International Journal of Core Engineering & Management
Volume-7, Issue-04, 2022 ISSN No: 2348-9510

DATA WAREHOUSING IN THE MODERN ERA: A COMPARISON OF AMAZON REDSHIFT AND GOOGLE BIGQUERY

Srinivasa Rao Karanam
Srinivasarao.karanam@gmail.com
New Jersey, USA

Abstract

Data warehousing has undergone extraordinary shifts over the last decade, especially as organizations attempt to harness rapidly accumulating data for deeper insights and quicker decision-making. In this extended paper, we delve thoroughly into the comparative intricacies of Amazon Redshift and Google BigQuery, two major forces in the domain of cloud-based data warehousing. We aim to unravel their fundamental architectural patterns, examine performance facets, explore cost frameworks, and investigate real-world applications. By weaving in consideration of concurrency challenges, advanced integration possibilities, and operational overhead, this discussion elaborates the respective strengths and shortfalls of these platforms. The ultimate objective is to empower data engineers, architects, and business stakeholders with a robust understanding to direct the selection of best-suited solutions for large-scale analytics in an ever-evolving ecosystem.

Keywords: Data warehousing, Amazon Redshift, Google BigQuery, cluster-based architecture, serverless approach, concurrency scaling, MPP architecture, pay-as-you-go, real-time analytics, cost optimization

I. INTRODUCTION

The unstoppable expansion of digital data reflects an era wherein institutions from diverse domains are inundated with both structured and unstructured data. E-commerce logs, social network posts, sensor streams from IoT devices, transaction-based data flows, and external feed from third-party sources have spurred the demand for advanced data management solutions. As companies expand, the impetus for actionable intelligence, real-time analytics, and predictive modeling intensify drastically.

Historically, the concept of data warehousing revolved around orchestrating data from multiple operational systems into a centralized repository that was designed to run complex queries effectively. On-premises data warehouse solutions typically relied on specialized hardware appliances or carefully tuned RDBMS setups which, while efficient for a narrower scope, often fell short in cost efficiency or scale agility under changing loads.

It is evident that cloud computing has established new paradigms in data warehousing, offering near-limitless expansions, pay-as-you-use costing models, and a managed environment that relieves many operational burdens. At the forefront of this shift, Amazon Redshift (introduced 2013) and Google BigQuery (debuted 2011) stand as exemplars, each representing distinct philosophies of cloud data warehousing. Redshift championed a cluster-based MPP (Massively Parallel Processing) architecture, while BigQuery introduced a serverless design. This paper devotes extensive attention to the complexities entailed in adopting either approach, ensuring readers with a technical lens gain the deeper vantage they require when building or upgrading enterprise data infrastructures.

II. BACKGROUND ON MODERN DATA WAREHOUSING

Data warehousing in the modern sense no longer revolves around static ETL processes feeding star or snowflake schema. In the earlier period, warehousing solutions were built primarily for weekly or monthly aggregated reporting. However, the onslaught of real-time usage scenarios, requiring near-instant feedback from streaming data, has introduced new challenges.

Companies are frequently adopting multi-tiered architectures that incorporate data lakes or data lakehouses in addition to conventional warehouses. While a data warehouse remains integral for high-performance analytics, governed data transformations, and concurrency handling, the data lake can store a wide range of raw data (including CSV, JSON, or parquet) in cheap, highly durable cloud storage. This separation of compute from storage facilitates flexible processing frameworks for batch or real-time tasks.

In advanced industries, the warehouse sits at the core, bridging data science, business intelligence, and advanced analytics workloads. The capacity to run complex SQL queries at scale, integrate with ML pipelines, and manage concurrency for multiple data consumers defines a modern solution. Additionally, aspects like strong security, compliance with sector-specific regulations, and integrated governance are no longer optional but demanded. The evolution of Amazon Redshift and Google BigQuery reflect this shifting environment. They each have developed specialized features to address the multi-faceted nature of modern data warehousing, from columnar storage and on-demand scaling to serverless query engines and integrated data catalogs.

III. OVERVIEW OF AMAZON REDSHIFT

Launched officially in 2013, Amazon Redshift was conceived as a means to democratize advanced warehousing capability in the AWS ecosystem. Instead of requiring expensive hardware appliances or complicated MPP systems, Redshift introduced a managed cluster approach that seamlessly connected with other AWS services.

