Postgres Architectures for Enhanced Availability and Manageability

May 18, 2016

Postgres Architectures for Enhanced Availability and Manageability
by EnterpriseDB® Corporation

EnterpriseDB Corporation, 34 Crosby Drive, Suite 201, Bedford, MA 01730, USA
T +1 781 357 3390   F +1 978 589 5701   E info@enterprisedb.com   www.enterprisedb.com

Table of Contents

1 Introduction

The purpose of this paper is to provide a systematic overview of the principal Postgres clustering types that EnterpriseDB® (EDB™) recommends to help a customer achieve enhanced availability and manageability. We will consider:

- Shared-disk clusters
- Streaming Replication clusters
- Logical Replication clusters
- Hybrid clusters (clusters that combine logical replication and streaming replication)

This paper evaluates each architecture option against key capabilities. The different cluster architectures are evaluated in terms of:

- high availability
- disaster recovery
- near-zero downtime maintenance
- cloud suitability
- transaction throughput

For convenience, a table that summarizes the availability of each capability to each architecture option is included in Section 3.

This paper focuses on solutions enabled by the EDB Postgres™ portfolio. We include a Slony-based solution, though newer Postgres releases are better supported with log-based solutions, such as the EDB Postgres Replication Server.

We do not include solutions building on Londiste, Bucardo or Postgres Bi-directional replication (BDR), which is still pending release.

1.1 Acronyms and Abbreviations Used in this Paper

Acronym	Description
Async	Asynchronous
BDR	Bi-directional Replication
CH	Commodity Hardware
DB	Database
DAS	Direct-attached Storage
DC	Domain Controller
DR	Disaster Recovery, also Off-site Disaster Recovery
EDB	EnterpriseDB Corporation
EFM	EDB Postgres Failover Manager
pre-GA	precedes General Availability
HA	High Availability
HW	Hardware
JClouds	Apache jclouds®
JDBC	Java Database Connectivity
MMR	Multi-master Replication
NZD	Near-Zero Downtime
NZDM	Near-Zero Downtime Maintenance
OS	Operating System
RO	Read Only
RS	Read Scalability
RW	Read/Write
SAN	Storage Area Network
SLA	Service-Level Agreement
SMR	Single-master Replication
SR	Streaming Replication
Sync	Synchronous
TL	Throughput and Latency
TPS	Transaction Processing System
TRC	Trigger-based Replication Clusters
UD	Update
UG	Upgrade
WAL	Write-ahead Log
WAN	Wide Area Network
WS	Write Scalability

2 Overview of Postgres Architectures for Enhanced Availability and Manageability

The following sections discuss system architectures that implement high-availability clusters that ensure data integrity, system performance, and disaster recovery readiness.

2.1 Shared Disk Clusters

A shared disk cluster uses underlying operating system or infrastructure capabilities to facilitate failover or switchover. In a shared disk cluster, different cluster members use one set of data files, and the operating system (or related infrastructure) ensures that only one cluster member is active at a given point in time. Key representatives are Red Hat Cluster Suite / High Availability add-on or Veritas Cluster.

Shared disk clusters should not be confused with streaming clusters that use a shared Storage Area Network (SAN).

2.1.1 Shared Disk Cluster

In a shared-disk cluster, the cluster is locally implemented; two database instances share one set of data files on a SAN.

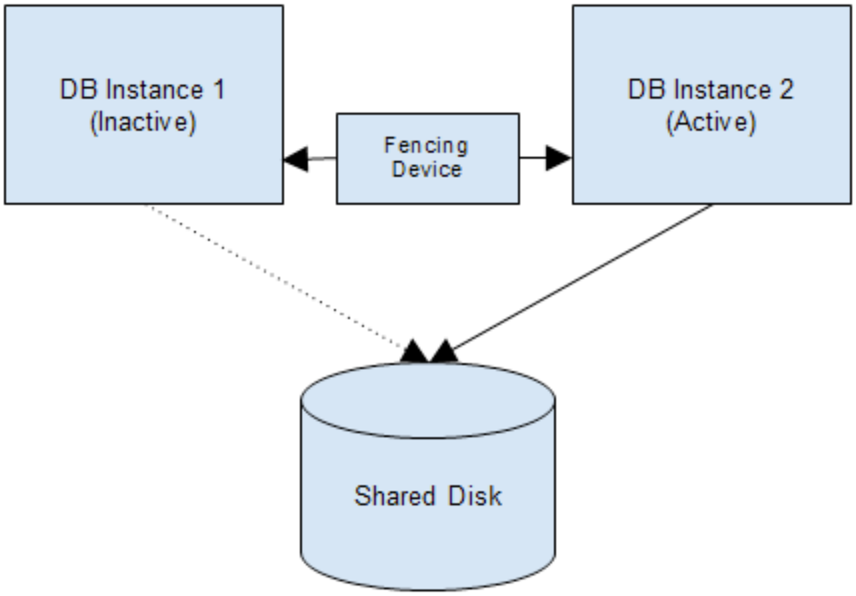

Figure 2.1 – Shared Disk Cluster

Only one instance is active at a given point in time. The use of fencing (hardware or software) avoids the potential for data corruption.

This provides a very robust high availability solution with short error detection cycles and fast failover/failback, although the SAN and the hardware fencing drive up cost. The solution does not provide disaster recovery capabilities – typically both database servers and the SAN are in the same data center.

Shared disk clusters can be used to support minor version updates with minimal downtime; they cannot be used to support near-zero downtime upgrades.

2.1.2 Shared Disk Cluster with Off-site DR Instance

In a shared-disk cluster with off-site disaster recovery, two local database instances share one set of data files on a SAN.

