and programmers may write and execute any kind of database queries by using an API called ODBAPI. Then after, we propose virtual data storage system to properly evaluate and execute such queries, more especially complex queries over the different data stores. And finally, here we give a declarative approach for discovering the legitimate cloud environment setting and deploying the applications on those environments while permitting the developers to simply focus on specifying their storage and their computing environments.

2. RELATED WORK

In our preceding work, we generally tend to focused on present solutions of the modern. supporting multiple data stores based broadly speaking applications within the cloud setting. greater precisely, (i) we have a tendency to represented completely one-of-a-kind situations regarding the technique applications use information stores, (ii) we enerally tend to outlined the data necessities of packages in cloud setting, and (iii) we've analyzed and categorized current works on cloud statistics control, that specialize in more than one information stores requirements. As a end result, we've got recounted six requirements of the use of the a couple of information stores in a completely cloud setting. By those requirements, we tend to advise our end-to-end way to guide more than one data stores applications in cloud environments. To the handiest of our facts, there aren't any another international solution addressing all the problems we generally tend to specialize in. There are a few proposals just like the Spring data Framework, SOS, and ONDM that enable an application to query the information from absolutely different NoSQL data stores. The Spring information Framework presents some standard abstractions to handle differing varieties of NoSQL DBMSs and relative DBMSs. however, the addition of a new know-how save is not truly clean and additionally the solution is powerfully coupled to the Java programming version. SOS affords CRUD operations at the quantity of man or woman expertise store. Those operations vicinity unit supplied thru GET, positioned, and DELETE methods. They argue that SOS can be extended to integrate relative expertise store; in the intervening time, there is no proof of the efficiency and additionally the extensibility in their device. while, ONDM affords ORM like API supported substantial Java endurance API (JPA) to utility builders to transport with NoSQL understanding stores. But, ONDM does not take underneath attention relative expertise stores. There is additionally a relaxation API referred to as RSenter that permits the records mining and seek of unstructured understanding hold on in NoSQL statistics stores. Though this work is charming, it does not adjust the CRUD operations and complicated queries execution. Moreover, it doesn't take underneath attention the relative information store. In any other preceding work, we generally tend to deliberate a universal asset version shaping the numerous notion employed in every style of information store. These resources area unit managed by using ODBAPI an efficient and a unified rest API sanctioning to execute CRUD operations on absolutely exceptional NoSQL and relative databases. ODBAPI decouples cloud packages from information stores assuaging so their migration. Moreover it relieves developers work by means of getting rid of the overhead of coping with absolutely one of a kind genus API's.

3. PORPOSED MODELLING

A. System Overview

The approach of a system mainly relies on the four

elements. Those are as follows:

i. A Unifying Data Model

ii. REST API/Services

iii. Virtual Data Stores

iv. Dedicated Components for Discovery and Deployment

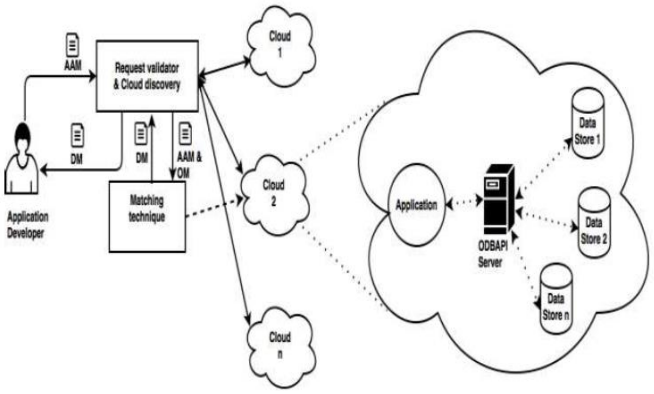

Figure2. Overview

Unifying data model. We define a data model which abstracts from the underlying (explicit/implicit) integrated data store models, and provide a common and unified view so that developers can define their queries over heterogeneous data stores. During the development step, the developers dispose of a global data model expressed according to our unifying model and which integrates local data store models. Our unifying data model decouples query definitions from the data stores specific languages. (contributing to resolving thereafter 1 and 2).

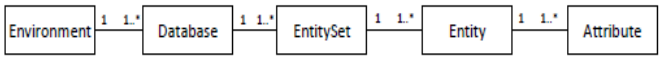

Figure3: Unifyng data model

REST API/services. Based on our unifying data model, we define a resource model upon which we develop a REST API, called ODBAPI, enabling to interact with involved data stores in a unique and uniform way. Each data store will be then wrapped behind a REST service implementing ODBAPI. Our API decouples the interactions with data stores from their specific drivers. By using our unifying data model to express the queries and ODBAPI to interact with the data stores, developers do not have to deal with various languages and APIs and do not have to adapt their code when migrating their applications (resolving thereafter1 and 3).

