**1. Introduction and Purpose**

This document presents the **Automation System Integration Testing (SIT) Test Plan** for validating the new ingestion methods being adopted as part of the **Guidewire Data Migration Initiative**. The goal is to ensure the accurate and consistent migration of data from **PolicyCenter**, **BillingCenter**, and **ClaimCenter** into the **Azure Data Lake**—specifically into the **Bronze layer**, followed by further processing into Silver and Gold layers.

**Current Ingestion Approach**

Historically, data from Guidewire systems has been ingested into Azure using **Azure Data Factory (ADF)** pipelines, pulling directly from **on-prem Oracle databases**. While functional, this approach is tightly coupled with legacy infrastructure and lacks scalability for cloud-native workflows.

**Need for New Ingestion Methods**

To modernize the data platform, the ingestion architecture is transitioning to a **cloud-first model** with two primary methods:

* **a. Read Replica Ingestion** (Interim): Data is ingested from a **PostgreSQL read replica** of the Guidewire database.
* **b. CDA File Ingestion** (Target): Batch files are generated via **Cloud Data Access (CDA)** and loaded from **Amazon S3**.

The migration strategy begins with read replica ingestion as an interim solution and will later transition to the more scalable CDA file-based ingestion.

**Challenges in Initial CDA-Based Load**

As documented, delays in the availability of CDA files within the **S3 bucket at Guidewire** posed a risk to project timelines. Additionally, **direct access restrictions** from Databricks to the S3 data further complicated validation efforts. To mitigate this, the team introduced an **interim ingestion using the PostgreSQL read replica**, which mirrors the source Guidewire databases and enables validation continuity.

**Why Validate the Read Replica Approach**

The read replica ensures minimal change to ingestion logic and provides near-real-time access to Guidewire data. It serves as a like-for-like replacement for Oracle in the short term and ensures that the ingestion process remains unblocked while the CDA pipeline matures.

Validating this read replica method ensures that data integrity is maintained during the initial cutover and that the migration path to CDA remains smooth and consistent.

**Why an Automation Framework is Essential**

Given the **large volume of ingestion tables** across all three Guidewire centers, manual testing is neither scalable nor sustainable. The validation must cover:

* **Three source systems** (PolicyCenter, BillingCenter, ClaimCenter)
* **Two ingestion mechanisms** (Read Replica and CDA)
* **Three data lake layers** (Bronze, Silver, Gold)

An **automation framework built on Databricks (PySpark)** is therefore necessary to:

* Handle high data volumes and table-level reconciliation
* Automate schema validation, data accuracy, and completeness checks
* Detect anomalies, duplicates, and SCD2 discrepancies
* Benchmark ingestion performance
* Store test results in **Delta Lake tables**
* Surface validation metrics in **Databricks Dashboards**
* Support CI/CD through **Azure DevOps** pipelines
* Generate test outputs for **manual Jira integration**, overcoming firewall constraints

**Purpose of This Test Plan**

This test plan defines the **strategy, validation criteria, execution plan, and automation scope** required to ensure the successful validation of both ingestion methods. It provides a standardized, repeatable, and traceable testing mechanism that supports both **interim (read replica)** and **final (CDA)** ingestion approaches, ensuring that the new cloud ingestion system is **functionally equivalent** to the legacy on-prem Oracle setup.

**1. Introduction and Purpose**

This test plan defines the approach and objectives for **System Integration Testing (SIT)** using an automated framework designed to validate new ingestion methods for migrating **Guidewire application data** — specifically from **PolicyCenter**, **BillingCenter**, and **ClaimCenter** — into the **Azure Data Lake Bronze Layer**.

**Current Ingestion Method**

The existing ingestion process extracts data from **on-premise Oracle databases** and moves it into Azure via **Azure Data Factory (ADF)** pipelines. While this method has served well historically, it introduces limitations when transitioning to a scalable, cloud-native architecture, especially with Guidewire's move to cloud hosting and evolving data delivery mechanisms.

**Why the New Ingestion Method**

To modernize the data integration approach and align with the **cloud-native infrastructure**, Guidewire is introducing two new ingestion methods:

1. **Read Replica Ingestion** – Extract data from a cloud-hosted **PostgreSQL read replica**, a near real-time copy of the Guidewire database.
2. **CDA File Ingestion** – Consume structured **Cloud Data Access (CDA) files** delivered as batch data into **AWS S3 buckets**, intended for ingestion via **Azure Databricks**.

