Exploring BigQuery and Google Cloud as A Cloud Solution:

An Analysis of Alliah's E-Commerce Data Challenges

Joanne Senoren

Master of Science, Data Analytics

Student ID: 011826418

March 30, 2025

Table of Contents

A. Requirements for A Data Cloud Solution

Alliah generates over 10TB of data daily, and multiple teams utilize the data for strategy, marketing, inventory management, and customer service. This process involves high-variety and high-volume data. Muñoz (2022) states that a transition from traditional data systems to a cloud-based system is "a radical change in application design patterns, from highly coupled monolithic systems based on synchronous interactions and with broad transactional contexts, to component-based systems with very high modularity, very low coupling, based on asynchronous interactions and without the possibility of extending transactional contexts outside of the components themselves."

Because of this extreme change in computing, data management, communication, and security, critical points are prone to failure within the system. Establishing functional and non-functional requirements for the business's transition to cloud service is important because it allows cloud-based users (Alliah's employees, customers, and stakeholders) to get on the same page as the cloud service provider. Setting a solid understanding of service level agreements (SLAs) lets concerned parties work towards the same goal (*BigQuery IAM Roles and Permissions*, n.d.).

1. Functional Requirements

The following functional requirements are listed in the table (Table 1) below that addresses Alliah's business needs.

Table 1

Functional Requirements

Functional Requirement	Description

Authentication & Authorization	The system should be able to provide high security with policy and data privacy compliance, identify users, grant/revoke user permissions, and provide multifactor user verification.
Dynamic Configurability	The cloud system should be able to automate and provide native capabilities to configure the volume of data ingestion, performance, and speed based on dynamic and timely business needs.
Data Validation	The system should be able to ingest structured, semi-structured (e.g., JSON), and unstructured data. It should perform validation processes to ensure data quality.
UX and UI	The cloud system should be able to provide self-service and role-based capabilities that allow users from different teams to retrieve and analyze data (such as report generation and visualization) within user-friendly interfaces and environments.
Recency	The cloud system should be able to operate on a set data operation schedule as defined by an authorized user that ensures data pipelines continuously receive and push data.

Establishing functional requirements prior to selecting a cloud-based computing system or vendor can help companies avoid falling short of their business needs or spending too much by overprovisioning (Systems, 2024). Migrating business and data operations to cloud-based systems benefit a company with reduced costs, improved collaboration, failover systems, and increased scalability (Erl & Monroy, 2023).

2. Non-Functional Requirements

An equally essential consideration for a cloud migration procedure is the non-functional requirements defined to align a cloud platform's specific processes. An organization, such as Alliah, using a traditional storage system that handles non-functional requirements on a local platform environment must heavily consider the implications of data integrity, reliability, and availability. The business must explicitly define and manage non-functional requirements when moving to a cloud storage solution (Muñoz, 2022).

An example by Muñoz (2022) exemplifies this heightened focus on proactive non-functional design due to a significant change in data retrieval. Traditional data processes utilize locally integrated systems within the same environment, so the system inherently handles non-functional requirements such as data availability and reliability in place. However, cloud databases make API calls to communicate with each other and cannot guarantee the same inherent reliability processes.

The table (Table 2) below demonstrates critical non-functional requirements based on Alliah's business needs and their descriptions. Muñoz (2022) emphasizes implementing a thorough specification framework for each requirement that encompasses guidelines for analysis, guidelines for development, implications on architecture, and external references.

Table 2

Non-functional Requirements

Aspect	**Description**
Availability and Reliability	The cloud system should have a high fault tolerance and proactive monitoring to minimize latency and business operation disruptions. The cloud system should have failover procedures that include routine backup to withstand unforeseen failures and implement recovery processes to ensure data integrity and business consistency.
Security and Privacy Compliance	The cloud system addresses sensitive data, such as customer data, that should be protected within the models and compliant with privacy regulatory bodies based on the locations of the user data generated.
Scalability	The cloud system should be able to scale up or down dynamically based on the resources needed per operating use case.
Multi-Tenancy	The cloud system should be able to provide various resources, including additional cloud storage and device-agnostic accessibility. The system should be able to upgrade software across all users to ensure software recency and minimize physical resource utilization.
Monitoring	The cloud system should inherently monitor performance, storage management, and user activity to ensure the cloud system maintains efficiency.