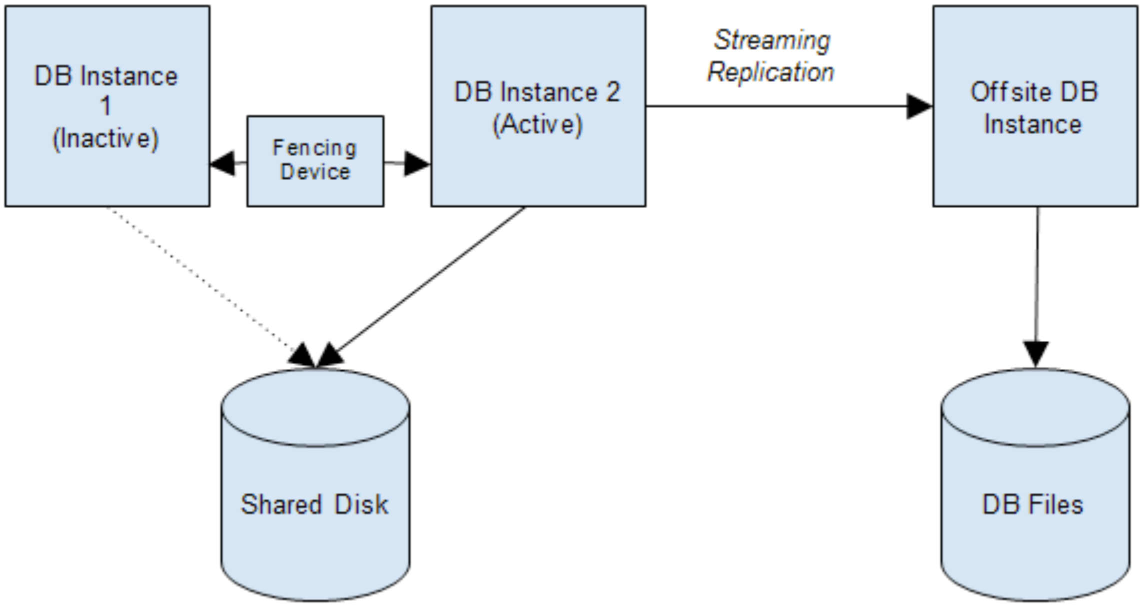

Figure 2.2 – Shared Disk Cluster with Off-site DR Instance

Only one instance is active at a given point in time. The use of fencing (hardware or software) avoids the potential for data corruption.

The currently active member of the cluster replicates to an off-site instance.

2.2 Streaming Replication Clusters

Postgres streaming replication clusters are extremely robust. Streaming replication is a core capability in Postgres that combines high throughput with low latency. A streaming replication scenario contains one master database instance and one or more replicas. The master node streams write-ahead log (WAL) records to the standby node as each WAL record is generated, ensuring that the replica is kept up-to-date with changes to the master.

The master node can accept read/write transactions, while a replica node can accept read-only transactions. A replica can also act as a standby system, providing high-availability and/or disaster recovery. Replicas can reside onsite or be regionally distributed.

Streaming replication clusters can be configured in two major variants: synchronous and asynchronous. Streaming replication clusters can take advantage of SAN technology to create extremely robust high-availability and disaster recovery solutions.

Streaming replication requires binary equivalence between the master and the replicas, which prevents its use across mixed hardware or operating system platforms, across major Postgres versions, or to facilitate upgrades. It can be used to support minor version updates with minimal downtime.

2.2.1 Synchronous Streaming Replication

In a synchronous streaming replication scenario, one master database (configured to accept read/write transactions) and one or more replicas (configured to accept read transactions only) use synchronous streaming replication to update the replicas as part of the transactions that are being committed on the master.

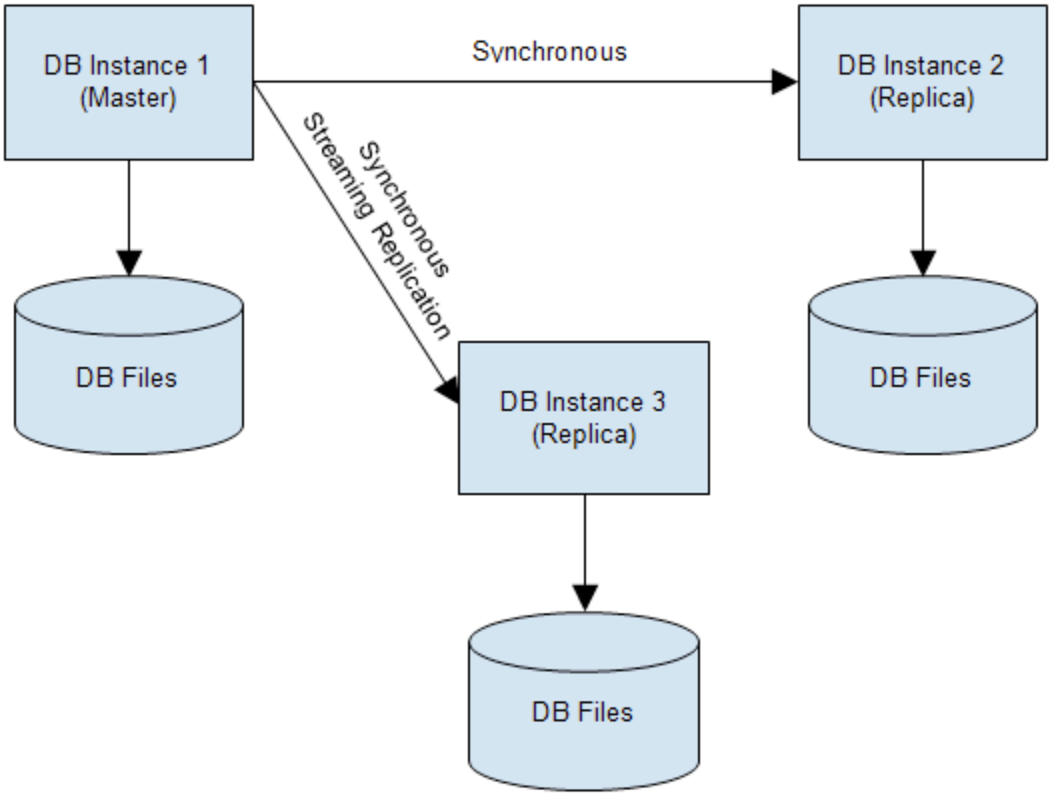

Figure 2.3 – Synchronous Streaming Replication

The master signals the application that a transaction has completed after the replica has committed the transaction to disk (or has indicated that the data for the transaction has been committed to disk on all replicas). In Postgres 9.6, you can set a flag to specify your preference of replication behavior (if the transaction is committed to disk or if the transaction has truly been replayed).

Synchronous replication introduces delays while the master waits for one of the replicas to finish committing the transaction. Depending on transaction volume, network latency, and workload on the replica, this delay can become significant.

While this architecture introduces delays in transaction processing, it can guarantee zero data loss in the case of a catastrophic failure on the master (assuming that the replicas are not impacted by the same calamity).

2.2.2 Asynchronous Streaming Replication on DAS

When using streaming replication on Direct-attached Storage (DAS), one master database, and one or more replicas use asynchronous streaming replication to update the replica after transactions have been committed on the master.