Virtual data stores. Wrapper REST services enable executing simple queries over the involved data stores. However, they are not meant to execute complex queries (such as join, union, etc.). In our approach, we consider virtual data store (VDS for short) a specific component responsible for executing queries submitted by a multiple data store application. A VDS (1) holds the global data model integrating the different data stores and which is specified according to our unifying data model and a set of correspondence rules, (2) is accessible as a REST service complying to the ODBAPI, and (3) maintains the end-points of the wrapper REST services (in other word the integrated data stores). A multiple data store application submits CRUD and complex queries to the VDS which is responsible of their execution by interacting with appropriate data stores via their REST services. VDSs enable developers to express their join queries over multiple data stores in a declarative way and take in charge the burden of their executions.

Dedicated components for discovery and deployment.

In our approach, we consider two components, the discovery and deployment modules, responsible of finding appropriate cloud environments and deploying multiple data store applications on them respectively. As depicted in Fig. 1, developers express first their requirements about the used data stores as well as the computation environment via an abstract application manifest. Based on that manifest, the discovery component finds and selects the appropriate cloud environment and produces an offer manifest. This manifest will be in turn used by the deployment component to deploy the application on that selected environment. The discovery and deployment modules relieves the application developers from the burden of dealing with different APIs and discovery/deployment procedures (resolving thereafter 4).

B. OPENPAAS DATABASE API: ODBAPI

In this section, we introduce ODBAPI which is a REST API enabling the execution of CRUD operations on different types of data stores supporting our unifying data model. This API is designed to provide an abstraction layer and seamless interaction with data stores deployed in a cloud environment. Developers can execute queries in a uniform way regardless of the type of the data store (relational or NoSQL). An overview of the API is given in Fig. 4. The figure is divided in four parts that we introduce in the following starting from the right to the left side. First, we have the deployed data stores (e.g. relational DBMS, Couch DB, etc.) that a developer may interact with. Second, we find the proprietary API and driver of each data store supported by ODBAPI. For instance, we use in our API implementation the JDBC API and MySQL driver to interact with a relational DBMS. The third part of Fig. 5 refers to the ODBAPI implementation. In fact, this part is shared between all the integrated data stores. In addition, it contains specific implementation of each data store. To integrate a new data store and define interactions with it, one has simply to add the

specific implementation of that data store. Finally, Fig. 4 shows the different operations that ODBAPI offers.

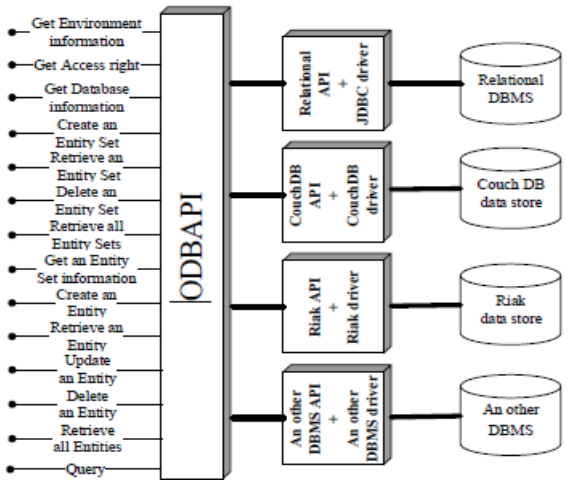

Figure 4. Overview of ODBAPI

4. CONCLUSION

The above discussion conclude a generic approach to facilitate the developer task and enable the development of applications using multiple data stores while remaining agnostic to these latter. We introduced three solutions:

- ODBAPI for CRUD operations:
- Virtual data stores for complex queries execution:
- Manifest for data stores discovery and automatic application deployment:

Currently, We are working on applying ODBAPI and the virtual data store query optimization and execution approach to other qualitatively and quantitatively various scenarios in the OpenPaaS project. This allows us to identify possible discrepancies and make my work more reliable for public use. In addition, we aim to study an implementation for Hive allowing access to Hadoop data stores. Our second perspective consists in providing another matching algorithm supporting approximate matching. Hence we enable more flexibility in data stores discovery and applications deployment. Our third perspective is an extension to virtual data stores, allowing to support a larger class of complex queries across NoSQL and relational data stores (union, intersection, aggregates, group by like operations) and introducing more elaborate query processing optimization techniques, including asynchronous evaluation.

