WHITE PAPER

DATABASE RESILIENCE AND SCALABILITY – A BY DESIGN PERSPECTIVE FOR FINANCIAL SERVICES

Abstract

The pandemic has taught us the importance of being resilient. During pandemic we all realized the importance of resilient Financial Systems as they are the backbone of the economy and can't afford downtimes. Recently US Federal Banking Regulators[1] and the Financial Conduct Authority[2] in the UK has come up with the operational resiliency requirements which mandates Financial System's compliance. Over the years financial institutes are using databases as a mechanism to persist the data. But due to new regulatory requirements it is necessary to think beyond persistence and to make the databases resilient to improve resiliency of the financial systems.

Infosys®
Navigate your next

Introduction

Data is the heart of financial systems and helps businesses to take informed decisions. Data is also helpful to feed in financial echo systems like peer banks, merchants, government departments, regulatory systems. Data is consumed as reports, dashboards, by downstream systems such as anti-money laundering, risk calculation engines, threat monitoring systems, financial budgeting systems, to list a few.

Digitization of Financial Systems is driving the need to get real time insights from the data. The insight could be related to customer behavior, cross selling opportunities, identifying malicious transactions and more. Digitization is helping financial institutes to go beyond their organization boundaries to tap data from various channels like social media, personal messaging platforms etc., leaving us with an enormous amount of data for consumption. It is important to act on the data as and when it is available. Failure to access data or respond faster can put businesses at a huge risk to be continuing business as usual, leading to financial losses, customer attrition and even regulatory penalties.

Data Management current scenario and challenges

Currently, data is mostly stored into relational databases which makes databases an integral part of architecture design. Relational databases are used for storing structured data. When it comes to unstructured or semi structured data, NoSQL databases are preferred choices. Databases are primarily used to persist and query data.

Databases are major contributors to resiliency and scalability issues. Below are some of the issues encountered with respect to the databases,

- User Screens taking unusually higher time to retrieve data
- Data retrieval time is not consistent and varies drastically
- During peak workload user transactions ends abruptly throwing errors
- Applications faces issues due to technical glitches like database failover
- Higher resource utilizations like CPU and Memory impacting application availability during peak workloads

Most of these scenarios has impact on the user experience. Sometimes the users lose the data they entered and need to reenter the data again resulting in anxious customers.

The database designs have been the least focused so far. As a result of this, most of the applications exhibit resiliency and scalability issues post few years into production. Some of the contributing factors which makes databases non-performant are listed below

- Adoption of Agile development has led to faster releases, but it has shortened the development time
- The development teams are mostly focused on implementing the functionality

- More attention is given to application design
- Non-production environments lack sufficient data volumes necessary for performance testing. Hence most of the applications pass the performance tests in lower environments but fail in the production
- Non-functional requirements like Throughput, High Availability, Recovery Point Objective (RPO), Recovery Time Objective (RTO) are often not thought through or neglected during the design phase
- New Database features, database design best practices are not considered during design phase
- Inadequate database monitoring from data growth and SQL monitoring perspective lead to frequent database issues
- Insufficient and inadequate database housekeeping jobs impacting database health
- Database sizes grows over a period of time impacting the database scalability and increase in storage footprint
- Database licensing and infrastructure costs plays an important role for taking decisions related to scalability and resiliency. Most of the databases are licensed per CPU Core. Which means more the no. of CPUs, the cost of licensing will be more.
- Overreliance on autoscaling feature of cloud

Cloud first strategy:

Financial Services clients have already started adopting cloud technologies and migrating their applications and databases to cloud. In order to curb the database licensing cost, clients have started evaluating the cloud native database solutions like AWS Aurora, DynamoDB, RDS or Open Source database solutions like MySQL, PostgreSQL etc.

Gartner expects that 75 percent of all databases will be deployed or migrated to cloud platforms by next year[3]

Adopting new technologies often fail to deliver the expected outcome. This white paper outlines the best practices to deal with database resilience and scalability issues. The paper also attempts to answer whether should the aforementioned database challenges be addressed first or migrate AS IS to cloud native database solutions.