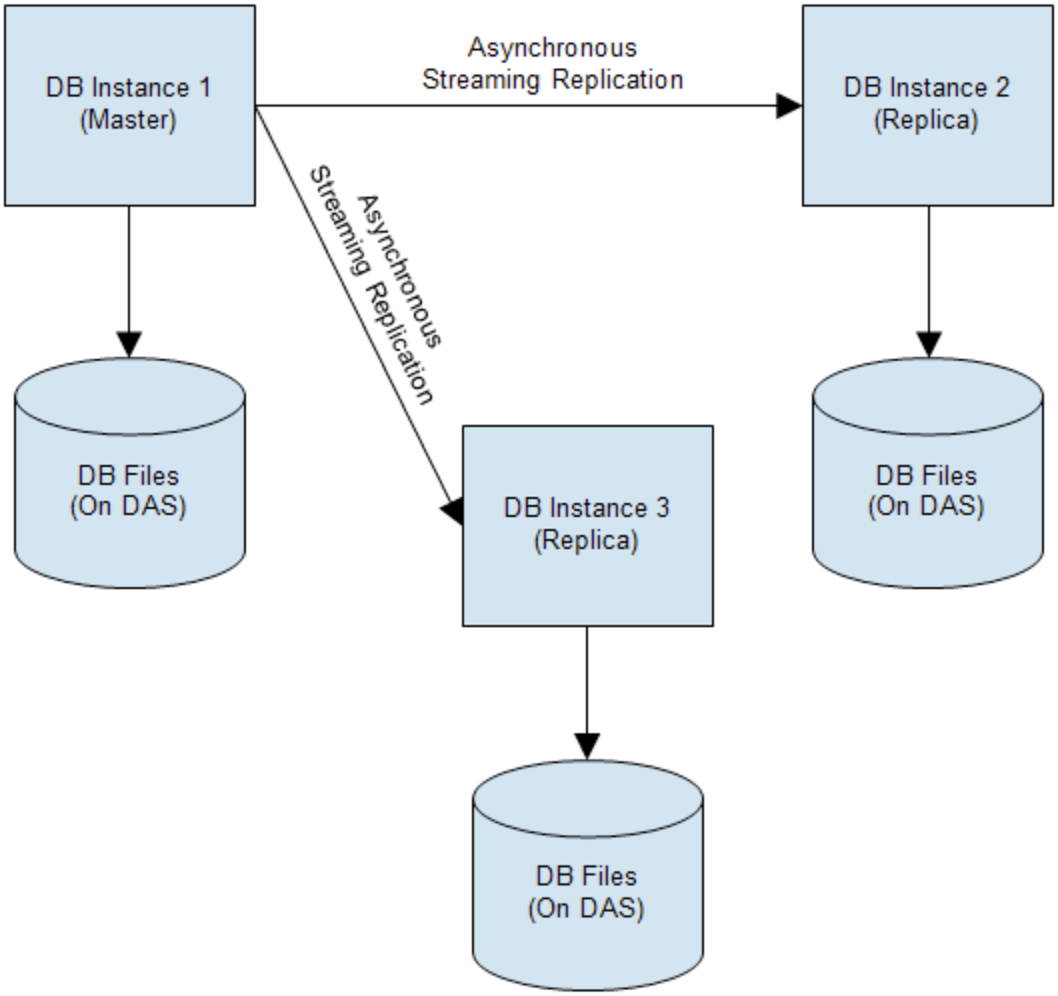

Figure 2.4 – Asynchronous Streaming Repliction on DAS

A load balancer and query router (*e.g.*, pgPool) can be used to distribute read queries to the replicas. Replicas are used for reporting and offloading of operational read queries, or for high availability purposes.

This scenario carries a potential for data loss. Transactions could be lost if the master suffers catastrophic failure before all WAL files have been streamed to the replica and the DAS becomes unavailable.

2.2.3 Asynchronous Streaming Replication on SAN, with Off-site Replica

By combining asynchronous streaming replication with SAN technology, and by moving one (or more) of the replicas off-site, you can create a very robust high-availablility and disaster recovery architecture that is protected from data loss (unless the SAN suffers a complete failure).

Data loss is extremely unlikely when asynchronous replication with direct-attached storage is in use. Additionally, the use of asynchronous streaming eliminates the potential impact on the transaction processing speed, because the master does not have to wait for the replica to commit transactions.

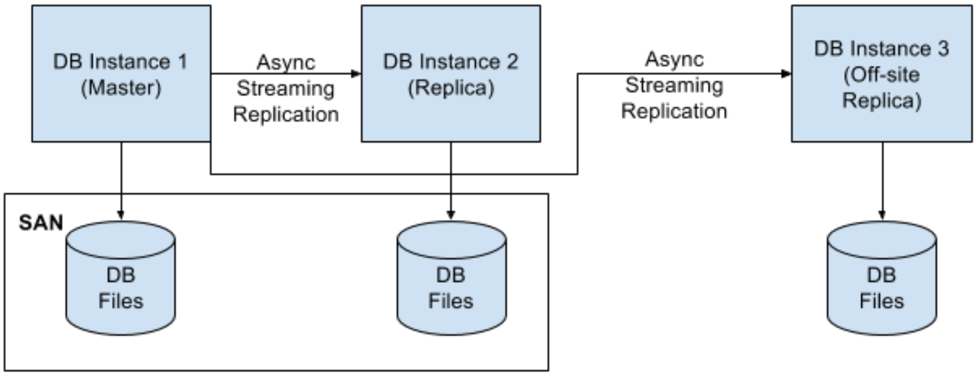
DB Instance 1 (Master)
Async Streaming Replication
DB Instance 2 (Replica)
Async Streaming Replication
DB Instance 3 (Off-site Replica)
SAN
DB Files
DB Files
DB Files

Figure 2.5 – Asynchronous Streaming Replication on SAN, with Off-site Replica

The off-site DR instance provides robust disaster recovery in the event the primary site fails. If the primary site suffers a catastrophic failure, with a complete loss of the SAN, then a small set of transactions (a few seconds worth) may be lost due to an interruption in streaming to the off-site replica. This is generally considered an extremely unlikely event; the potential loss of a few seconds worth of transactions is often regarded as an acceptable risk.

2.2.4 Synchronous Streaming Replication on DAS, with Off-site Replica

By combining local synchronous replication on a low-latency, highly reliable network with an offsite asynchronous replica, you can take advantage of each system’s strengths. This architecture lessens exposure to the throughput impact potentially caused by network latency during synchronous streaming replication.