Redshift employs a column-oriented store, which allows scanning only the relevant columns for analytic queries, reducing I/O overhead significantly. This design integrates compression techniques, frequently offering an order-of-magnitude improvement in query performance for

typical analytics workloads. The cluster itself is composed of at least one leader node, tasked with orchestrating queries, and one or more compute nodes, each storing a portion of the data.
Scaling out in Redshift typically involves adding more compute nodes or switching to a different node type. This process can cause data redistribution, incurring downtime or partial unavailability, requiring administrators to plan meticulously for usage spikes. To handle concurrency bursts, Redshift introduced a feature known as Concurrency Scaling, which spins up additional transient clusters behind the scenes to service queries when concurrency thresholds are reached. While extremely beneficial, this concurrency scaling can have cost implications if usage exceeds the free credits allocated to the primary cluster.

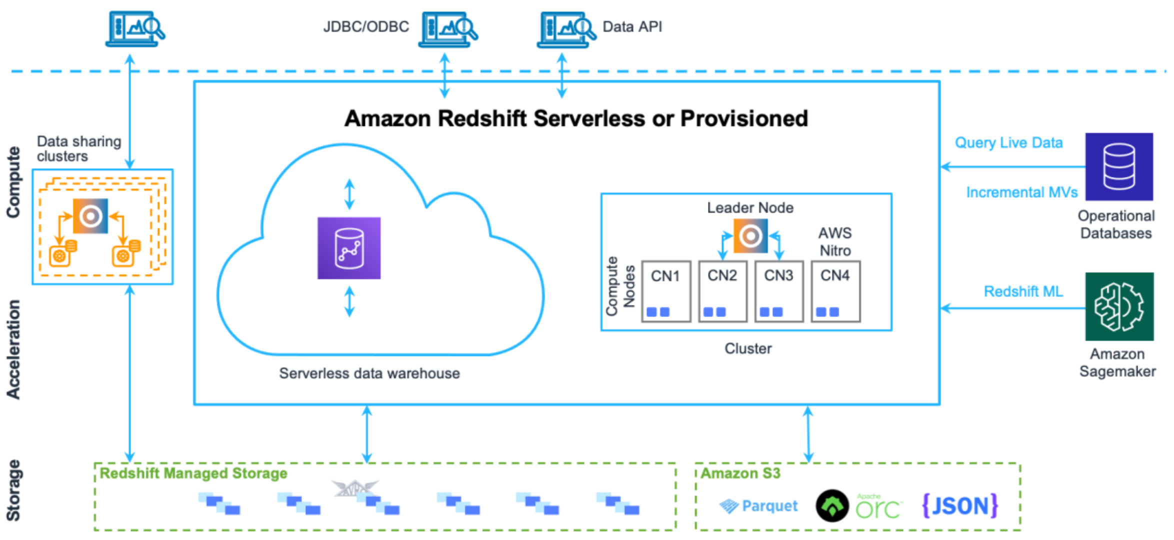

Figure 1: Illustration of Amazon Redshift architecture, showcasing serverless and provisioned deployment options, data sharing clusters, compute and storage layers, integration with AWS services, and support for various data formats.

Amazon Redshift ties deeply into the AWS ecosystem. A hallmark feature is Redshift Spectrum, which enables queries over data residing in Amazon S3, effectively bridging a data lake approach with the data warehouse cluster. Glue integration helps automate data cataloging and schema discovery. For real-time analytics, AWS Kinesis can feed data into S3 or directly into Redshift. BI solutions such as Amazon QuickSight, Tableau, or Looker have native connectors to Redshift, simplifying analytics consumption.
Though Redshift is fully managed, the user must still handle tasks such as designing table distribution keys, sort keys, and performing VACUUM operations to reclaim space from deleted rows. The complexity is not non-trivial: poor choice of distribution key might lead to data skew, causing performance degradation. Similarly, concurrency scaling, WLM (Workload Management) queue configurations, and relevant cluster parameter must be tuned for optimal

International Journal of Core Engineering & Management
Volume-7, Issue-04, 2022 ISSN No: 2348-9510

throughput. Over time, new Redshift features like Automatic Table Optimization have alleviated some administrative overhead, but a robust skill in data warehousing fundamentals remains advisable to harness Redshift effectively.