The long-term goal is to phase out the read replica and **transition fully to CDA-based ingestion**.

**Challenge in Initial Load: The CDA Delay**

The original plan involved loading initial bulk data from CDA files into the Bronze layer. However, **delays in making initial CDA data available in S3** required an interim solution to keep the migration timeline on track.

**Interim Solution: Read Replica Approach**

In response to CDA delays, the team adopted an **interim ingestion path using the PostgreSQL read replica**. This approach involves:

* Setting up new linked services to the read replica in Azure.
* Migrating configurations to read from the replica instead of the on-prem Oracle DB.
* Updating schemas and source definitions where needed.
* Modifying ADF pipelines to use the new replica source.
* Executing ingestion as normal, using the **same extract queries**.

This solution ensures **minimal disruption** and continuity of ingestion validation while CDA delivery stabilizes.

**Future Path: CDA and Delta-Only Ingestion**

Once CDA ingestion catches up, the system will transition to the **CDA-first ingestion path**. Existing Databricks tables will be suffixed (e.g., \_old) and replaced by **views pointing to CDA-driven tables**, maintaining compatibility with downstream consumers.

Guidewire is also exploring a **delta-only ingestion strategy**, where instead of performing a full initial load via CDA, only incremental changes will be sent. However, this is not yet available and requires combining old and new data for a complete view.

**Why Automation Is Critical**

The ingestion spans **hundreds of tables** across three major Guidewire domains. Validating each table manually is not scalable or efficient. Key reasons for implementing an **automated testing framework** include:

* High **volume of tables and data** requiring validation
* **Frequent delta loads** that must be reconciled reliably
* Need to validate across **multiple ingestion sources** (Oracle, PostgreSQL, CDA)
* Performing **three-way reconciliation** between source systems, staging layers, and Azure Data Lake
* Enabling **regression testing** for future ingestion releases
* Supporting both **test and production** environments with audit trails

**What the Automation Framework Will Validate**

The automated system integration tests will cover:

* **Data Accuracy** – Field-to-field comparison between source and target
* **Data Completeness** – Record count validation across all layers
* **Data Integrity** – Primary and foreign key validations, duplicate checks
* **Schema Validation** – Column types, structures, and constraints
* **SCD Validation** – Detect slowly changing dimensions where applicable
* **Anomaly Detection** – Null spikes, unexpected deltas
* **Performance Metrics** – Load time and resource usage per table
* **Auditability** – SQL used, test logic, and results will be stored and accessible
* **Reporting** – Results displayed via dashboards, and optionally exported for JIRA tracking

This **automation framework is pivotal** in ensuring that the new ingestion methods deliver the same level of data reliability, traceability, and structure as the legacy system. It supports a seamless and efficient migration to the cloud while enabling confidence in the accuracy of data used for downstream analytics and reporting.

**1. Introduction and Purpose**

This document defines the **Automation System Integration Testing (SIT) Test Plan** for validating new data ingestion methods implemented as part of the **Guidewire Data Migration Project**. The purpose of this migration is to modernize the data ingestion architecture by shifting from legacy on-premise systems to a scalable and cloud-native solution using **Azure Data Lake**, **Azure Data Factory (ADF)**, and **Databricks**.

**Current Ingestion Method**

Historically, data from Guidewire systems — **PolicyCenter**, **BillingCenter**, and **ClaimCenter** — has been ingested from on-premises **Oracle databases** using **Azure Data Factory (ADF)** into the Azure Data Lake’s **Bronze Layer**. While this approach was functional, it presented challenges in scalability, maintainability, and performance in a cloud-native ecosystem.

**Why a New Ingestion Method?**

The organization is now transitioning to a **cloud-first architecture** with a two-phase ingestion approach designed for improved performance, flexibility, and future-readiness:

* **Phase A (Interim)**: Ingest data from a **PostgreSQL read replica** of the Guidewire database.
* **Phase B (Target)**: Shift ingestion to **Cloud Data Access (CDA) files** sourced from **Amazon S3**.

This approach is designed to ensure minimal disruption while achieving long-term goals of performance optimization and alignment with cloud-native patterns.