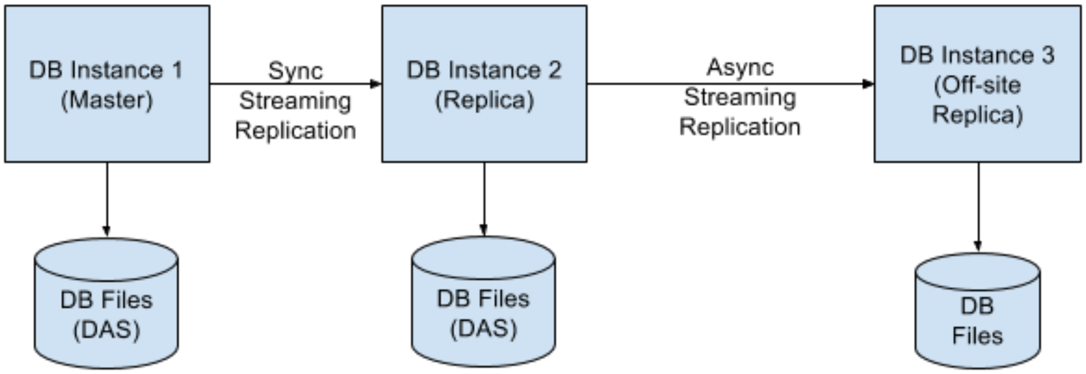

Figure 2.6 – Synchronous Streaming Replication on DAS, with Off-site Replica

The first replica within the data center ensures that there is minimal network latency and thus can operate in synchronous mode. This assures good performance as well as zero data loss. The second replica, which resides off-site, provides read scalability, high availability, and disaster recovery. The off-site replica can be configured to replicate directly off the master or configured as a cascading replica. The second architecture is not currently supported for automatic failover in EDB Postgres Failover Manager.

This architecture is similar to the Oracle® Dataguard Far Sync, introduced in Oracle Database version 12c.

2.3 EFM and Streaming Replication

EDB Postgres Failover Manager (EFM) provides failover infrastructure for streaming replication clusters. EFM monitors the members of a streaming cluster (asynchronous or synchronous, remote or local, SAN or DAS) for failure of the master, and promotes a replica node to become the new master should the master fail.

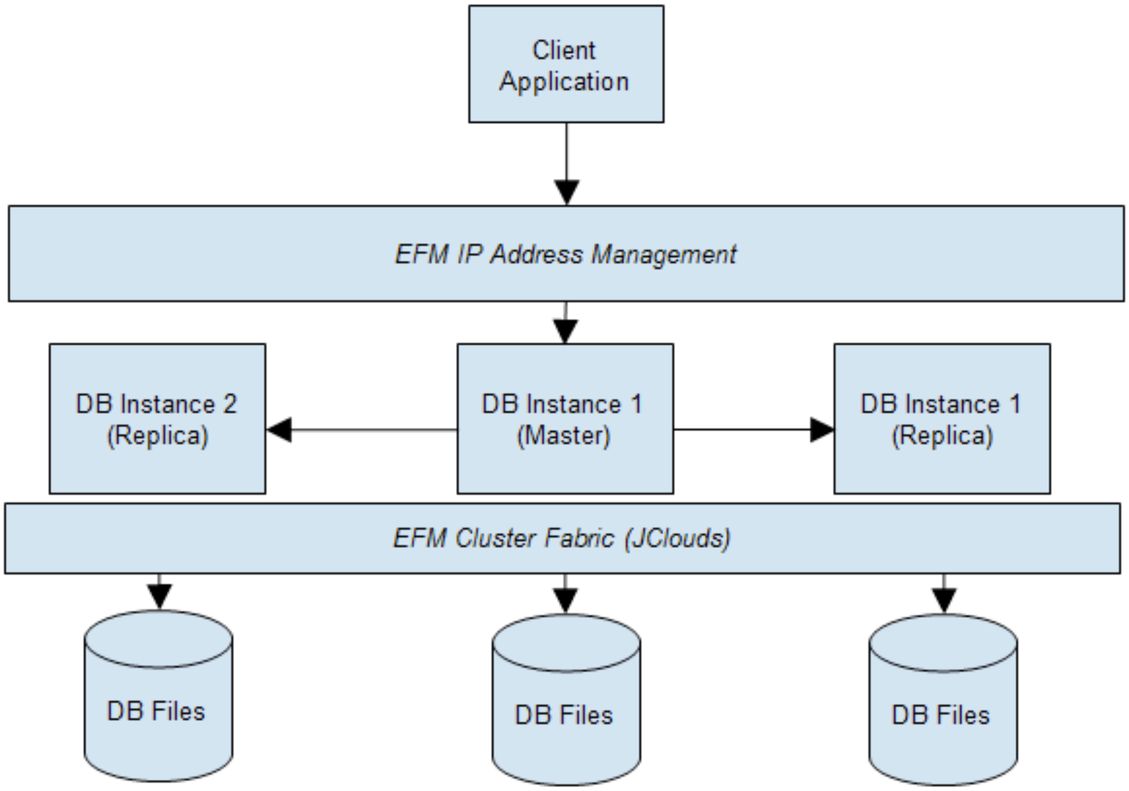

Figure 2.7 – EFM and Streaming Replication

EFM uses virtual IP address assignments to ensure that only one master can be active at one given point in time.

2.4 Logical Replication Clusters

Logical replication clusters use logical decoding functionality introduced in Postgres 9.4. Postgres logical decoding provides a way to use SQL statements to replicate objects from a master node to a standby node. Changes to the master node are sent to the replica node in a stream, identified by a logical replication slot.

A logical replication cluster does not require that the participating masters are the binary equivalent of their standby node. Logical replication can also be used to support system maintenance (*e.g.*, operating system upgrades) or Postgres major version upgrades.

Logical replication with conflict detection does impact transaction throughput and latency.

In this document we describe the use of EDB Postgres Replication Server 6.X (xDB).

2.4.1 Multi-master Replication (MMR)

In a Multi-master replication scenario, multiple master nodes propagate their changes via logical decoding (set the WAL log level to `logical`) to a central Replication Server. The replication server (after checking for conflicts) uses JDBC to propagate the changes to the other cluster participants.