IV. OVERVIEW OF GOOGLE BIGQUERY

In 2011, Google introduced BigQuery, a serverless data analytics platform that drastically differs from the cluster-based approach. Instead of requiring the user to provision or maintain nodes, BigQuery decouples compute and storage, letting Google's internal, massively distributed infrastructure handle resource assignment. The overarching objective is to let the end user focus purely on the data and queries rather than operational details.

At the heart of BigQuery is the philosophy that the cloud provider, in this case Google, handles hardware provisioning, concurrency, and scaling behind the scenes. A user can simply upload or reference data and run queries with standard SQL, leaving the underlying MPP infrastructure entirely abstracted. The advantage is that the user only pays for what they query or store. The ephemeral nature of provisioning allows BigQuery to spin up large compute resources to satisfy queries quickly, then free them up.

BigQuery leverages Google's proprietary columnar format. On the query side, Dremel is the distributed engine enabling parallel processing of enormous data volumes. The high-bandwidth, low-latency network infrastructure in Google data centers is a significant factor behind BigQuery's performance potential. Yet, performance depends heavily on how well queries are structured, how the user leverages partitioned tables, and how data is stored.

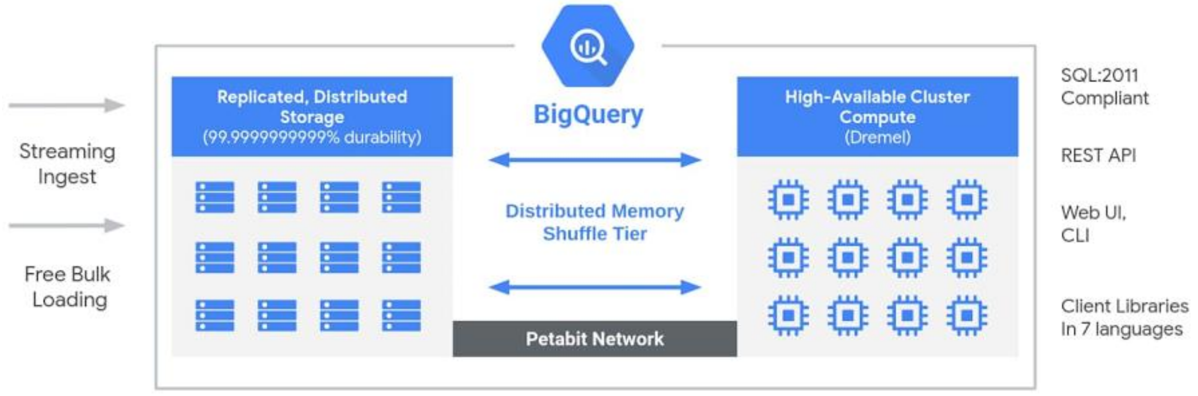
Streaming Ingest
Free Bulk Loading
Replicated, Distributed Storage (99.9999999999% durability)
BigQuery
Distributed Memory Shuffle Tier
Petabit Network
High-Available Cluster Compute (Dremel)
SQL:2011 Compliant
REST API
Web UI, CLI
Client Libraries In 7 languages

Figure 2: Illustration of Google BigQuery architecture, highlighting replicated distributed storage, high-availability cluster compute (Dremel), a distributed memory shuffle tier, and petabit network connectivity, along with support for SQL:2011, REST API, and multiple client interfaces.

The pay-as-you-go model in BigQuery charges based on the volume of data scanned by each query. This system can be cost-effective for organizations running sporadic queries. However, if

International Journal of Core Engineering & Management
Volume-7, Issue-04, 2022 ISSN No: 2348-9510

queries scan billions of rows repetitively or if job concurrency is extremely high, costs can escalate quickly. As an alternative, Google introduced flat-rate pricing, allowing users to purchase dedicated query slots on a monthly basis, guaranteeing stable performance and cost predictability.

BigQuery integrates seamlessly with Google Cloud Storage, enabling direct query on data stored there in various formats. Data can be loaded in streaming fashion as well, achieving near real-time updates for analytics. BigQuery ML, an integrated machine learning feature, is another distinctive advantage, allowing data scientists or even analytics-savvy business users to train ML models directly on warehouse data with standard SQL.