**Challenge in Initial Data Load Approach**

Initially, the plan was to begin with the **CDA file ingestion** from S3. However, due to **delays in the availability of the initial CDA data** in the S3 bucket at Guidewire, an **interim solution was required** to prevent blocking the migration timeline.

**Interim Approach: Read Replica Ingestion**

To bridge this gap, the interim plan introduces the ingestion of data from a **read replica of the Guidewire database**, using the same extraction logic as the on-premise Oracle DB. This method allows the ingestion pipelines to proceed while maintaining alignment with the future CDA structure.

Steps involved:

1. Establish and test new linked services to the read replica.
2. Once the Oracle source is frozen, run ingestion pipelines from the replica.
3. Reconfigure PolicyCenter, BillingCenter, and ClaimCenter data sources to use the replica.
4. Update ADF datasets accordingly.
5. Execute ingestion and validation as normal.

This ensures data continuity while waiting for the CDA pipeline to stabilize.

**CDA Load Strategy**

When the CDA ingestion becomes production-ready:

* CDA-backed **Databricks virtual tables** will replace existing ones.
* Older tables will be renamed with an \_old suffix.
* Views with original table names will point to the CDA tables, enabling seamless switchover.

**Delta-Only Approach (Future Consideration)**

Guidewire is exploring a **delta-only ingestion** model where no initial full load is provided, and only change data is sent through. In this scenario, **read replica ingestion will be skipped**, and both new (CDA) and old data will need to be merged to ensure completeness. However, this approach is **not yet available or validated**.

**Why Validate Read Replica?**

The **read replica ingestion** is critical as it serves as the **only viable method** for initial data migration while ensuring that:

* The same queries and schema logic are retained.
* Data consistency with legacy Oracle sources is maintained.
* There's no delay in migration progress while CDA is still catching up.

This interim strategy needs robust validation to avoid data gaps, schema drifts, or inconsistencies before production use.

**Why Automation is Required**

The ingestion spans **hundreds of tables** across **three core applications** (PolicyCenter, BillingCenter, ClaimCenter). Manual validation is impractical due to:

* Large **volume of data and objects**.
* Need for **consistent, repeatable** regression validation.
* Timely analysis for delta-only scenarios and data anomaly detection.

Automation ensures:

* Field-level and record-level comparison across ingestion sources.
* Schema validation (structure, types, constraints).
* SCD handling, duplicate checks, referential integrity, and count checks.
* Performance benchmarking and dashboard-based reporting.

**High-Level Automation Framework Strategy**

* **Engine**: Databricks (PySpark Notebooks)
* **CI/CD**: Azure DevOps (Integrated with Git for version control)
* **Data Sources**: Read Replica (PostgreSQL), CDA (S3), On-Prem Oracle (via staging)
* **Validation Scope**:
  + Ingestion (Staging → Bronze)
  + Transformation (Bronze → Silver/Gold)
* **Data Output**: Validation results stored in Delta Tables, exposed via Databricks views
* **Reporting**: Dashboards for performance, quality, and reconciliation metrics
* **Test Coverage**: Based on delta loads only, assuming history is preloaded
* **JIRA Integration**: Metrics exported manually due to firewall restrictions

This framework is built to handle evolving ingestion methods and ensures that data flowing into the cloud aligns precisely with what was produced by the legacy system—supporting a **seamless, low-risk migration**.

## **2. Test Objectives and Scope**

### ****2.1 Test Objectives****

The primary objective of this test plan is to define and execute a robust **automation-driven validation strategy** that ensures the accuracy, completeness, and integrity of data ingested from both interim and target ingestion sources into the Azure Data Lake environment. The automation framework supports validation across the **Bronze**, **Silver**, and **Gold** layers of the Delta Lake architecture, with a key focus on ingestion validation from **Read Replica** and **CDA files**.

The detailed test objectives are as follows:

#### ✅ ****Data Ingestion Validation****

* Validate that data ingested from the **PostgreSQL read replica** and **CDA S3 files** matches data from the legacy **on-prem Oracle source**.
* Perform **three-way reconciliation**: On-Prem Oracle vs. Read Replica/CDA vs. Azure Data Lake (Bronze Layer).

#### ✅ ****Schema Validation****

* Ensure **schema consistency** between source systems and target layers.
* Validate **table structures**, **column data types**, **nullable constraints**, and **primary/foreign key relationships**.