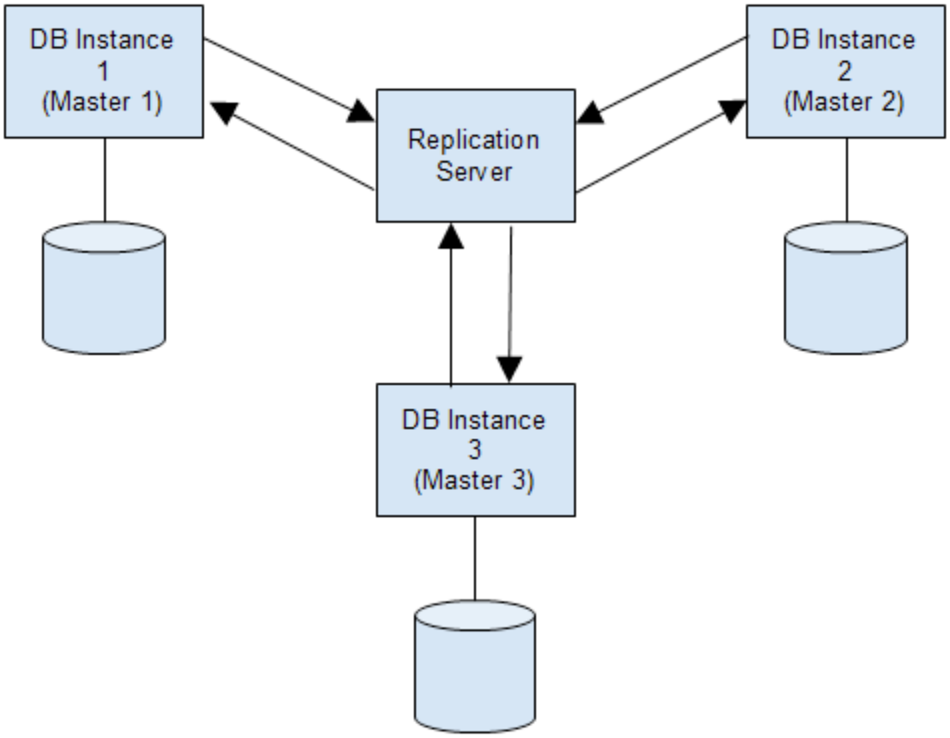

Figure 2.8 – Multi-master Replication

The performance of MMR architectures depends on the number of active masters. Conflict detection operations have a noticeable impact on transaction throughput and latency.

MMR architectures are often used in situations where data sharing is restricted to local instances (e.g., west coast customers versus east coast customers), but a holistic picture of the business must be available at all times.

2.4.2 Single-master Replication (SMR)

In a single-master replication scenario, a single master node propagates its changes via logical decoding (set the Postgres WAL log level to `logical`) to a central replication server. The replication server uses JDBC to propagate the changes to the other cluster participants.

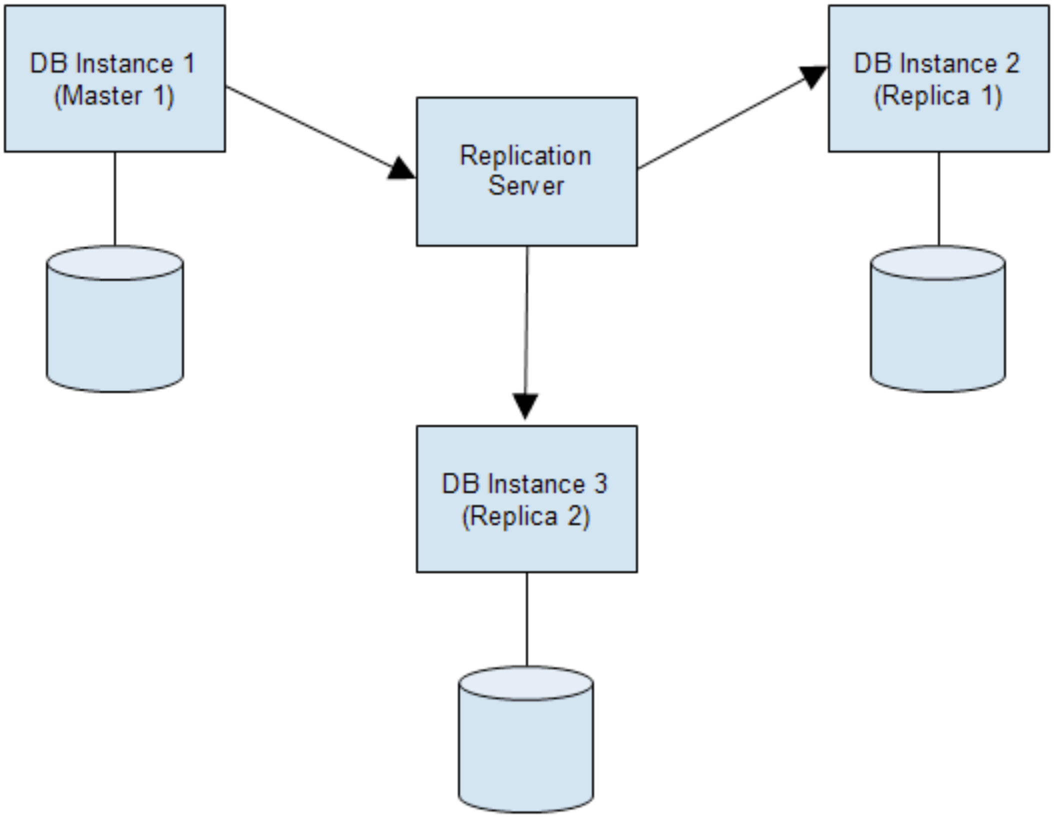

Figure 2.9 – Single-master Replication

SMR configurations are often used during system upgrades.

2.5 Trigger-based Replication Clusters (TRC)

Trigger-based replication clusters can be built on older versions of EDB Postgres Replication Server (5.0 and 5.1) or Slony. We will only consider Slony-based approaches in this document; for EDB Replication Server, we recommend using log-based replication as it is faster and easier to manage. If the system runs on recent Postgres releases (9.4 and 9.5), then log-based replication, available through EDB Replication Server, is the recommended approach.