Although minimal cluster tuning is required, BigQuery imposes specific quotas, such as maximum query durations, concurrency limits, and daily usage caps for free tiers. Query performance might degrade if poorly designed SQL scans huge amounts of unfiltered data. Also, advanced DBAs or system architects might find the lack of hardware-level control limiting if they have specialized performance demands. The serverless nature, while easy to manage, restricts certain granular optimizations.

V. COMPARATIVE ANALYSIS

Amazon Redshift revolves around the concept of cluster provisioning, with each cluster sized for an expected peak load. This approach fosters a sense of dedicated environment, letting the user fine-tune distribution style. However, dynamic scaling remains more complicated, as resizing generally involves partial or full cluster re-allocation.

Google BigQuery, conversely, is natively serverless. The user is largely abstracted from scaling tasks. In an ideal scenario, if concurrent queries spike, BigQuery automatically spin up more compute behind the scenes. This elasticity is extremely convenient. The potential downside is that multi-tenancy might lead to performance variations at times of extreme global usage, albeit typically overshadowed by the inherent advantages of ephemeral scaling.

Performance in Redshift is reliant on how effectively data is distributed among nodes, how well sort keys align to typical queries, and the concurrency settings. Efficiently using column compression and ensuring minimal data skew lead to strong performance. Over time, Redshift has introduced features like Automatic Table Optimization, which tries to adapt table design dynamically, but results vary.

BigQuery's performance, on the other hand, leverages Google's Dremel engine and massive parallel scanning. In many real-world benchmarks, BigQuery can handle large ad-hoc queries swiftly. Yet performance degrades if queries read entire tables unnecessarily or if partition pruning is not used. Partitioning, clustering, and avoiding repeated large cross-joins remain key best practices. The inherent advantage for BigQuery is the ability to spin up large compute resources per query rather than rely on a fixed cluster.

Amazon Redshift pricing revolves around an hourly rate based on the node type. This can yield a stable monthly cost for heavy, consistent workloads. Even though concurrency scaling is free up to a limit, going beyond that can add cost. Redshift Spectrum usage for scanning external

International Journal of Core Engineering & Management
Volume-7, Issue-04, 2022 ISSN No: 2348-9510

data in S3 also adds separate charges based on the volume of data scanned.

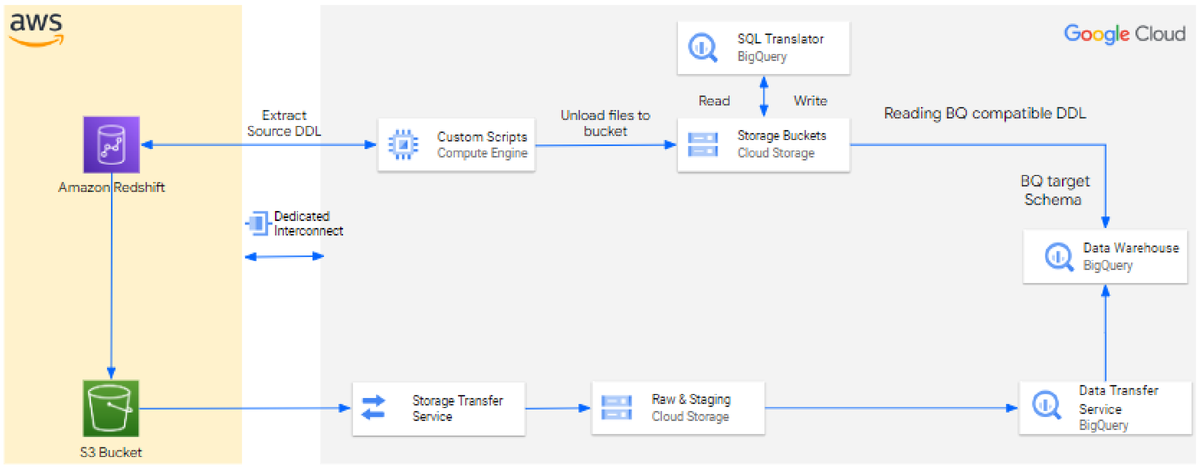

Figure 3: Illustration of AWS to Google Cloud data migration, depicting the transfer of data from Amazon Redshift and S3 to Google BigQuery using Storage Transfer Service, Compute Engine scripts, and SQL Translator, with intermediate staging in Cloud Storage and final loading into a BigQuery data warehouse.