#### ✅ ****Delta Load Validation****

* Focus on **delta-only** data loads as historical data is assumed to be pre-loaded.
* Validate **record-level changes**, **insert/update logic**, and **SCD2 handling** where applicable.

#### ✅ ****Data Quality Checks****

* Check for **duplicates**, **null anomalies**, **referential integrity**, and **transformation accuracy** across ingestion layers.

#### ✅ ****Performance Benchmarking****

* Measure ingestion performance in terms of **data volume processed**, **time to load**, and **resource utilization** in Databricks.

#### ✅ ****Test Evidence and Reporting****

* Automatically generate:
  + SQL used in validation
  + Test conditions and assertions
  + Test results stored in Delta Live Tables
* Provide dashboards for visibility and traceability of data quality

#### ✅ ****Integration with JIRA (Manual Sync)****

* Export test metrics to standardized formats for **manual update into JIRA** (due to firewall limitations between on-prem and cloud systems).

### ****2.2 Test Scope****

#### ✅ ****In-Scope****

The following components, layers, and activities are included in the scope of this automation test plan:

##### **Sources:**

* **On-Prem Oracle (legacy)** — Reference for original ingestion
* **Read Replica (PostgreSQL)** — Interim ingestion source
* **CDA Files (Amazon S3)** — Final ingestion source

##### **Targets:**

* **Azure Data Lake (Bronze Layer)** — Focus of ingestion validation
* **ODS and ODS\_VIEW tables** — Used for cross-validation
* **Silver and Gold Layers** — Covered under **risk-based testing**

##### **Validation Types:**

* Schema validation
* Record count comparison
* Field-level data matching
* SCD2 logic verification
* Null/duplicate detection
* Referential integrity checks
* Performance benchmarks

##### **Automation System Coverage:**

* CI/CD via Azure DevOps for test execution
* Databricks Notebooks for validation logic
* Delta Lake Tables for storing results
* Views and Dashboards for test visualization

#### ❌ ****Out-of-Scope****

The following components or testing activities are not included in the current scope:

* Historical data validation (assumed to be pre-loaded and validated separately)
* Functional/UI testing of the Guidewire applications
* Validation of downstream applications or outbound feeds beyond the Gold layer
* Performance or stress testing of Azure infrastructure
* Test data creation for upstream Guidewire systems (handled manually by QA/UI automation teams)
* Silver/Gold logic testing beyond duplicate, count, and referential integrity

#### ****Risk-Based Scope for Silver and Gold Layers****

* Due to **license limitations** for tools like ER Studio and **code stability** across these layers, the test coverage for Silver and Gold is **risk-based**.
* Focus is given to:
  + Duplicate detection
  + Referential integrity
  + Count consistency on key entities
* Testing is **timeline-driven** and **binary column-focused**, mimicking current manual QA checks.

#### ****Environment-Based Scope****

|  |  |
| --- | --- |
| **Environment** | **Scope** |
| **Test** | Reconciliation of sample test data across legacy and cloud systems. Ingestion validation using 1-day delta loads from staging layers. |
| **Production** | Post-deployment validation of Read Replica and CDA ingestion. Comparison of data in staging vs. ODS. Emphasis on accuracy and delta-only validation. |

### -------------------------------------------------------------------------------------------------------------------- ****2. Test Objectives and Scope****

#### ****2.1 Test Objectives****

The primary objective of this test automation framework is to **validate the integrity, accuracy, and completeness** of data ingested through the newly introduced cloud-native pipelines, replacing the legacy Oracle-based ingestion.

The framework aims to ensure that data ingested from both the **read replica (interim source)** and **CDA files (target source)** into the Azure Data Lake’s **Bronze layer**, and subsequently transformed through **Silver and Gold layers**, is **technically equivalent** to data from the legacy on-premise systems.