B. Cloud Data Solutions

1. Cloud Data Solution Recommendation and Relevance to Scenario

The transition from a self-managed on-premises data storage unit to a cloud-based Database as a Service (DBaaS) where the cloud-service provider manages the hardware and software will allow Alliah to focus more on its multifaceted business objectives. Alliah can benefit from several other DBaaS features. A cloud-based data warehouse can scale up or down based on resources needed, making it elastic. It also lets users set up databases and upload data quickly. Users can also leverage the fully managed database infrastructure provided by the service provider, which covers hardware, software, upgrades, and automated patching (Oracle, 2020). Subsequently, some DBaaS can integrate with PaaS to provide services that include environments to perform analytics with queries and generate reports and dashboards.

a. Cloud Vendor Recommendation and Justification

BigQuery and Google's robust cloud platform services are an appropriate choice for this type of data migration. BigQuery is a cloud-based managed data warehouse that can quickly ingest and process the high volume and variety of data that Alliah generates daily. BigQuery is fully managed by Google across its infrastructure, from provisioning servers, networks, and storage. However, Alliah can benefit from using BigQuery's two-part architecture without overextending it to other Google Cloud Platform tools. BigQuery first layer, the storage layer, ingests and processes data, while its second layer, the analytics layer, provides analytical OLAP services and tools (*Overview of BigQuery Storage*, n.d.). Google manages the database and OLAP maintenance, which includes updates, upgrades, and software patching.

Based on Ravindranath's (2025) comparison against Azure, BigQuery is a serverless warehouse, making it highly performant, meaning that the infrastructure is distributed across servers, which allows for parallel query processing so that there is minimal latency in query times. BigQuery is also cost-effective, requiring businesses to only pay for the storage used and queries run (Ravindranath, 2025). BigQuery also provides native support for JSON-formatted data, while other providers require secondary paid software to conduct analysis, such as RedShift Spectrum for RedShift (Data, n.d.). BigQuery also beats other competitors in terms of security, focusing on compliant privacy laws as it automatically encrypts sensitive data at rest without requiring users to configure it manually (Klein, 2024). BigQuery has a robust backup recovery strategy that includes an automated backup of data over the past seven days and establishing copies of data by writing data across two availability zones (Pandey & Akunuru, 2022). Pandey and Akunuru (2022) state that Google ensures BigQuery can recover from zone-level failures with 99.99% availability. If a zone fails, BigQuery is designed to maintain availability by failing over to other working zones.

b. Major Features of Proposed Solution

BigQuery utilizes storage classes that are available on Google Cloud. Four storage classes are available: standard, nearline, coldline, and archive. Storage classes are properties of how data is stored within the system. The following table (Table 3) describes the four Google Cloud storage classes, the appropriate data type for each class, and an example of how Alliah can implement each storage class (*Storage Classes*, n.d.).

Table 3

Storage Classes and Potential Use Cases

Storage Class	Data Type	Storage Class Features	Potential Use Case
Standard	Frequently accessed or “hot” data	• No minimum storage duration • Data can be stored in different locations to maintain high availability	Standard storage would become the primary storage class for Alliah’s transaction data, where data can be ingested, processed, and queried for real-time analytics.
Nearline	Infrequently accessed data	• 30 days minimum storage duration • Ideal for data to be read or modified an average of a month or less	The Nearline class can store less frequently accessed data used for monthly reporting to determine product trends and maintain inventory.
Coldline	Less frequently accessed data than Nearline storage	• 90 days minimum storage duration • Low cost compared to Nearline and Standard • Ideal for data read or modified within quarterly frequencies	The Coldline class can store even less frequently accessed data used for observing quarterly performance across sales, inventory, and customer service performance.
Archive	Archived and/or backup data	• 365 days minimum storage duration • Google Cloud’s Archive class storage provides low latency in accessing recovery data during disaster events	The Archive storage class can store data needed for regulatory compliance and backups for disaster recovery purposes.

All storage classes in the table above feature unlimited storage with unlimited access time, no minimum object size, worldwide availability and storage locations, low latency, high durability, and redundancy for data distributed across multi-region storage (*Storage Classes*, n.d.).

Consequently, one of BigQuery's key features is its ability to dynamically allocate the size based on the data volume, such as Alliah's 10TB of daily data. Businesses do not need to provision or manage storage space within BigQuery since it inherently scales resources up or down depending on the size of the data (*Overview of BigQuery Storage*, n.d.). Other key features that make BigQuery a standout choice for Alliah's business needs include durability, security encryption, and query efficiency.