Building Scalable and Resilient database – *by Design* Perspective

The database specific issues our financial clients are facing are common in the industry. These issues can be tackled by implementing best practices which are technology agnostic and provide predictable outcomes if implemented properly. A ***by Design*** approach rather than a reactive way of solving resilience and scalability issues helps arrest most of the issues during design phase itself. These best practices would also help clients not only in improving resilience and scalability of their on-premise databases but also facilitate the cloud adoption journey.

Best practices to make the database Resilient and Scalable

I - Archival and Purging – A secret weapon:

Archival and Purging helps in keeping the databases lean. This reduces the storage footprint of the database and the backup. Lean databases help in improving overall database performance. Due to reduction in amount of data, the strategy also helps in cloud or database migration journey. We have often observed that transactional databases are designed in a way that very few tables occupy more than 80% of total database size. Data retrieval from these tables is always a challenge and results in heavy resource utilization. Archival and purging policy is often a neglected but more powerful weapon to keep check on growing tables. It is a very simple concept and the most effective one while dealing with oversized tables. Effective implementation of Archival and Purging process results in lean database tables.

Below diagram depicts archival and purging process,

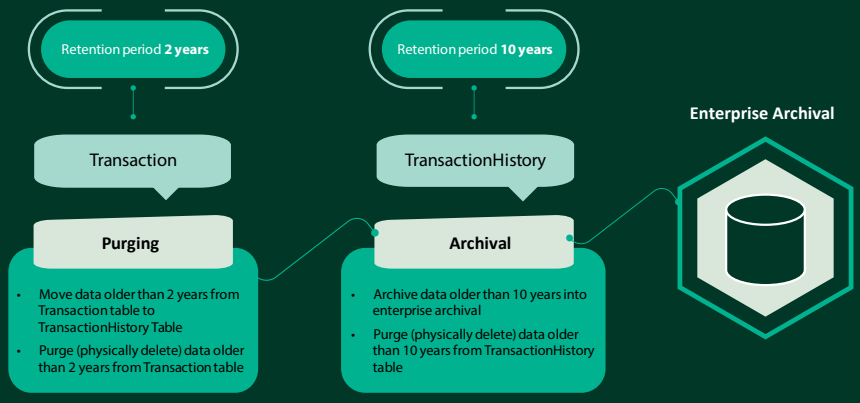

Figure 1: Archival and Purging Process

II - Adopt Shift Left strategy – to gain more insights into resilience and scalability issues right from design phase

For many financial services clients, scalability issues are often visible only in production and never detected in lower environments. One of the reasons is, lower environment databases lacks data volume of production databases. Due to less data volumes most of the applications satisfy Service Level Agreement (SLA) requirements during performance test. Hence it is recommended

- To refresh the lower environment databases with production data for each release cycle.
- To avoid data security issues, it is necessary to mask sensitive data in lower environments.

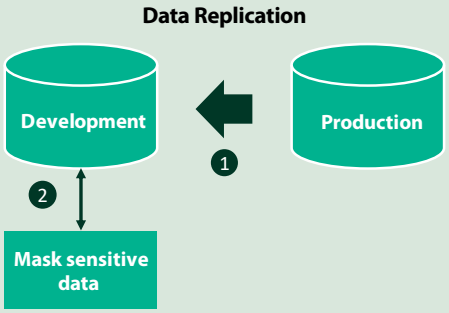

Figure 2: Data Refresh and Masking activity

- Along with this, performance testing should be integrated as part of build process in DevOps pipeline. The build should get deployed in upper environments only when the SLAs are met during performance testing. Below diagram showcases indicative build process with control gates to restrict deployment to upper environment in case SLAs are not met.

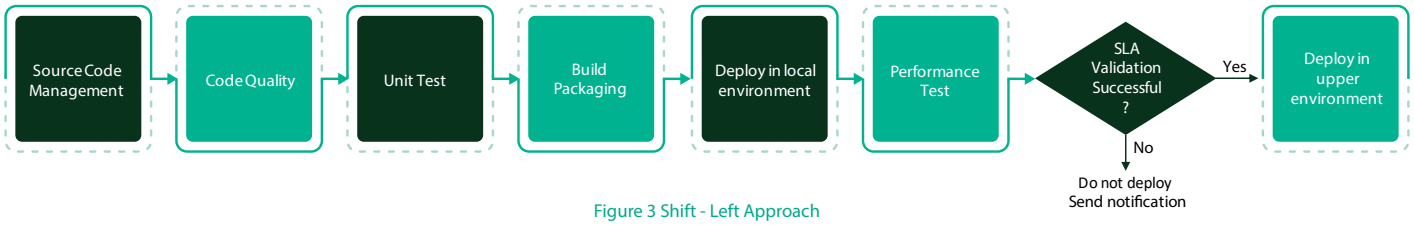

Figure 3 Shift - Left Approach

This way the development/ testing teams could arrest performance issues during development phase itself. This best practice is part of a Shift Left strategy, which enables development teams to identify resiliency and scalability related issues at lower environments.