##### ✅ The specific objectives of this automation system integration test plan are:

1. **Validate End-to-End Ingestion Pipeline**
   * Ensure seamless ingestion of delta loads from:
     + PostgreSQL Read Replica
     + CDA files from S3
   * Verify compatibility with existing Azure Data Factory (ADF) pipelines and configurations.
2. **Ensure Schema Consistency**
   * Validate schema structure (table definitions, column data types, nullability, constraints) between:
     + Source systems (Oracle / PostgreSQL / CDA)
     + Bronze layer in Azure Data Lake
3. **Verify Data Quality Across Layers**
   * Confirm that the ingested data into Bronze, Silver, and Gold layers is accurate and complete.
   * Validate that no transformation logic is unintentionally altered in the new ingestion process.
4. **Perform Three-Way Reconciliation**
   * Compare and reconcile data across:
     + On-prem Oracle
     + Read Replica or CDA
     + Bronze Layer
   * Ensure consistency across staging and final data layers.
5. **Automate Testing for High Coverage**
   * Automate validation for:
     + 1000+ tables across three systems (PolicyCenter, BillingCenter, ClaimCenter)
     + Multiple validation types: SCD2, duplicates, referential checks, null checks
6. **Enable Scalable and Repeatable Validation**
   * Support CI/CD pipeline-based automated regression testing.
   * Ensure quick validation cycles during migration windows and post-release sanity.
7. **Capture and Report Metrics**
   * Store validation results in Delta tables.
   * Generate dashboards for QA stakeholders.
   * Export test outcomes to JIRA-compatible format for audit and tracking.

#### ****2.2 Test Scope****

##### ✅ **In-Scope Activities**

| **Area** | **Scope** |
| --- | --- |
| **Ingestion Validation** | Validate delta load from PostgreSQL Read Replica and CDA (S3) to Bronze Layer |
| **Schema Validation** | Programmatic comparison of schema metadata across source and lake |
| **Data Validation** | Row-level field comparison, key integrity, null checks |
| **Duplicate Detection** | Identify and flag duplicate records in ingestion |
| **SCD2 Validation** | Detect and validate Slowly Changing Dimension Type 2 handling |
| **Referential Integrity** | Validate foreign key relationships and joins |
| **Count Validation** | Match row counts across ingestion sources and Bronze Layer |
| **Performance Benchmarking** | Track load time per table and ingestion duration metrics |
| **Risk-Based Silver/Gold Testing** | Limited validation based on binary columns, row counts, referential checks |
| **Result Storage & Reporting** | Persist test results in Delta Live Tables and render via dashboards |

##### ❌ **Out-of-Scope Activities**

| **Exclusion** | **Reason** |
| --- | --- |
| **Functional UI Testing** | Out of scope for technical data validation |
| **Historical Load Testing** | Assumed to be preloaded in ODS layers |
| **Transformation Logic Changes** | Not expected as part of the migration (technical migration only) |
| **Outbound System Testing** | Silver → Gold → Outbound layers are validated via existing test processes |
| **Real-time Data Feeds** | Project scope limited to batch ingestion pipelines |
| **JIRA Direct Integration** | Due to firewall constraints, direct sync is not possible |

##### ⚠️ **Risk-Based Testing Scope (Silver and Gold Layers)**

* Only critical tables based on usage and business impact will be selected.
* Focused checks include:
  + Duplicate detection
  + Referential integrity
  + Binary column comparison
  + Count validation across historical partitions
* Testing prioritized based on transformation timelines and business SLA

##### 🔍 **Test Environment Scope**

* **Staging Layers**: Created for Oracle, PostgreSQL, and CDA to feed consistent data to automation.
* **Delta Load Only**: No full load testing; focus is on daily deltas.
* **ODS & ODS\_VIEW**: Bronze validation compared to these layers.
* **Manual Data Setup**: Test team supported by UI automation team for scenario preparation.

## **2. Test Objectives and Scope**

### ****2.1 Test Objectives****

The primary objective of the Automation System Integration Testing (SIT) is to validate the **accuracy, completeness, and integrity** of data ingested through the new cloud-based architecture — both during and after the migration from the legacy on-premise Oracle-based solution.

This will be achieved by leveraging an **automated, scalable, and CI/CD-integrated test framework** that ensures consistent data validation across the ingestion layers (Bronze, Silver, and Gold) for **Guidewire systems**: PolicyCenter, BillingCenter, and ClaimCenter.

The detailed objectives include:

#### 🔹 ****1. End-to-End Validation of New Ingestion Methods****

* Ensure accurate ingestion of data from both **PostgreSQL read replicas** (interim) and **CDA files from S3** (final).
* Verify that data processed from these new sources reflects the same structure, content, and lineage as that from the legacy system.

#### 🔹 ****2. Schema Validation****

* Confirm that all ingested tables match expected schemas (table names, column names, datatypes, nullability, constraints).
* Detect and report schema drifts, incompatible changes, or missing attributes.