BigQuery facilitates data durability by replicating data across multiple availability zones. For security, BigQuery automatically encrypts all data at rest and in transit. It also has identity and access management (IAM) capabilities that let data owners and managers configure roles and permissions. The IAM configuration also follows the least privilege principle, a best practice that states nobody should have more permissions than they need (*Overview of BigQuery Storage*, n.d.). Query returns are fast because BigQuery stores data in a columnar format. It means that it stores each column separately and scans each column instead of an entire table. They are helpful in efficiently retrieving data for analytical work, especially when it comes to a large number of records. Users can also load JSON files into BigQuery without using separate software since BigQuery natively supports JSON formats, including an auto-detect schema capability (*Overview of BigQuery Storage*, n.d.).

Alliah can benefit from several of BigQuery's cost-effective features. Since BigQuery is serverless and fully managed by Google as the service provider, Alliah does not have to pay to maintain hardware or software licenses. BigQuery's pay-as-you-go offering allows Alliah to scale their storage for what they need and only pay for that amount of storage, avoiding waste. The layers of BigQuery's architecture can be scaled separately, so if Alliah does not use the OLAP services for some time, they will only pay for the storage services.

Overall, BigQuery's ability to provide Alliah with a fully managed and resourced data storage solution and a robust and optimized OLAP interface means that Alliah can focus entirely on business intelligence rather than spend time on database logistics.

2. Cloud Storage Solution Justification for Functional and Non-Functional Requirements

Let us revisit Alliah's business needs below and how BigQuery can address each need. Alliah's e-commerce transaction can rack up to 10TB in data size, and a cloud-based data warehouse can ingest, manage, and centralize large amounts of data for analysis with elastic capabilities, scaling up or down depending on data needs and resources. A cloud-based data warehouse can also manage a variety of data formats, such as Alliah's data transaction JSON objects. When it comes to cost-effectiveness, cloud-based data warehouses are a great fit because of their predictable pricing plans based on storage usage. Costs can be as flexible as fixed amount plans based on the number and types of resources used or based on throughput (*What Is a Data Warehouse? | Google Cloud*, n.d.). BigQuery also has the capability to ingest data from various sources and transfer analyzed data to other destinations for cross-functional purposes. Alliah also emphasized real-time data analysis. BigQuery can support streaming data, which lets users query data in real time (*What Is a Data Warehouse? | Google Cloud*, n.d.).

The table below (Table 4) justifies how BigQuery's features, and Google Cloud's services address each functional requirement mentioned in section A.

Table 4

Functional Requirement Justification

Requirement	Aspects	BigQuery Features

Authentication & Authorization	• Provide high security with policy and data privacy compliance • Identify users, grant/revoke user permissions	BigQuery has identity and access management capabilities, allowing database owners to configure user roles and permissions (*BigQuery IAM Roles and Permissions*, n.d.). It inherently uses the least privilege principle so that users only have the necessary access they need (*BigQuery IAM Roles and Permissions*, n.d.).
Dynamic Configurability	• Provide capabilities to configure the volume of data ingestion, performance, and speed based on dynamic and timely business needs.	BigQuery manages hardware (e.g., servers) and software and can scale resources up or down based on the data ingested and processed and settings established by the user (*Overview of BigQuery Storage*, n.d.).
Data Validation	• Ingest data and perform validation processes • Set validation rules to ensure data quality	For JSON files, BigQuery lets users establish validation and schema rules while offering native support for automatically detecting JSON schema (*Working With JSON Data in GoogleSQL*, n.d.).
UX and UI	• Provide self-service and role-based capabilities analytics • Provide user-friendly interfaces and environments	BigQuery leverages the Google Cloud Console, an interface where users can query data using SQL to generate results tables and perform analysis (*Overview of BigQuery Storage*, n.d.). BigQuery can also integrate with visualization tools like Google's Looker Studio and third-party tools like PowerBI and Tableau (*Overview of BigQuery Storage*, n.d.).
Recency	• Operate on a set data schedule defined by an authorized user • Ensures data pipelines continuously receive and push data	BigQuery natively supports real-time streaming data ingestion. It can leverage Google Cloud Composer to plan and schedule data pipelines from additional sources and then leverage Cloud Data Flow to execute the data processing (*Schedule Workloads*, n.d.).

Table 5 describes how BigQuery can address the non-functional requirements discussed in Section A.