BigQuery's default approach is on-demand: scanning costs per query are usually denominated in dollars per terabyte scanned. This is highly beneficial for use cases that query data intermittently. However, workloads with high query frequency scanning large data sets might see costs balloon. For these heavier workloads, BigQuery's flat-rate model is an alternative. Storage cost is separate, though generally competitive, especially with automatic discounts for data that remain unchanged for 90 days or more.
Redshift demands that an admin or data engineer carefully design table distribution, sorting, and occasionally conduct vacuum or analyze statements. Also, concurrency management with WLM queues can get intricate, especially if different workloads share the same cluster. This overhead might be acceptable if the organization wants that degree of control or if they have resources with the skill to continuously tune the environment.
BigQuery's serverless approach significantly reduces administrative overhead. The user can concentrate on data design, table partitioning, and writing cost-effective queries. However, for teams that want deeper control, the black-box nature can be restricting. They cannot choose specific node type or underlying hardware specs. They must rely on Google's platform to handle concurrency, memory, and CPU resource allocations.
Both solutions provide encryption at rest and in transit, with the possibility of employing customer-managed keys if desired. Amazon Redshift leverages VPC for network isolation, AWS IAM for identity management, and integrates with AWS CloudTrail for auditing and logging. Similarly, BigQuery is integrated with Google Cloud IAM, VPC Service Controls, and supports encryption options, ensuring data is protected. They also meet or exceed major compliance

standard (HIPAA, SOC 1/2/3, PCI DSS, ISO 27001, etc.). The choice generally revolves around the existing corporate environment.
As a part of AWS, Redshift links easily with S3, AWS Glue, EMR, Kinesis, and QuickSight. Third-party ETL tools such as Informatica, Talend, or Matillion also integrate natively. The synergy with other AWS components might be a deciding factor if the entire stack is already in AWS.
Google BigQuery, in turn, is deeply connected to GCP's portfolio: Cloud Storage, Dataflow, Dataproc, Pub/Sub, Vertex AI, and so forth. For data ingestion, one can create direct pipelines from GCP or external sources. For visualization, Google's Looker or third-party BI tools can be used. If the user organization is heavily invested in GCP, BigQuery is a natural extension for analytics.

VI. TYPICAL USE CASES

Redshift is frequently the choice for stable workloads that remain in the AWS environment. Retail companies, finance institutions, and healthcare providers with large volumes of structured data, generating complex queries, might enjoy the sense of control provided by cluster provisioning. If the usage is quite consistent, a fixed cluster can be cost-friendly, especially if Reserved Instances are purchased.
BigQuery, with its usage-based or flat-rate plans, is often more appealing for data science teams or organizations that run spiky or unpredictable workloads. The integrated ML features can reduce overhead for building predictive models. Ad-hoc queries scanning gigabytes to petabytes is done seamlessly, with minimal operational overhead. Enterprises that are already leveraging GCP for compute, storage, or AI solutions find BigQuery the path of least resistance to advanced analytics.
Migrating from on-premises or other data warehouse technologies might be simpler with Redshift if the source environment is a Postgres or a conventional RDBMS. Tools exist to shift schema and data. Yet, users must design new distribution strategies to harness Redshift's performance.
Transitioning to BigQuery often revolves around converting ingestion scripts, ensuring partition logic is aligned to BigQuery's approach, and rewriting some SQL features if the original environment used stored procedures or custom indexes. Some legacy ETL tools might require reconfiguration for GCP. Once integrated, the user no longer worries about node provisioning or cluster expansions, which is often a net benefit if the cost is properly managed.

VII. FUTURE OUTLOOK AND INNOVATIONS

The future direction for Amazon Redshift is strongly oriented around making the platform more dynamic and integrated with data lake workflows. With more frequent data streaming from IoT endpoints or real-time events, we might see further improvements in concurrency scaling, cost governance, and automatic schema optimization. There is an impetus to reduce manual overhead in cluster management by introducing more sophisticated AI-driven optimizations or ephemeral usage patterns.