#### 🔹 ****3. Data Accuracy and Consistency****

* Perform **field-level data comparison** between:
  + On-prem Oracle staging vs. PostgreSQL staging vs. CDA
  + Staging layer vs. Bronze layer
* Validate SCD (Slowly Changing Dimensions) where applicable.
* Confirm transformation rules (if any) are applied identically.

#### 🔹 ****4. Data Completeness and Reconciliation****

* Ensure **record counts** match across ingestion points.
* Perform **three-way reconciliation**: Oracle → PostgreSQL → CDA → Bronze.

#### 🔹 ****5. Performance Validation****

* Track ingestion times per table/source.
* Monitor resource consumption in Databricks for benchmarking and optimization.

#### 🔹 ****6. Automation Coverage & Traceability****

* Auto-generate SQL and test logic for each validation.
* Store test results in Delta tables with traceable metadata.
* Enable dashboard reporting for stakeholders.

### ****2.2 Scope of Testing****

The test plan addresses validation across both **test and production environments**, accounting for delta-only loads and aligning with real-world ingestion patterns.

#### ✅ ****In-Scope Activities****

| **Category** | **Scope Details** |
| --- | --- |
| **Ingestion Validation** | Read Replica → Bronze CDA (S3) → Bronze |
| **Layer Validation** | Staging → Bronze (for delta only) Bronze → Silver/Gold (risk-based) |
| **Data Validation** | Field-to-field match, record count, duplicate detection, referential integrity, SCD |
| **Schema Validation** | Table structure, data types, constraints, column mapping |
| **Performance Testing** | Load durations, ingestion failures, resource usage in Databricks |
| **Test Automation Execution** | Notebook-driven validation, CI/CD integration, result persistence, and reporting |
| **Result Tracking** | Delta table logs, views for dashboards, manual JIRA export |

#### ❌ ****Out-of-Scope Activities****

| **Out of Scope** | **Reason** |
| --- | --- |
| Historical data validation | Assumed to be pre-loaded in ODS layer |
| Manual UI testing | Covered by a separate QA team |
| Functional test cases | This is a technical migration with no business logic changes |
| Defect triaging of legacy load issues | Existing known data defects are excluded from scope |
| Real-time ingestion testing | All ingestion is batch-based and delta-only for now |
| Full Silver/Gold testing | Limited to risk-based sampling due to no changes in logic |

### ****2.3 Environments in Scope****

| **Environment** | **Purpose** |
| --- | --- |
| **Test Environment** | Sample scenarios injected into both legacy and cloud versions Staging layers created for Oracle, PostgreSQL, and CDA |
| **Production Environment** | Post-load comparison of staging vs. Bronze ODS for each source Used for final validation before deployment cutover |

### ****2.4 Validation Sources and Targets****

| **Source** | **Validation Target** |
| --- | --- |
| On-Prem Oracle | Bronze ODS Layer |
| PostgreSQL Read Replica | Bronze ODS Layer |
| CDA Files in S3 | Bronze ODS Layer |
| Staging Tables | Used as input to all validation processes |

### ****2.5 Test Data Coverage Strategy****

* Sample test scenarios to be created with help from UI test teams.
* Test data will simulate various transaction types (create, update, delete).
* Staging layer expected to hold only **1-day delta data**, matching Bronze ingestion for the same day.
* Test coverage targets key business entities across all three Guidewire applications.

## **2. Test Objectives and Scope**

### ****2.1 Test Objectives****

The primary objective of this testing effort is to ensure that the data migration process from legacy Guidewire systems to Azure Data Lake is accurate, consistent, and technically equivalent to the legacy process. This includes validation of both **interim (Read Replica)** and **final (CDA)** ingestion methods, ensuring schema fidelity, data completeness, and transformation correctness.

The automation framework is designed to provide **scalable and repeatable validation** across Bronze, Silver, Gold, and Outbound layers, while prioritizing delta-based data loads due to the staging layer's truncate-and-load design.