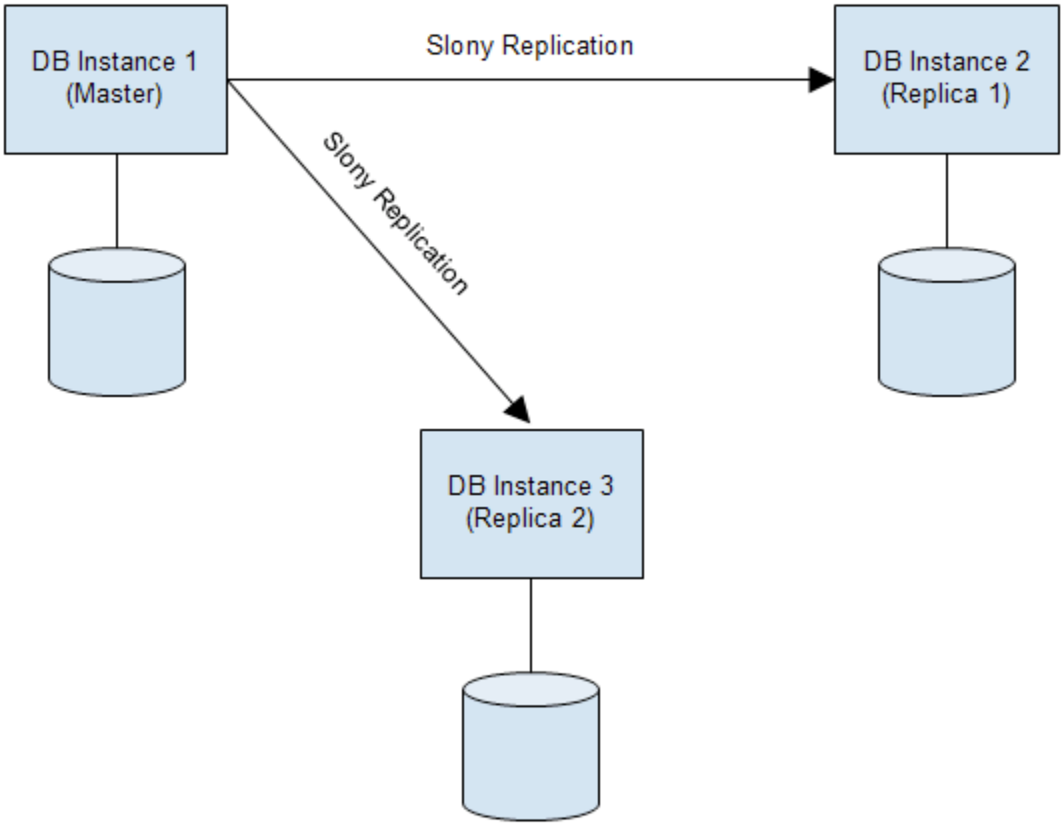

Figure 2.10 – Trigger-based Replication Clusters

A single master propagates its changes to one or several replicas. Slony replication can be used to support high availability and disaster recovery (though there is a lag that must be monitored closely) with near-zero downtime updates and upgrades.

Slony is a trigger-based solution that increases the workload on the master.

2.6 Hybrid Clusters

A hybrid cluster integrates logical replication and streaming replication to create an architecture that combines the strengths of both approaches.

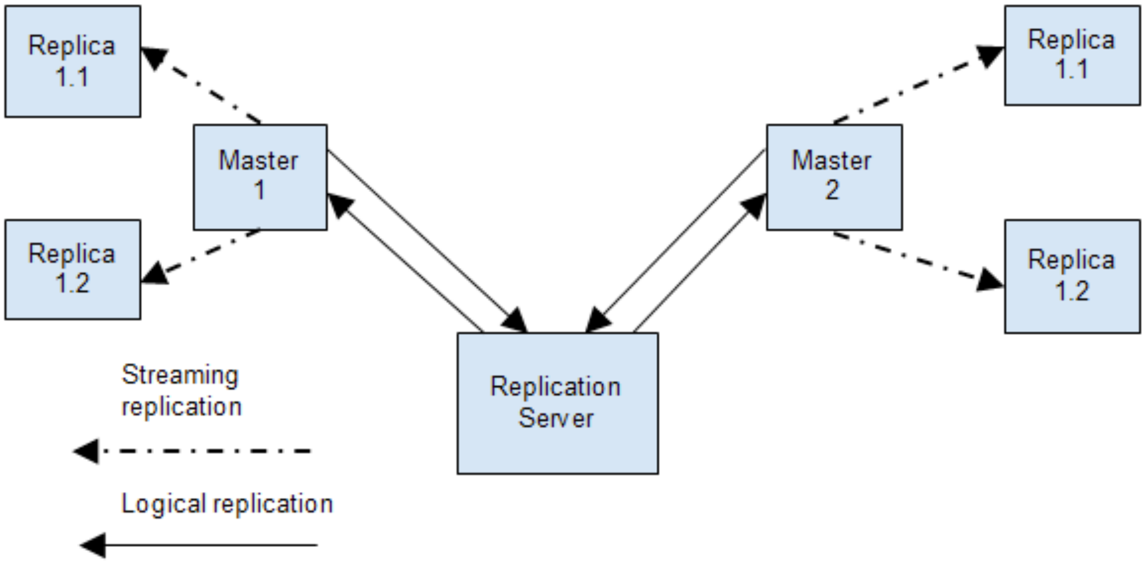

Figure 2.11 – Hybrid Clusters

Two streaming clusters (asynchronous or synchronous) are connected via a centralized logical replication server. Both streaming clusters act as masters and can process read/write transactions, while replicas can be used to offload read transactions. Conflicts are detected (and managed) by the replication server.

Hybrid configurations can be used to create geographically distributed clusters. They work best when the data between both sub-clusters have been logically sharded (to minimize the number of conflicts).

3 Key Capabilities and Characteristics of Postgres Clusters

The following table includes a high-level list of the characteristics of different clustering techniques. Not every application will require all capabilities, and depending on the use case, some capabilities may be more important than others.

Capability	Key Characteristic
High availability (HA)	Speed of failover (including error detection) and minimal (no) data loss
Off-site disaster recovery (DR)	Speed of failover to the DR site and minimal (no) data loss
Read scalability (RS)	Ability to offload read-only workloads with minimal (or no) lag time
Write scalability (WS)	Ability to scale the number of read/write transactions by adding database instances to the cluster
Near-Zero Downtime Update (NZD UD)	Ability to perform a minor version update (e.g. for Postgres 9.5.1 to 9.5.2) with minimal downtime.
Near-Zero Downtime Upgrade (NZD UG)	Ability to perform a major version upgrade (e.g. for Postgres 9.4 to 9.5) with minimal downtime
Near-Zero Downtime System Maintenance (NZDM)	Ability to perform OS and hardware maintenance with minimal downtime
Supported on commodity hardware (CH)	Does not require specialized storage or hardware components; can be implemented on major public cloud infrastructures.
Throughput and Latency (TL)	Impact of the clustering technology on the transaction load on the master
Cost ($)	Additional hardware cost

Feature Availability by Cluster Architecture

The table that follows rates the availability of each capability to a specific architecture. Within the table, we assign a rating for each capability and include a reference note about how the architecture uses the capability. Refer to the **Notes** section that follows the table for more information.