International Journal of Core Engineering & Management
Volume-7, Issue-04, 2022 ISSN No: 2348-9510

Google BigQuery, by contrast, will likely progress deeper into ML, real-time analytics, and cross-platform synergy. The serverless approach is increasingly attractive for organizations lacking specialized DBA resources. The platform may introduce advanced caching or query acceleration layer to further reduce scanned data volumes, mitigating the cost pitfalls of scanning massive tables. With multi-cloud or hybrid strategies gaining steam, BigQuery's connections to external data storages might also expand.
Simultaneously, competitor solutions in the data warehousing space, such as Snowflake or Azure Synapse, push these market leaders to innovate rapidly. The lines between operational and analytical workloads are blurring, giving rise to a new generation of data platforms that unify transactional and analytics capabilities. The ongoing expansion of machine learning, streaming ingestion, and data sharing marketplace ensures that both Redshift and BigQuery will continue to incorporate new features, refine performance, and lower total cost of ownership.

VIII. CONCLUSION

In an era saturated with data, the question is not merely how to store information, but how to orchestrate large-scale analytics in a robust, cost-effective, and near real-time manner. Amazon Redshift and Google BigQuery have emerged as formidable solutions in the cloud data warehouse arena, each underpinned by a distinct architectural philosophy. Redshift's cluster-based model resonates with enterprises seeking granular controls, strong synergy with AWS, and consistent workloads. BigQuery's serverless design resonates with organizations favoring minimal operations overhead, ephemeral high concurrency, and pay-as-you-query cost structures.
Which system is superior? The reality is that such a question lacks a universal answer. Enterprises must weigh cost patterns, skill sets, existing ecosystem, security demands, and concurrency patterns. For some, the stable nature of Redshift, combined with AWS integration, is a boon. For others, the near-limitless elasticity and serverless approach of BigQuery overshadow the constraints of cluster-based provisioning. With both platforms continuing to incorporate advanced features that address data governance, machine learning, and streaming workloads, the final decision is deeply contextual.
Nonetheless, from a high-level vantage, Redshift remains well-suited for companies with established presence in AWS or a preference for cluster-based data warehouse approaches. Conversely, BigQuery is an appealing choice for those seeking to minimize overhead, to pay primarily for usage, or to plug into the dynamic Google Cloud environment. As the data warehousing ecosystem evolves further, these platforms will likely remain cornerstones of modern analytics, driving innovation and shaping how organizations glean insights from the expansive data at their disposal.

REFERENCES

1. S. Alotaibi and K. Reed, "Adaptive security for Cloud data warehouse as a service," 2015 IEEE International Conference on Big Data (Big Data), 2015, pp. 2526-2531.
2. Y. Wu and S. B. Yao, "Benchmarking data warehouse systems in the cloud," 2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW), 2013, pp. 194-197.
3. O. A. Elsayed and M. A. El-Sehiemy, "A Data Warehouse Approach for Business Intelligence," 2019 International Conference on Advanced Communication Technologies and Networking (CommNet), 2019, pp. 1-6.
4. V. V. Subrahmanyam and M. N. Doja, "Cloud Computing and Data Warehousing," 2014 National Conference on Recent Trends in Computing and Communication, 2014, pp. 1-5.
5. A. Nambiar and D. Mundra, "An Overview of Data Warehouse and Data Lake in Modern Enterprise Data Management," Big Data and Cognitive Computing, vol. 6, no. 4, Article 132, 2022.
6. A. Abelló, J. Darmont, L. Etcheverry, M. Golfarelli, S. Mazon, J. N. Mothe, and P. Obradovic, "Design of Cloud Data Warehouse and Its Application in Smart Grid," IEEE Transactions on Knowledge and Data Engineering, vol. 28, no. 7, pp. 1739-1753, 2016.
7. M. Golfarelli and S. Rizzi, "From Star Schemas to Big Data: 20+ Years of Data Warehouse Research," Studies in big data, pp. 93–107, May 2017.
8. Kawthar Karkouda, Ahlem Nabli, and Faiez Gargouri, "CloudWar: A new schema for securing and querying data warehouse hosted in the cloud," pp. 6–12, Oct. 2018.
9. J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh, "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Total," in Proc. of the 12th International Conference on Data Engineering, 1996, pp. 152-159.
10. Yuan, L.-Y., Wu, L., You, J.-H. and Chi, Y. 2014. Rubato DB: A Highly Scalable Staged Grid Database System for OLTP and Big Data Applications. In 23rd ACM International Conference on Information and Knowledge Management (Melbourne, Australia, 2014). CIKM 2014. ACM Press, 1-10.