The key objectives are:

* Validate **ingestion consistency** from both PostgreSQL Read Replica and CDA S3 files into the Bronze layer.
* Perform **parallel testing** of Read Replica and CDA flows **if both are available** during the test design and execution phase.
  + If not, **priority will be given to validating Read Replica** ingestion first.
* Ensure **schema validation** (table structure, column types, constraints) for all ingested tables.
* Verify **data consistency** at row and field level across all ingestion points (three-way reconciliation).
* Perform **risk-based validation** for Silver and Gold layer transformation logic.
* Confirm that **reference tables** are fully validated at each layer.
* Provide **high-level ingestion performance insights** using available one-day delta loads by comparing ADF vs. Databricks execution time.
* Validate **file generation in outbound layer**, provided access to mount/storage locations exists.
* Exclude cost comparison and business logic validation beyond ingestion equivalence.

### ****2.2 In-Scope****

The following elements are included in the scope of this automation-based validation plan:

#### ****Data Sources & Flows****

* **Read Replica ingestion** into Bronze Layer
* **CDA (Cloud Data Access) ingestion** into Bronze Layer
* **Staging Layer** created by data engineering for Read Replica, CDA, and On-Prem Oracle (for test environment only)
* **Three-way data reconciliation**: Oracle (legacy) vs. Read Replica / CDA vs. Bronze Layer

#### ****Validation Scope by Layer****

| **Layer** | **Scope** |
| --- | --- |
| **Bronze** | Full schema, count, and data-level validation for all tables provided by Data Engineering |
| **Silver** | Risk-based testing of transformation logic; full validation of reference tables only |
| **Gold** | Risk-based testing only; transformation logic spot-checks and count comparisons |
| **Outbound** | Only validation of file generation (presence in mount path); content-level validation is out of scope |

#### ****Performance Testing****

* **High-level ingestion timing** comparison:
  + ADF vs. Databricks execution times
  + Only if one-day delta data is available in a production-like environment
* Performance metrics are **informational only**, not pass/fail criteria.

#### ****Automation Coverage****

* Only the **table list agreed upon with the Data Engineering team** will be in scope.
* Validation will include:
  + Schema comparison
  + Record count
  + Field-level value comparison
  + Duplicate detection
  + SCD Type 2 checks (where applicable)
  + Referential integrity (where defined)

### ****2.3 Out of Scope****

The following areas are considered out of scope for the automation system integration test plan:

* Cost analysis or cost-based optimization comparisons (requires separate access and tooling).
* Validation of **business logic** beyond technical data matching.
* Identification or resolution of **existing known defects** in Silver, Gold, or Outbound layers.
* Any **functional defects** or issues already raised in legacy systems.
* **Transformation logic coverage** for Silver and Gold layers **except where agreed based on risk**.
* Full content-level validation of outbound feeds (unless explicitly required and access is provided).

### ****2. Test Objectives and Scope****

#### ****2.1 Test Objectives****

The objective of this automation system integration testing is to validate the **data accuracy, consistency, completeness, and structural integrity** of Guidewire data as it is ingested into Azure Data Lake using two distinct ingestion paths:

* **Read Replica Ingestion** (Interim approach)
* **CDA (Cloud Data Access) File Ingestion** (Final target approach)

The key goals of this automation testing framework are:

* To **automate end-to-end validation** of data flow across all Delta Lake layers — **Bronze**, **Silver**, and **Gold**.
* To verify that data ingested from the **read replica** or **CDA files** matches the data available in the legacy **Oracle on-premise environment**, ensuring **three-way reconciliation**.
* To validate that **no schema drift or data transformation anomalies** occur between ingestion sources and target layers.
* To conduct **parallel validation**, where possible, across both ingestion flows to ensure equivalence and readiness of the final solution.
* To generate test evidence (queries, conditions, and results) in a traceable and reusable format for audit and reporting purposes.

#### ****2.2 Scope of Testing****

The scope of the automated testing framework includes:

* Validation of ingestion from **PostgreSQL read replica** and/or **CDA S3 files** into the **Bronze layer**.
* **Transformation checks** across Bronze → Silver → Gold for selected tables.
* **Automated regression testing** on all in-scope tables provided and agreed by the **Data Engineering team**.
* Generation of **metrics, dashboards, and test logs** to support analysis and traceability.
* Automation of **schema validation**, **row count checks**, **SCD (Slowly Changing Dimensions)** checks, **duplicate detection**, and **referential integrity**.
* Testing across **Test and Production environments**, restricted to **delta loads only** (assumed to hold one day’s worth of data).
* High-level **performance benchmarking** using ingestion time comparison between **ADF vs. Databricks** for available data.