III : Monitor SQL queries on frequent basis

Generally financial services clients prefer monitoring end-to-end business flows for any financial impacts. Whereas at the infrastructure level, most of the time the database CPU and Memory utilizations are monitored. SQL monitoring is not implemented in case of most of the databases. As part of SQL monitoring, DBAs should monitor production databases on daily basis to identify,

- SQLs changing execution plans, resulting in phenomena called SQL Regression
- SQLs suddenly consuming more CPUs
- SQLs having sudden increase in elapsed time
- SQLs with high no. of execution count in a day or in an hour

This monitoring could help in identifying change in behavior of SQL performance. Along with monitoring, the SMART alerting mechanism would help in taking necessary actions before the problem aggravate. Below diagram depicts scenario for monitoring and alerting of SQL performance.

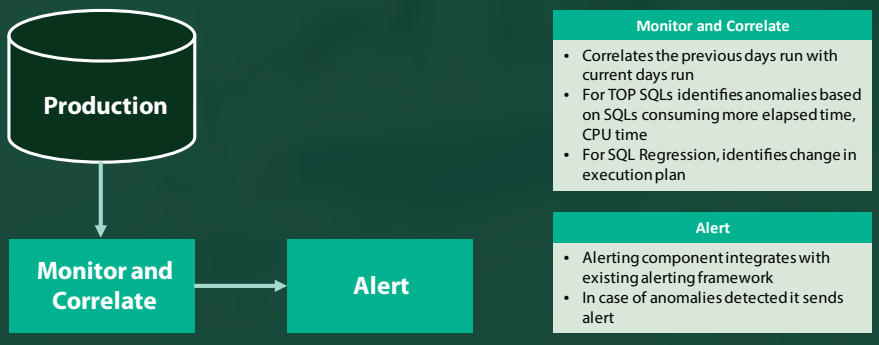

Figure 4: SQL monitoring and alerting

IV : Avoid overreliance on a particular database

It is highly likely that one database could become an integral part of the entire echo system for any data requirements. A typical example could be a legacy mainframe system. These systems hold crucial data needed for day-to-day business operations, for example, customer information, accounts information etc. Hence most of the applications within the financial system source data from these systems. But overreliance on one database could result in catastrophe. Recently one of the US banks faced major outages for their mainframe systems which resulted in collapse of their dependent software systems and caused outages for IT applications.

To avoid such dependency, it is recommended to create a redundant database. This database should be synched up with mainframe database on Realtime or Near-Realtime basis. The IT applications which were dependent on mainframe system could be pointed to redundant database to read data. Only when the redundant database is not available the application should read data from mainframe database.

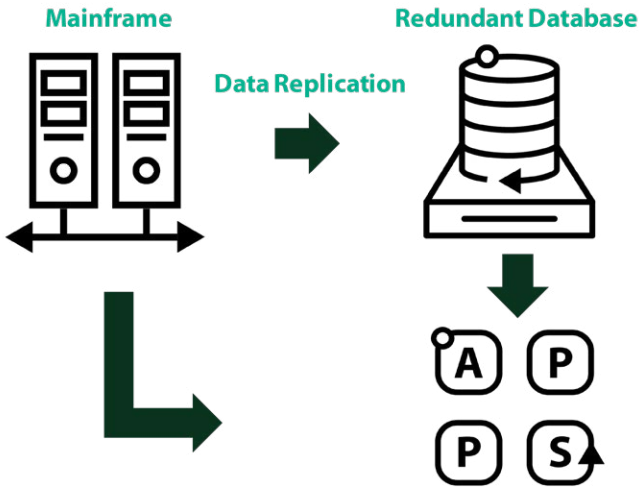

Figure 5: Data replication solution to avoid database dependency

V : Blue Green Deployments

Database clustering helps in taking care of single point of failure at primary site. But what if the entire primary site is down due to a natural disaster like earth quake, fire, flooding? How can the business continuity is maintained in such situations?