The following table uses the rating system below:

0 ─ not available, or not an optimal choice
1 ─ not the best choice
2 ─ best choice

	Shared Disk	Shared Disk w.off site DR	Sync SR	Async SR	Async SR w. SAN and Off-site DR	Sync SR w. DAS and Off-site DR	MMR	SMR	Slony TRC	Hybrid
HA	2[1]	2[11]	2[21]	1[31]	2[41]	2[51]	1[61]	1[71]	1[81]	2[91]
DR	0[2]	2[12]	2[22]	2[32]	2[42]	2[52]	1[62]	2[72]	1[82]	2[92]
RS	0[3]	0[13]	2[23]	1[33]	2[43]	2[53]	1[63]	1[73]	1[83]	2[93]
WS	0[4]	0[14]	0[24]	0[34]	0[44]	0[54]	0[64]	0[74]	0[84]	0[94]
NZD UD	1[5]	1[15]	2[25]	2[35]	2[45]	2[55]	2[65]	2[75]	2[85]	2[95]
NZD UG	0[6]	0[16]	0[26]	0[36]	0[46]	0[56]	2[66]	2[76]	2[86]	2[96]
NZD M	1[7]	1[17]	1[27]	1[37]	1[47]	1[57]	2[67]	2[77]	2[87]	2[97]
CH	0[8]	0[4][18]	2[28]	2[38]	0[48]	2[58]	2[68]	2[78]	2[88]	2[98]
TL	2[9]	2[19]	1[29]	2[39]	2[49]	2[59]	1[69]	2[79]	1[89]	1[99]
$	0[10]	0[20]	2[20]	2[40]	1[50]	2[60]	1[70]	1[80]	2[90]	1[100]

Notes

(1) Very fast detection of cluster failure and fast switch over.

(2) All components must be in the same DC; this does not provide good DR.

(3) Only one member of the cluster is active at the same time.

(4) Only one member of the cluster is active at the same time.

(5) Most minor upgrades can be executed as the on-disk format does not change, except if the upgrade addresses disk format issues.

(6) Both cluster members have to be on the same major version.

(7) Any maintenance on the OS that does not impact the on-disk storage can be executed.

(8) Requires proprietary SAN technology, which is not available in public clouds.

(9) Shared disk clustering does not reduce the transaction throughput and does not introduce additional latency.

(10) Requires proprietary SAN technology, which tends to be much more expensive than a second server with direct attached storage.

(11) Very fast detection of cluster failure and fast switch over.

(12) Streaming replication (synchronous or asynchronous) provides an effective off-site DR mechanism.

(13) Only one member of the cluster is active at the same time, except if the off-site replica is used for read traffic (in which case it is not a real DR-only solution).
(14) Only one member of the cluster is active at the same time.
(15) Most minor upgrades can be executed as the on-disk format does not change, except if the upgrade addresses disk format issues.
(16) All cluster members have to be on the same major version.
(17) Any maintenance on the OS that does not impact the on-disk storage can be executed.
(18) Requires proprietary SAN technology, which is not available in public clouds.
(19) Shared disk clustering and streaming replication do not reduce the transaction throughput and do not introduce additional latency.
(20) Requires proprietary SAN technology, which tends to be much more expensive than a second server with direct-attached storage.
(21) This solution guarantees that that there will be no loss of data. In combination with EFM, you can achieve a robust HA solution.
(22) Streaming replication provides a very good and robust off-site DR model, guarantees that that there will be no loss of data. However, synchronous streaming over the WAN may lead to significant delays.
(23) Synch SR provides a very popular read scalability solution as the replica is in synch with the master.
(24) Only one member of the cluster is accepting update transactions at any given point in time.
(25) The system can be switched over to the replica to allow for updating of the master, and then switched back. Details are described here on the EDB Blog post titled *Switchover/Switchback in PostgreSQL 9.3*. Automatic switchover/switchback will be one of the key capabilities in EFM 2.1.
(26) All cluster members have to be on the same major version in a streaming replication environment.
(27) Binary equivalence of the OS must be maintained, including all details of the collation definitions.
(28) This solution is fully supported in all major public clouds.
(29) Synchronous replication will introduce transaction delay; network latency may exacerbate this. This solution should be used over a WAN. Timeout settings (and occurrence of timeouts) must be managed carefully.
(30) Very cost effective solution that can be implemented on commodity hardware.
(31) This solution provides a popular HA solution; however under certain circumstances a loss of data can occur during failover. In combination with EFM, one achieves a popular HA solution.
(32) Streaming replication provides a very good and robust off-site DR model.
(33) Asynch SR provides a very popular read scalability solution. The read replica may be slightly delayed.
(34) Only one member of the cluster is accepting update transactions at any given point in time.
(35) The system can be switched over to the replica to allow for updating of the master, and then switched back. Details are described here on the EDB Blog

post titled *Switchover/Switchback in PostgreSQL 9.3*. Automatic switchover/switchback will be one of the key capabilities in EFM 2.1.

(36) All cluster members have to be on the same major version in a streaming replication environment.

(37) Binary equivalence of the OS must be maintained, including all details of the collation definitions.

(38) The solution can be implemented in the major public cloud services.

(39) Asynchronous streaming replication introduces negligible impact on transactions load on the master and does not add additional latency to the transactions.

(40) A proven, cost effective solution that is used frequently on premises and in the cloud.

(41) The SAN helps avoid loss of data during switchover, except in the case of a catastrophic unrecoverable failover of the SAN.

(42) Streaming replication provides a very good and robust off-site DR model. A loss of data is extremely unlikely and will only occur in the event of an unrecoverable (catastrophic) failure of the SAN while transactions have not been streamed to the replica before system failure.

(43) Asynchronous SR provides a very popular read scalability solution. Reads may be slightly delayed if the replication is delayed or the replica has not caught up yet.

(44) Only one member of the cluster is accepting update transactions at any given point in time.

(45) The system can be switched over to the replica to allow for upgrading of the master, and then switched back. Details are described here on the EDB Blog post titled *Switchover/Switchback in PostgreSQL 9.3*. Automatic switchover/switch back will be one of the key capabilities in EFM 2.1.

(46) All cluster members have to be on the same major version in a streaming replication environment.

(47) Binary equivalence of the OS must be maintained, including all details of the collation definitions.

(48) This solution requires proprietary hardware (SAN).

(49) Asynchronous streaming replication introduces negligible impact on transactions load on the master and does not add additional latency to the transactions.

(50) This solution requires proprietary hardware (SAN).

(51) Synchronous replication provides a very reliable HA solution.

(52) Streaming replication provides a very good and robust off-site DR model. A loss of data is unlikely and will only occur in case both the master and the first (onsite) replica are lost and cannot be recovered.

(53) Synch SR provides a very robust read scalability solution.

