Vol.3 Issue.7, Pg.: 92-94
July 2015

INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

BIG DATA AND POLYGLOT PERSISTENCE

Anu Taneja

anutaneja16@gmail.com

Abstract: Different databases are designed to solve different problems. Using a single database engine for all of the requirements usually leads to non- performing solutions. So in this paper, the term polyglot persistence has been introduced to describe using different data storage technologies to handle varying data storage needs.

***Keywords*:** OLAP-On-line Transaction Processing, OLTP- On-line Analytical Processing, RDBMS-Relational Database Management System.

I. Introduction

Different databases are designed to solve different problems. Using a single database engine for all of the requirements usually leads to non- performing solutions; storing transactional data, caching session information, traversing graph of customers and the products their friends bought are essentially different problems. Even in the RDBMS space, the requirements of an OLAP and OLTP system are very different—nonetheless, they are often forced into the same schema.
Let’s think of data relationships. RDBMS solutions are good at enforcing that relationships exist. If we want to discover relationships, or have to find data from different tables that belong to the same object, then the use of RDBMS starts being difficult.
Database engines are designed to perform certain operations on certain data structures and data amounts very well—such as operating on sets of data or a store and retrieving keys and their values really fast, or storing rich documents or complex graphs of information.

II. Disparate Data Storage Needs

Many enterprises tend to use the same database engine to store business transactions, session management data, and for other storage needs such as reporting, data warehousing, or logging information.

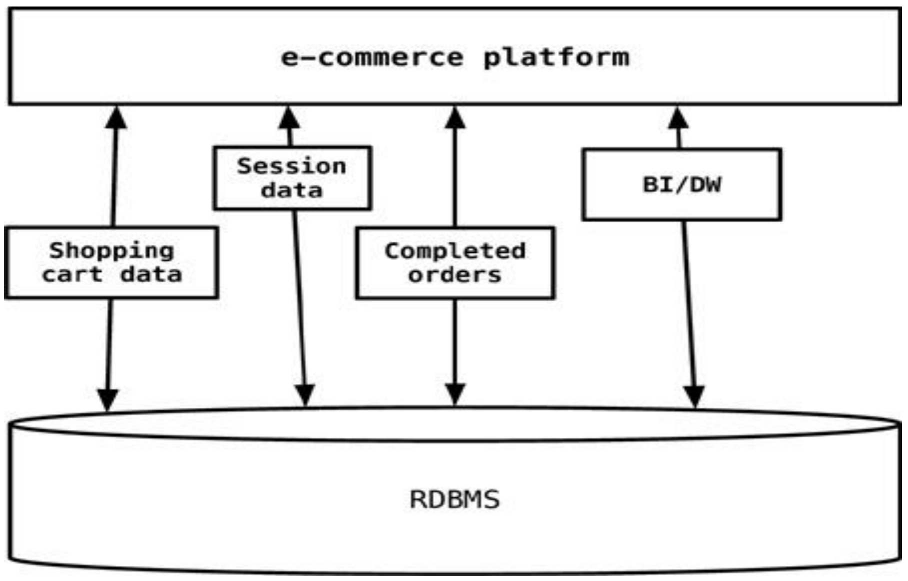

Figure 1: Use of RDBMS for every aspect of storage for the application

The session, shopping cart, or order data do not need the same properties of availability, consistency, or backup requirements. Does session management storage need the same rigorous backup/recovery strategy as the e-commerce orders data? Does the session management storage need more availability of an instance of database engine to write/read session data?

III. Polyglot Persistence

The term polyglot is borrowed and redefined for big data as a set of applications that use several core database technologies, and this is the most likely outcome of implementation planning. The official definition of *polyglot* is "someone who speaks or writes several languages." It is going to be difficult to choose one persistence style no matter how narrow your approach to big data might be.

A polyglot persistence [1] database is used when it is necessary to solve a complex problem by breaking that problem into segments and applying different database models. It is then necessary to aggregate the results into a hybrid data storage and analysis solution. A number of factors affect this decision:

We are already using polyglot persistence in your existing workplace. If your enterprise or organization is large, you are probably using multiple RDBMS, data warehouses, data marts, flat files, content management servers, and so on. This hybrid environment is common, and you need to understand it so that you can make the right decisions about integration, analytics, timeliness of data, data visibility, and so on. You need to understand all of that because you need to figure out how it is going to fit into your big data implementation.

The most ideal of environments, where you have only one persistence technology, is probably not suited to big data problem solving. At the very least, you will need to introduce another style of database and other supporting technologies for your new implementation.

Depending on the variety and velocity of your big data [2] gathering, you may need to consider different databases to support one implementation. You should also consider your requirements for transactional integrity. Do you need to support ACID compliance or will BASE compliance be sufficient?

This is a big data [3] challenge at its best. Multiple sources of data with very different structures need to be collected and analyzed. This type of problem cannot be solved easily or cost-effectively with one type of database technology. Even though some of the basic information is transactional and probably in an RDBMS, the other information is non-relational and will require at least two types of persistence engines (spatial and graph) so for this we have polyglot persistence.

IV. Advantages of Polyglot Persistence

a) **Faster Response Time:** You leverage all the features of databases in your app which makes the response time of your app very fast.

b) **Helps your app to scale well:** Your app scales exceptionally well with the data. All the Non SQL databases scale well when you model databases properly for the data that you want to store.

c) **A rich experience:** You have a very rich experience when you harness the power of multiple databases at the same time.

V. Disadvantages of Polyglot Persistence

a) **Requires you to hire people to integrate different databases:** If you are an enterprise, you will have the resources to hire experts for each database type. But if you are a small company or startup then you may not have the resources to hire people to implement a good polyglot persistence model.

b) **Implementers need to learn different databases:** If you are an individual developer or start-up building apps on databases, then you have no choice but to learn multiple types of databases and implement a good polyglot persistence model for your app.

c) **Requires resources to manage databases:** If you are running multiple databases for your app, then you need to take care of backups, replicas for each type of databases which is time-consuming.

d) **Testing can be tough:** If you share your data into multiple databases then testing of data layer can be complicated and debugging can usually be time consuming.

VI. Conclusion

The design of a database determines its optimal use. A single database engine is inefficient and insufficient for all data searches. This is where polyglot persistence comes in to share or divide your data into multiple databases and leverage their power together.

VII. REFERENCES

1. http://martinfowler.com/bliki/PolyglotPersistence
2. https://www.mapr.com/products/polyglot-persistence
3. http://www.dummies.com/how-to/content/big-data-and-polyglot-persistence