The answer is, by provisioning disaster recovery site or standby site. This will add up to the IT cost, but make sure that, even though the primary site is not available, the business could continue from Standby site. There are regulatory requirements that define the distance between the Standby site and the Primary site. This ensures that the standby site is intact even though the primary is impacted, thus maintaining business continuity.

Another important benefit of this design is that it helps minimize the application downtime due to maintenance activities like OS upgrades, database patch upgrades, network switch upgrades etc. These upgrades are carried out at the Standby site first. Hence any impact due to upgrades is isolated only to the Standby site. Only when the upgrades are successful at Standby, they are performed at Primary site. This setup helps in Blue-Green deployments to ensure application availability.

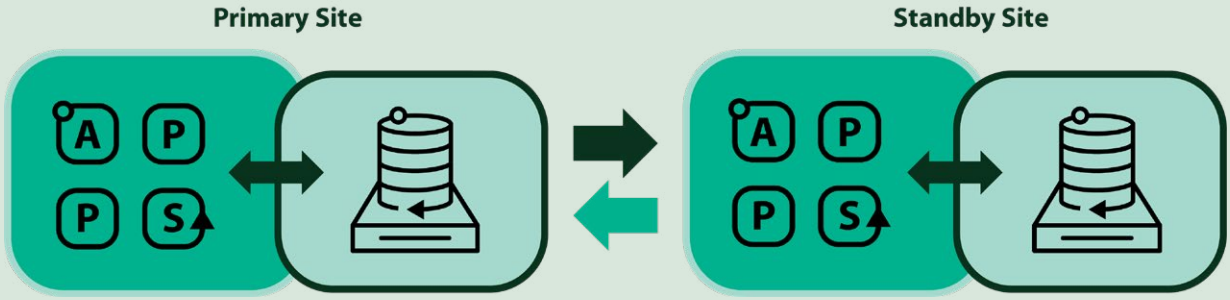

Figure 6: Blue Green deployment setup

VI : Application Testing post OS/ Database patch or version upgrades in production

Often, database version upgrades, patching or even OS upgrades are classified as non-disruptive changes. Hence no proper application testing is carried out in production post such upgrades. This could lead to application downtime in case of impact due to upgrades. Hence it is recommended to perform testing of critical workflows in production to make sure that they are working fine.

Such testing becomes easy if a Standby site is available. The upgrades could be first done on Standby site and tested thoroughly by executing some critical end-to-end workflows. Only when the testing is successful at standby, the upgrades could be carried at Primary site and tested again at primary site for critical workflows.

In the past many banks faced outages due to software upgrades. Hence it is recommended to follow this best practice for incident free upgrades.

VII : Implement Chaos Engineering to simulate failures and analyze their impact on application

Often during development and coding phase, the developers assume availability of certain things and overlook unlikely events while coding. But when such events happen, the application availability gets impacted and sometimes the situation results in IT outages.

For example, developers always assume that databases are available 100% of the time. It is claimed that database failover is transparent to application and don't have an impact on application processing. But that's not true.

We simulated a database failover for one of the application in lower environment and assess the impact on application and batches. Post failover, most of the applications started throwing database errors and crashed. We have discovered that the applications were never designed to take care of database failover situation and hence failed the test. It is necessary to build resilience in the application to deal with database failover.