REFERENCES

[1] Geelan, J., et al.: Twenty-one experts define cloud computing
[2] Team, M.: Microsoft azure (2013)
[3] Weissman, C.D., Bobrowski, S.: The design of the force.com multitenant internet application development platform. In: Proceedings of the 2009 ACM SIGMOD International Conference on Management of data. (2009) 889-896
[4] Google: Google appengine documentation (2009)
[5] Fay, C., et al.: Bigtable: A distributed storage system for structured data. ACMTrans. Comput. Syst. 26(2) (2008)
[6] Cooper, B.F., et al.: Pnuts: Yahoo!'s hosted data serving platform. PVLDB 1(2)(2008) 1277-1288
[7] DeCandia, G., et al.: Dynamo: amazon's highly available key-value store. In:Proceedings of the 21st ACM Symposium on Operating Systems Principles, SOSP2007, Stevenson, Washington, USA, October 14-17. (2007) 205-220
[8] D. Agrawal, El Abbadi, A., Das, S., Elmore, A.J.: Database scalability, elasticity, and autonomy in the cloud - (extended abstract). In: Database Systems for Advanced Applications - 16th International Conference, DASFAA 2011, Hong Kong, China, April 22-25, 2011, Proceedings, Part I. (2011) 2-15
[9] Mark, C., et al.: Cloud application management for platforms (August 2012)
[10] Truong, H.L., et al.: Exchanging data agreements in the daas model. In: 2011 IEEE Asia-Pacific Services Computing Conference, APSCC 2011, Jeju, Korea (South), December 12-15. (2011) 153-160
[11] Vu, Q.H., et al.: Demods: A description model for data-as-a-service. In: IEEE 26th International Conference on Advanced Information Networking and Applications, AINA, 2012 , Fukuoka, Japan, March 26-29. (2012) 605-612
[12] Truong, H.L., Comerio, M., Paoli, F.D., Gangadharan, G.R., Dustdar, S.: Data contracts for cloud-based data marketplaces. IJCSE 7(4) (2012) 280-295
[13] Ruiz-Alvarez, A., Humphrey, M.: An automated approach to cloud storage service selection. In: Proceedings of the 2nd international workshop on Scientific cloud computing. ScienceCloud '11, New York, NY, USA, ACM (2011) 39-48
[14] Poole, J.D.: Model-driven architecture: Vision, standards and emerging technologies. In: In In ECOOP 2001, Workshop on Metamodeling and Adaptive Object Models. (2001)
[15] Peidro, J.E., Mu~noz-Esco__, F.D.: Towards the next generation of model driven cloud platforms. In: CLOSER 2011 - Proceedings of the 1st International Conferenceon Cloud Computing and Services Science, Noordwijkerhout, Netherlands,7-9 May. (2011) 494-500
[16] Lim, H., Han, Y., Babu, S.: How to _t when no one size fits. In: CIDR 2013, sixthBiennial Conference on Innovative Data Systems Research, Asilomar, CA, USA,January 6-9, Online Proceedings. (2013)
[17] Kossmann, D., et al.: Cloudy: A modular cloud storage system. PVLDB 3(2) (2010) 1533-1536
[18] R. Sellami, S. Bhiri, and B. Defude, "ODBAPI: a unified REST API for relational and NoSQL data stores," in The IEEE 3rd international Congress on Big Data (BigData'14), Anchorage, Alaska, USA, June 27 - July 2, 2014, 2014.
[19] R. Sellami and B. Defude, "Using multiple data stores in the cloud: Challenges and solutions," in Data Management in Cloud, Grid and P2P Systems - 6th International Conference, Globe 2013, Prague, Czech Republic, August 28-29, 2013. Proceedings, 2013, pp. 87–98.
[20] M. Pollack, O. Gierke, T. Risberg, J. Brisbin, and M. Hunger, Eds., Spring Data. O'Reilly Media, October 2012.
[21] P. Atzeni, F. Bugiotti, and L. Rossi, "Uniform access to nonrelational database systems: The sos platform," in Advanced Information Systems Engineering - 24th International Conference, CAiSE 2012, Gdansk, Poland, June 25-29, 2012. Proceedings, 2012, pp. 160– 174.
[22] L. Cabibbo, "Ondm: an object-nosqldatastore mapper," Faculty of Engineering, Roma Tre University. Retrieved June 15th, 2013.
[23] R. K. Lomotey and R. Deters, "Rsenter: Tool for topics and terms extraction from unstructured data debris," in IEEE International Congress on Big Data, BigData Congress 2013, June 27 2013-July 2, 2013, 2013, pp. 395–402.
[24] "Data mining from document-append nosql," International Journal of Services Computing (IJSC), vol. 2, no. 2, pp. 17–29, 2014.