(54) Only one member of the cluster is accepting update transactions at any given point in time.

(55) The system can be switched over to the replica to allow for upgrading of the master, and then switched back. Details are described here on the EDB Blog

post titled *Switchover/Switchback in PostgreSQL 9.3*. Automatic switchover/switchback will be one of the key capabilities in EFM 2.1.

(56) All cluster members have to be on the same major version in a streaming replication environment.

(57) Binary equivalence of the OS must be maintained, including all details of the collation definitions.

(58) This solution is fully supported in all major public clouds.

(59) Local synchronous replication introduces minimal impact on transaction load on the master, and adds a small additional latency to the transactions.

(60) Very cost effective solution that can be implemented on commodity hardware.

(61) Synchronization delays caused by latencies at the central replication server can lead to situations where masters are out of sync.

(62) Synchronization delays caused by latencies at the central replication server can lead to situations where the master copies are out of sync.

(63) See the section that follows about MMR and Scalability.

(64) See the section that follows about MMR and Scalability.

(65) MMR clustering allows cluster members to be on different major and minor Postgres versions.

(66) MMR clustering allows cluster members to be on different major and minor Postgres versions.

(67) MMR clustering allows cluster members to be on different hardware and OS versions.

(68) This solution is fully supported in all major public clouds.

(69) Conflict detection and conflict resolution have a negative impact on overall transaction throughput. A larger number of active masters intensifies the latencies.

(70) The replication server adds additional cost to the architecture.

(71) Initial performance tests show that logical SMR compares well to asynchronous streaming replication. When using direct-attached storage (DAS), a loss of data could occur if transactions have not been streamed to the replication server before system failure.

(72) Logical SMR provides a very fast and robust off-site DR model. When using direct attached storage (DAS), a loss of data could occur if transactions have not been streamed to the replica before system failure. In many use cases this appears to be an acceptable risk, possibly because SANs tend to be used at the main DC for environments that require an off-site DR solution.

(73) Logical SMR provides a strong read scalability solution. Reads may be slightly delayed if the replication is delayed or the replica has not caught up yet. Replication delays are not expected to be significantly larger than on streaming replication.

(74) Only one member of the cluster is accepting update transactions at any given point in time.

(75) SMR clustering allows cluster members to be on different major and minor Postgres versions.

(76) SMR clustering allows cluster members to be on different major and minor Postgres versions.

(77) SMR clustering allows cluster members to be on different hardware and OS versions.
(78) This solution is fully supported in all major public clouds.
(79) SMR has a negligible impact on latency and has no impact on throughput on the master.
(80) The replication server adds additional cost to the architecture.
(81) Slony replication introduces delay, and not all transactions may have been replicated to the HA replica.
(82) Slony replication introduces delay, and not all transactions may have been replicated to the DR replica.
(83) Slony replication introduces delay, and not all transactions may have been replicated to the HA replica.
(84) Only one member of the cluster is accepting update transactions at any given point in time.
(85) Slony is a proven choice for minor version update.
(86) Slony is a proven choice for major version update.
(87) Slony clustering allows cluster members to be on different hardware and OS versions.
(88) This solution is fully supported in all major public clouds.
(89) Slony replication triggers have a negative impact on transaction throughput on the master.
(90) Slony does not require a separate replication server.
(91) When combined with synchronous streaming replication, The Hybrid Cluster provides excellent HA.
(92) When combined with synchronous streaming replication, The Hybrid Cluster provides excellent DR.
(93) When combined with synchronous streaming replication, The Hybrid Cluster provides excellent read scalability.
(94) See section below about MMR and Scalability.
(95) The MMR component allows both streaming clusters to be updated separately.
(96) The MMR component allows both streaming clusters to be upgraded separately.
(97) The MMR component allows both streaming clusters to operate on different hardware and software versions.
(98) This solution is fully supported in all major public clouds.
(99) Conflict detection and conflict resolution have a negative impact on overall transaction throughput. A larger number of active masters may intensify the latency.
(100) The replication server adds additional cost to the architecture.

4 MMR and Scalability Discussion

This section discusses a frequent question,*"Does MMR provide write scalability?"* and describes why an MMR solution will not provide write scalability. MMR is a great solution for geographically distributed systems, for NZD upgrade and NZD update, but it does not provide write scalability. In fact, it reduces overall transaction throughput.

Let's assume Server A and Server B are part of the same 2-node MMR cluster:

Scenario 1 ─ The application cannot segregate read transactions from write transactions

If A gets 70 tps read/write (RW) from the application pool and B gets 30 tps RW from the application pool at a steady rate, and if replication delay is 1 second, then A and B will see a total load of 100 tps RW each (after the first second).

Thus, the load is equal to sending 100 tps (RW) to A, and using B as a hot stand-by, getting no transactions at all.

Scenario 2 ─ Read transactions can be separated from write transactions

Server A gets 20 tps RW, 50 tps RO (read only) from the application pool; B get 10 tps RW, 20 tps RO from the application pool at a steady rate. If MMR replication delay is 1 second, then A gets a total of 30 tps RW, 50 tps RO and B gets a total 30 tps RW, 20 tps RO.

In a streaming replication model where all RW transactions would go to the master, A (the master) would get 30 tps RW, and the replica (B) would get 70 tps RO.

Assuming that the application could differentiate RW from RO transactions, and use the JDBC driver accordingly, then this will provide higher scalability (as all RO transactions are shunted off to the replica).

MMR Scalability Conclusions

While MMR provides architectural advantages, such as NZD upgrade or geographic distribution of the database, it does NOT provide write scalability of the database.

5 Solution Discussion

Asynchronous streaming replication, with SAN and off-site disaster recover, combined with monitoring by EDB Failover Manager, is a very popular solution for mission-critical enterprise solutions that cannot suffer any loss of data. Because of the SAN requirement, this solution is limited to on-premises deployments.

In cloud deployments, a combination of local (within the cloud region or availability zone) synchronous streaming replication with off-site (different region) disaster recovery via asynchronous streaming replication is used to achieve similar results while avoiding the performance impact of synchronous replication over the WAN.

High Availability through shared disk-based systems is often the preferred solution, where a Veritas or Red Hat Cluster is the corporate standard. While this used to be a very popular solution up until three years ago, EDB Failover Manager has gained significant popularity, providing health monitoring, failure detection, and automatic failover mechanisms.

MMR/SMR solutions are finding significant interest in applications where system maintenance must be executed without significant downtime. Often, MMR/SMR solutions are introduced for the upgrade/update timeframes only. MMR/SMR also continues to be used successfully for geographically distributed solutions.