Table 5

Non-Functional Requirement Justification

Requirement	Aspects	BigQuery Features
Availability and Reliability	• Have a high fault tolerance • Have a proactive system to minimize latency and business operation disruptions • Have failover procedures that encompass routine backup to withstand unforeseen failures • Implement recovery processes to ensure data integrity and business consistency	BigQuery uses Google's Colossus, an advanced distributed filing system, to replicate data across nodes throughout its regional network to prevent any single point of failure (Singh, 2024). They offer a 99.99% uptime SLA in one region (Henderson & Welcker, 2024). When the primary node goes down, the secondary region with the replicated data becomes the central data node (Henderson & Welcker, 2024). BigQuery has a recovery time objective (RTO) of less than 5 minutes in accessing the replicated data (Henderson & Welcker, 2024). BigQuery's recovery point objective (RPO) means the recovered data will be less than 15 minutes old (Henderson & Welcker, 2024).
Security and Privacy Compliance	• Addresses sensitive data, such as customer data that should be protected within the models • Compliant with privacy regulatory bodies based on the locations of the user data generated.	BigQuery encrypts data at rest and in transit using separate protocols. It also has a service called Sensitive Data Protection, which identifies and transforms personally identifiable information, or PII (*Using Sensitive Data Protection With BigQuery*, n.d.). BigQuery allows users to establish data location and industry to automatically comply with local and industry data rules

		and regulations (*Introduction to Data Governance in BigQuery*, n.d.).
Scalability	• Have elastic storage properties to manage changes in data size and volume • Be able to dynamically scale up or down based on the number of resources needed per operating case	BigQuery automatically allocates storage using Colossus and deploys automated computing efficiencies with Dremel (Singh, 2024). BigQuery uses Massively Parallel Processing (MPP), which distributes a query across multiple servers to make its processing highly performant, efficient, and fast (Singh, 2024).
Multi-Tenancy	• Be able to provide a variety of ways to access data • Provide additional cloud storage, device-agnostic accessibility	BigQuery integrates with Google Cloud Storage and third-party data warehouses. It lets users access and connect to its data through various environments, including the Google Cloud Platform interface, command line interface, and BigQuery API (*Overview of BigQuery Storage*, n.d.).
Monitoring	• Inherently monitor performance, storage management, and user activity to ensure the cloud system maintains efficiency	BigQuery inherently provides query insights, such as query plans, to observe for optimizations (*BigQuery IAM Roles and Permissions*, n.d.). Users can leverage Google Cloud Monitoring to observe performance metrics such as query latency, throughput, and errors. It can integrate with Cloud Audit logs to monitor user activity and data access (*BigQuery IAM Roles and Permissions*, n.d.).

C. Cloud Strategies and Requirements

1. Security and Legal Requirements to Consider for Cloud Migration

Security and legal compliance are critical considerations, especially when the nature of data involves personally identifiable information. Organizations risk exposing their data to malicious actors without a correctly laid out migration plan that encompasses security, access management, and regulatory compliance (Shackleford, 2024). Data breaches, operational disruptions, and compliance failure can all lead to detrimental business consequences. Alliah must establish a solid security strategy to securely and confidentially migrate their data to a cloud storage solution.

Alliah needs to, first and foremost, protect sensitive data. Organizations should take advantage of advanced encryption capabilities for customer names, addresses, and other PII that need to be encrypted at rest and in transit and ensure strict controls (Shackleford, 2024). The company must also enact IAM settings based on users with authority to access this data. Alliah should implement additional IAM policies, such as multifactor authentication. Overseeing data cloud migration, especially to new privileged users such as cloud provider employees (e.g., cloud engineers and site reliability engineers), is critical to minimize unauthorized access, misuse of data, or accidental data exposure. Establishing migration supervision also ensures that the external teams store and process data based on the company's data residency and sovereignty requirements (Shackleford, 2024).

Based on the cloud's data center location, the customers' location, and the type of data stored, all data must inherently comply with regulatory bodies such as GDPR to meet legal and compliance expectations (Shackleford, 2024). Alliah can extend BigQuery's inherently compliant data security policy by adding further cloud security controls to meet their industry requirements, especially regarding sensitive customer data (*Introduction to Data Governance in BigQuery*, n.d.-b).

Alliah should provide visibility to the cloud control plane so that the security teams can monitor and adjust configuration settings in areas needing securing. Shackleford (2024) states that cloud operations could be automated with enabled security tracking. However, establishing the ability to configure and secure cloud accounts for users with monitoring enabled should be the highest priority.

2. Cloud Backup Strategies

Alliah must set up a comprehensive backup procedure and a data recovery protocol during disaster events. The organization must define the RTO (recovery time objective) and RPO (recovery point objective) before setting up a backup strategy that minimizes the disruption and impact on revenue and customer service. The RTO is the maximum amount of acceptable time a business's systems, such as websites and databases, can go down and work again without losing significant revenue or causing user frustration. The RPO is the maximum amount of lost data (measured in time) that a business can sacrifice in a data loss event. The RPO will affect backup frequency because, ideally, the backup frequency ensures that the RPO is consistent with the expected data recovery (PingCAP, 2024).