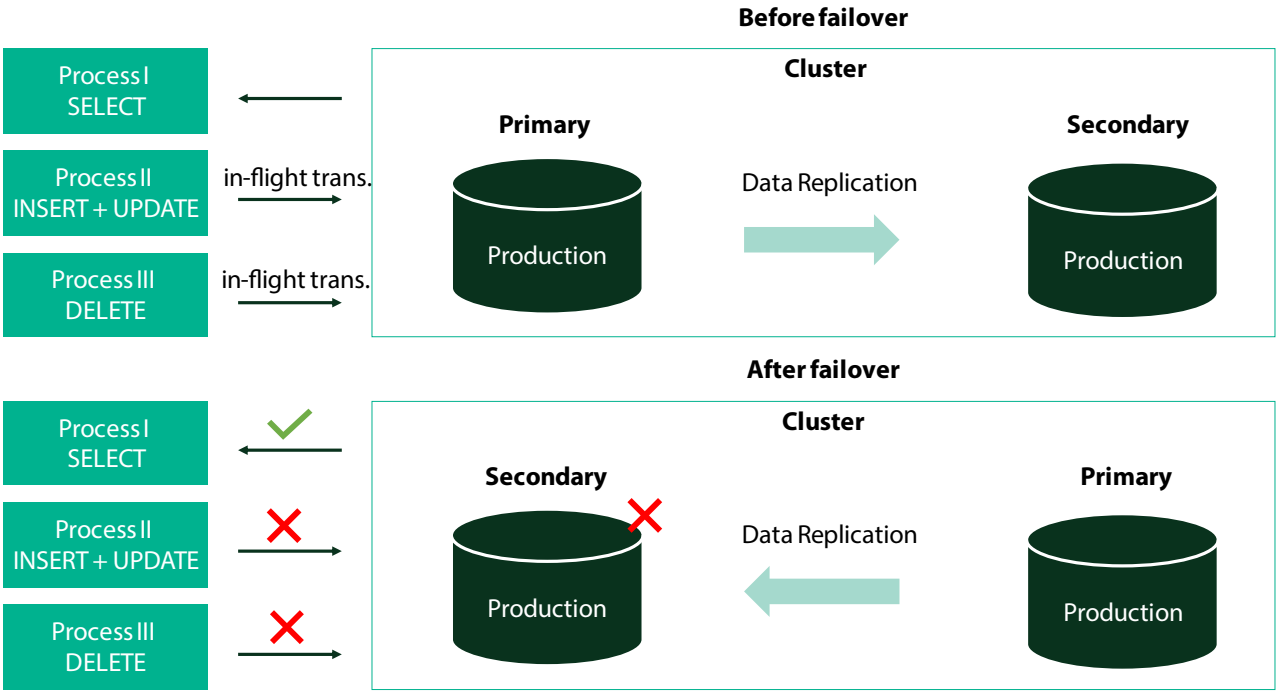

Figure 7 Database Failover - Application Impact

VIII : Setup Site Reliability Engineering teams to take care of application resilience and scalability for long term

Site Reliability Engineering[4] is pioneered at Google in 2003. It's a discipline which helps improving and maintaining the systems resiliency and scalability over the period. It treats operational problems as software problems. As per Google's recommendations[4], the SRE engineers should spend 50% of their time on development and rest 50% of the time in operations, supporting the applications. The team which develops the software is best suited to maintain it too, that's the simple thought behind this. The SRE Engineers helps in building capabilities around various areas like,

- Application Monitoring capabilities
- Automated End to End correlation of data gathered through monitoring
- SMART alerting based on the AI-ML algorithms to avoid any false positives
- Automating manual operational works to reduce the TOILs in production
- Blameless retrospection of production issues to identify Root Cause
- Chaos Engineering to simulate failures and assess their impact on the applications
- Build self-healing capabilities within application to sustain failures

There is a wide adoption of SRE discipline across financial services. In order to comply with regulatory mandates financial institutions have started building SRE teams to improve system resiliency and scalability.

Conclusion:

The implementation of the various best practices outlined in the paper help in improving the database resiliency. The awareness towards the criticality of improving software system resiliency is increasing across the globe. Financial institutions have realized the importance of being resilient and the benefits of resilient systems . Many financial institutions have embarked on a legacy modernization journey as part of monolith to microservices migration. Hence this is the right time to adopt Resilience by Design methodology to take care of the resiliency requirements right from the design phase and deliver reliable and robust systems.

About the author

Shailendra Shantaram Hirlekar, *Senior Technology Architect, Financial Services, Infosys Limited*

Passionate about learning new technologies. Performance Engineering and Resilience practitioner. Helping FS client to achieve Resilience and Scalability of software systems.

References

1 US Federal Banking Regulators, October 30, 2020, federalreserve.gov
2 Financial Conduct Authority in the UK, March 29, 2021, fca.org.uk
3 Gartner Says the Future of the Database Market Is the Cloud, July 1, 2019, gartner.com
4 Google's Approach to Service Management: Site Reliability Engineering, Benjamin Treynor Sloss, Betsy Beyer, sre.google

For more information, contact askus@infosys.com

Infosys®
Navigate your next