For an e-commerce business like Alliah, a low RTO and low RPO are very important since downtime can harm revenue loss and customer dissatisfaction (PingCAP, 2024). Downtime can also impact Alliah's financial systems, including transactions and operational systems such as inventory management. The TiBD team (2024) recommends a very low RTO of 30 seconds and a stringent RPO of 15 minutes for e-commerce businesses to ensure online business continuity, maintain brand reputation, and safeguard customer trust. Consequently, an RPO of 15 minutes means that data ideally must be backed up and replicated every 15 minutes. Very stringent recovery objectives for an extensive volume of data like Alliah's daily 10TB would require much higher payment plans due to increased storage and computing power resources.

Identifying and defining potential failures is key in choosing appropriate backup methods. BigQuery identifies different types of failures, which include soft (non-hardware with no actual data loss), hard (destroyed hardware with data loss), and bugs caused by the software or users (Pandey & Akunuru, 2022b). Lower-level and more serious failures include zone and region failures, severely affecting data availability and downtime.

BigQuery has several built-in and cost-effective backup options available to its customers. Time travel is a service that backs up all modified data from the past 7 days and can quickly recover data from software failures and bugs. Cloud users can extend time travel beyond 7 days by utilizing table snapshots to extend the recovery window. For zonal failures, BigQuery can copy datasets from one zone to another and set the replication on a recurring schedule. It automatically replicates data across two separate zones so that it can failover queries to the copied-over zone without delay to the user (Pandey & Akunuru, 2022). For regional failures, Alliah can set up recurring backups in a separate space, such as Google Cloud Storage, that seamlessly integrates with BigQuery (Pandey & Akunuru, 2022). Leveraging storage classes (standard, nearline, coldline, archive) within Google Cloud Storage can potentially streamline costs (*Storage Classes*, n.d.).

Depending on the size and frequency of data backups, there are additional considerations towards the overall architecture. Scheduling and automating backups are important to streamline the backup and recovery process. Looking into costs for the size of backup storage needed is also important. The system must also support large volumes of incremental data transfer for replicated data across regional data centers.

The proposed frequency of backing up incremental data up to 10TB daily every 15 minutes will take a lot of processing, resources, and cost. BigQuery has a managed disaster recovery offering automated failover of compute and storage workloads across configured regions (Henderson & Welcker, 2024). A replicated dataset will be stored in a secondary region on standby should disaster events occur in the primary region's data center. The RPO for this service is less than 15 minutes old, and the RTO is less than 5 minutes. These recovery objectives are ideal for an e-commerce business with the 10TB of daily data Alliah generates. However, this feature comes at a premium price point. Alliah would have to pay much more to leverage this service. An alternate solution is to increase the proposed RTO and RPO to benefit from the more cost-effective options within BigQuery.

3. Cloud Solution Failover and Disaster Recovery Planning

BigQuery automatically replicates data across two zones to maintain data availability for failovers should the primary zone become unavailable. It will failover the queries to the secondary location with the replicated data. This feature is built into BigQuery so users do not notice a delay. This is crucial for Alliah's recommended RTO of 30 seconds. This feature is especially critical for hard failures, as defined by BigQuery.

BigQuery accommodates the suggested disaster recovery plans in Section 2 with several options available as inherently features or as a pricier investment. Customers can leverage Google Cloud Storage with BigQuery by setting up copies of datasets on a recurring schedule. Configurations based on data storage classes can be adjusted to manage costs.

In case of a more catastrophic regional failure, the dataset in Google Cloud Storage can be retrieved within BigQuery as a backup. Replication across two regions is also possible with BigQuery, which supports both failover and backup efforts. However, as mentioned in Section 2, this managed disaster recovery feature would come with a higher price tag.

Additionally, monitoring all backup and recovery environments is essential to maintain data integrity and availability.

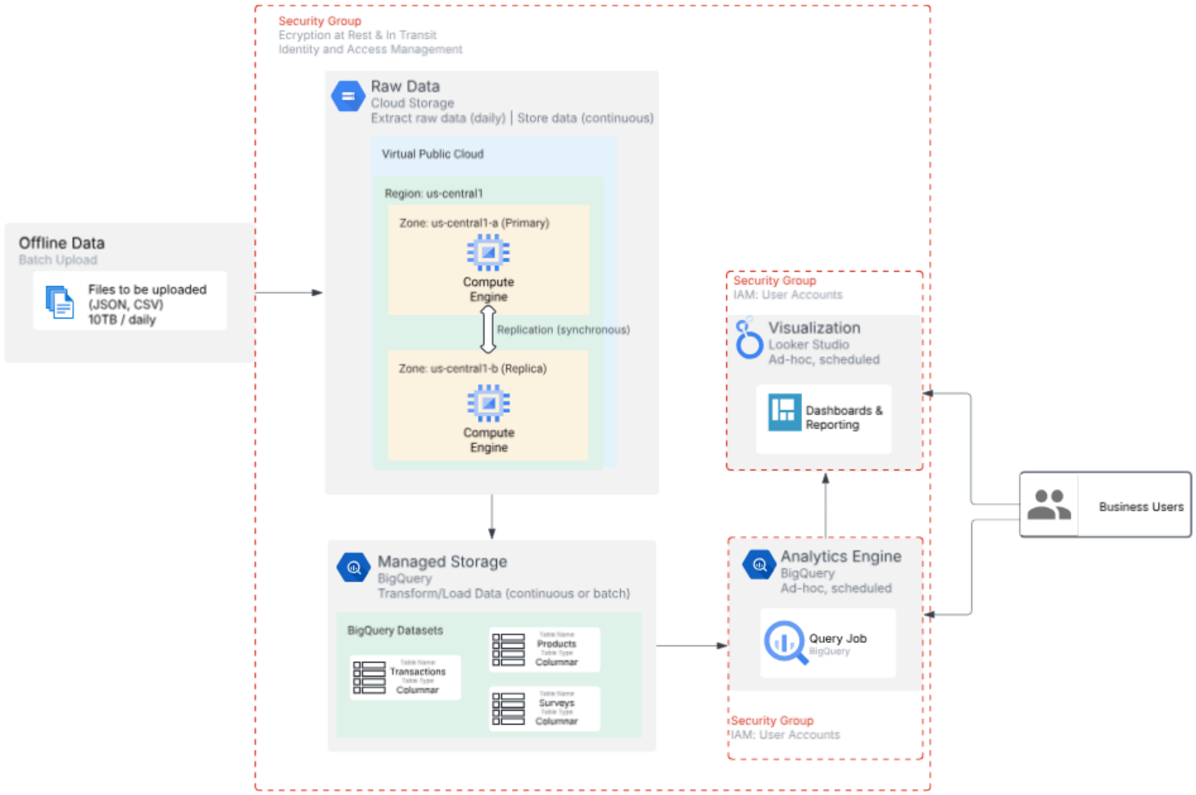
D. Architecture Diagram

1. Cloud Architecture Diagram

The cloud architecture diagram in Figure 1 depicts the flow of data ingestion from raw data to extraction within Google Cloud Storage, as leveraged by BigQuery. Then, within BigQuery's managed storage system, the data is transformed into columnar tables and loaded. The tables in the diagram are example tables that Alliah may create as they move to scale their data. The analytics portion of BigQuery is separated from the managed storage services because it manages another security group involving user accounts. Because Alliah seeks to generate reports and create dashboards for their data, the diagram also consists of Looker Studio, the visualization tool they can leverage within the Google Cloud Platform.

Figure 1

Google Cloud and BigQuery Architecture Diagram



The diagram includes labels for volume and type of data ingested, labels for example, databases (type and schema), labels for processes and frequency of operations, security groups, and labels for availability zones.

2. Database Object Identification and Logical Diagram

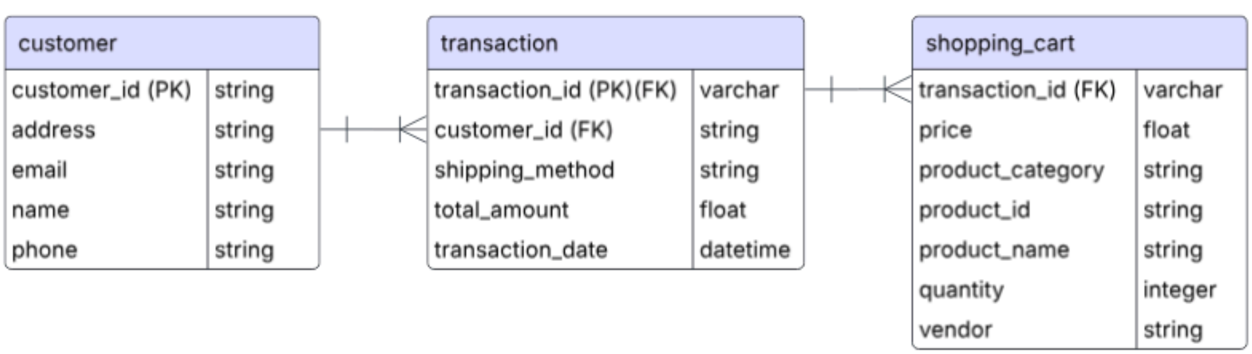
There are three tables identified based on the JSON objects: transactions, customers, and shopping carts, which also represent critical points of Alliah's e-commerce entities. Transactions should be its table as a record of every transaction made on Alliah's e-commerce site. Sensitive information, such as customer data, should be kept in a separate table to minimize data redundancy and integrity (Coronel & Morris, 2022). Additional security configurations with customer data should also be implemented within the table. The shopping cart is a separate entity because Alliah must track all purchased items. The transaction table is related to the customer table using the customer ID, and the transaction is related to the shopping_cart table using the transaction ID. This logical diagram follows a star schema with one fact table and two dimension tables (Dearmer, 2024).

a. Logical Data Diagram

The logical data diagram in Figure 2 reflects the database objects from the provided JSON file. BigQuery stores its data in columnar format, which is not the same as the diagram's structured format. This logical design focuses on the entities' structure, identification, and relationships and their attributes regardless of the type of database implemented. The customer table establishes a relation to the transaction table with the customer id. One customer can have many transactions, but each transaction belongs to only one. The shopping_cart table relates to the transaction table using the transaction ID. The transaction can consist of multiple products in a shopping cart. However, each product and its composition of other attributes (e.g., price, quantity) must relate to a single transaction.

Figure 2

Logical Data Diagram



3. Database Objects Justification

Structuring the data across the three tables is a logical and normalized approach to reflect data contained inside the objects of the JSON array. Normalization is an important step to remove redundancy and create efficiency in the database (Coronel & Morris, 2022). Separate tables for transactions, customers, and shopping cart items are important because they represent separate units of business data. The conceptual schema contributes to improved data governance when the relevant teams understand the data structure within their dataset.

Each transaction must be recorded so that Alliah can analyze revenue. Alliah has a high volume of transaction data generated daily, so the table must remain streamlined so that users can analyze transaction data accurately. The customer table removes redundancy in the database by keeping unique instances of customer information only once in the system. Additionally, a separate customer table will allow BigQuery to support region-specific privacy laws by implementing additional security on the customer table. It is related to the transaction table using the customer ID. The shopping_cart table also minimizes redundancy from the transaction table by storing all product information. It has a relationship to the transaction table using the transaction ID.

Having analyzed Alliah's business challenges regarding their need to manage 10TB of daily data and various data types and provide analytical tools, the exploration of BigQuery and Google Cloud features has proven to be a scalable and efficient solution. The justification of BigQuery highlights the advantages that Alliah can gain by using it as their cloud storage solution. By employing a fully managed cloud database solution, Alliah can focus on impactful data-driven decisions, improve customer service, and support its business growth. The following steps for Alliah include establishing a detailed plan of execution based on the BigQuery solution and developing a robust SLA with the cloud provider surrounding data types, size, and security requirements.

E1. Resources (Tools and Architecture Tutorials)

1. *BigQuery — The Cloud Girl*. (n.d.). The Cloud Girl. https://www.thecloudgirl.dev/data-analytics/bigquery
2. *Cloud architectures diagrams [2024 Complete Guide] | Exoway*. (n.d.). https://www.exoway.io/blog/cloud-architecture-diagram
3. Cloud Career Mentor. (2023, November 29). *How to easily draw cloud architecture Diagrams* [Video]. YouTube. https://www.youtube.com/watch?v=zK8DUZM18UA
4. *Lucid visual collaboration suite: Log in*. (n.d.). https://lucid.app/documents/#/home?folder_id=recent
5. Thallam, R. (2020, September 2). An overview of BigQuery's architecture and how to quickly get started. *Google Cloud Blog*. https://cloud.google.com/blog/products/data-analytics/new-blog-series-bigquery-explained-overview
6. Thallam, R. (2021, December 15). BigQuery explained: Data ingestion - Google Cloud - community - medium. *Medium*. https://medium.com/google-cloud/bigquery-explained-data-ingestion-cdc26a588d0
7. Truong Thuc Tam. (2020, November 6). *Walkthrough: Data Architecture Diagram - Exploring and Preparing your Data with BigQuery* [Video]. YouTube. https://www.youtube.com/watch?v=v9GOysqh4dU

E2. References (Citations)

1. Altexsoft Editorial Team. (2023, November 30). Functional and nonfunctional requirements: Specification and types. *AltexSoft*. https://www.altexsoft.com/blog/functional-and-non-functional-requirements-specification-and-types/
2. *BigQuery IAM roles and permissions*. (n.d.). Google Cloud. https://cloud.google.com/bigquery/docs/access-control
3. Coronel, C., & Morris, S. (2022). *Database systems*.
4. Data, S. (n.d.). *Redshift and BigQuery have many similarities, but also important differences that can tip the scales in a cloud data warehouse comparison.* Stitch Data. https://www.stitchdata.com/resources/redshift-vs-bigquery/
5. Dearmer, A. (2024, February 15). *Complete Guide to Database Schema design*. Integrate.io. https://www.integrate.io/blog/complete-guide-to-database-schema-design-guide/
6. Erl, T., & Monroy, E. B. (2023). *Cloud computing: Concepts, Technology, Security, and Architecture*. Pearson.
7. Henderson, L., & Welcker, B. (2024, May 6). BigQuery gets managed disaster recovery. *Google Cloud Blog*. https://cloud.google.com/blog/products/data-analytics/bigquery-gets-managed-disaster-recovery
8. *Introduction to data governance in BigQuery*. (n.d.). Google Cloud. https://cloud.google.com/bigquery/docs/data-governance
9. Klein, C. (2024, December 18). *AWS vs. Azure vs. Google Cloud: A Security Feature Comparison*. Jit. https://www.jit.io/resources/cloud-sec-tools/aws-vs-azure-vs-google-cloud-a-security-feature-comparison

10. Muñoz, A. (2022, January 7). Non functional requirements specification Framework for Corebanking Transformation to cloud. *Medium*. https://alfredomunoz.medium.com/non-functional-requirements-specification-framework-for-corebanking-transformation-to-cloud-51f87effc998

11. Oracle. (2020, October 29). *What is DBaaS?* https://www.oracle.com/database/what-is-a-cloud-database/dbaas/

12. *Overview of BigQuery storage*. (n.d.). Google Cloud. https://cloud.google.com/bigquery/docs/storage_overview#:~:text=BigQuery%20automatically%20allocates%20storage%20for,of%20storage%20that%20you%20use.

13. Pandey, S., & Akunuru, P. K. (2022, October 19). Backup & Disaster Recovery strategies for BigQuery. *Google Cloud Blog*. https://cloud.google.com/blog/topics/developers-practitioners/backup-disaster-recovery-strategies-bigquery

14. PingCAP. (2024, December 12). *RPO and RTO: Understanding the Concepts with Examples*. TiDB. https://www.pingcap.com/article/rpo-and-rto-understanding-the-concepts-with-examples/

15. Ravindranath, A. C., PhD. (2025, March 7). Why choose Google BigQuery over Azure Databases? - Aakash Chavan Ravindranath, Ph.D - medium. *Medium*. https://medium.com/@craakash/why-choose-google-bigquery-over-azure-databases-181b9dc6b55e

16. *Schedule workloads*. (n.d.). Google Cloud. https://cloud.google.com/bigquery/docs/orchestrate-workloads

17. Shackleford, D. (2024, December 26). *Cloud migration security challenges and best practices*. Search Cloud Computing.

https://www.techtarget.com/searchcloudcomputing/tip/9-cloud-migration-security-considerations-and-challenges#:~:text=Data%20in%20transit%20and%20at,cloud%20data%20storage%20and%20transit.

18. Singh, V. (2024, March 7). BigQuery’s architecture and working mechanism - VIKRANT SINGH - medium. *Medium*. https://medium.com/@vkrntkmrsngh/bigquerys-architecture-and-working-mechanism-dad5038ebc28#:~:text=BigQuery's%20ability%20to%20rapidly%20process,Massively%20Parallel%20Processing%20(MPP).&text=This%20parallel%20processing%20capability%20ensures,seconds%20to%20minutes%2C%20showcasing%20BigQuery's
19. *Storage classes*. (n.d.). Google Cloud. https://cloud.google.com/storage/docs/storage-classes
20. Systems, E. (2024, July 12). Requirements for cloud computing for SMB and SME | EIRE. *EIRE Systems*. https://www.eiresystems.com/8-smb-sme-cloud-computing-requirements/
21. *Using Sensitive Data Protection with BigQuery*. (n.d.). Google Cloud. https://cloud.google.com/sensitive-data-protection/docs/dlp-bigquery
22. *What is a Data Warehouse? | Google Cloud*. (n.d.). Google Cloud. https://cloud.google.com/learn/what-is-a-data-warehouse?hl=en#advantages-of-data-warehousing-in-the-cloud
23. *Working with JSON data in GoogleSQL*. (n.d.). Google Cloud. https://cloud.google.com/bigquery/docs/json-data