#### ****2.3 Out-of-Scope Items****

* **Cost analysis** related to ADF or Databricks execution — this requires cost analytics access which is not within the automation scope.
* **Detailed business logic validation** or historical defect tracking on **Silver**, **Gold**, or **Outbound layers**.
* Validation of any existing **known issues** in legacy datasets or transformation logic.
* Deep performance tuning recommendations or long-term historical trend benchmarking.
* **Manual validation activities** or UI-based test scenarios.

#### ****2.4 Special Considerations****

* **Parallel Testing**: If both **Read Replica** and **CDA** flows are available during test execution, **parallel validation will be performed** to ensure their equivalency. Otherwise, **priority will be given to validating the Read Replica flow first**, as it is the initial ingestion method implemented.
* **Table Scope**: Only the tables **explicitly listed and approved by the Data Engineering team** will be considered for test automation coverage across all layers.
* **Performance Testing**: High-level only — based on comparing **load times** for one-day delta data across ADF and Databricks pipelines. Deeper performance engineering is out of scope.
* **Silver/Gold Layers**:
  + Only **risk-based testing** will be performed.
  + **Reference tables** (such as lookup and configuration tables) will receive full coverage.
  + **Transformation-heavy tables** will be tested based on impact and availability of test data.
* **Outbound Feeds**: Only **file generation validation** will be done for outbound layers — ensuring that files are created and appear in the expected mount or storage locations. No further validation of file content or downstream consumption logic is included.

### ****3. Test Case Matrix****

| **ID** | **Test Scenario** | **Source(s)** | **Layer(s)** | **Validation Type** | **Automation Scope** | **Remarks** |
| --- | --- | --- | --- | --- | --- | --- |
| TC001 | Schema Validation – Table & Column Structure | Read Replica, CDA, Oracle | Staging → Bronze | Table/Column Match, Data Types, Nullability, Constraints | ✅ Yes | Based on metadata extraction |
| TC002 | Row Count Validation | Read Replica, CDA, Oracle | Staging → Bronze | Count Match | ✅ Yes | Validates completeness |
| TC003 | Duplicate Detection | Read Replica, CDA | Bronze | Primary/Composite Key Check | ✅ Yes | Detects data quality issues |
| TC004 | Referential Integrity Check | Read Replica, CDA | Bronze | Foreign Key Validation | ✅ Yes | Joins against reference tables |
| TC005 | Data Value Comparison | Read Replica, CDA, Oracle | Staging → Bronze | Field-by-field Check | ✅ Yes | For sample records; deterministic rules only |
| TC006 | SCD Type 2 Validation | Read Replica, CDA | Bronze → Silver | Insert/Update Validation | ✅ Yes (Risk-based) | Applied on selected tables only |
| TC007 | Transformation Rule Validation | Read Replica, CDA | Silver → Gold | Business Rule-based Comparison | ✅ Yes (Risk-based) | Based on defined logic and sample data |
| TC008 | Reference Table Validation | Read Replica, CDA | All Layers | Full Data Match | ✅ Yes | 100% coverage expected |
| TC009 | File Generation Check – Outbound Layer | Any | Outbound | File Existence & Format Check | ✅ Yes | Only checks presence and structure |
| TC010 | ADF vs. Databricks Load Time | Read Replica, CDA | Bronze | Performance Benchmark | ✅ Yes (High-level) | One-day delta load only |
| TC011 | CDA vs. Read Replica Comparison | Read Replica, CDA | Staging | Parallel Reconciliation | ✅ Conditional | If both sources are available in test |
| TC012 | Data Freshness Validation | Read Replica, CDA | Bronze | Timestamp/Load Date Check | ✅ Yes | Ensures current day load |
| TC013 | Schema Drift Detection (Long Term) | Read Replica, CDA | All | Schema Change Tracking | ❌ No | Out of scope; potential enhancement |
| TC014 | Cost Analysis | N/A | N/A | Pipeline Cost Validation | ❌ No | Out of scope due to access restrictions |
| TC015 | Known Defect Regression | N/A | Silver, Gold, Outbound | Business Rule Validation | ❌ No | Excluded from automation scope |

#### ✅ Notes:

* **Automation Priority**: Bronze layer gets full test automation; Silver/Gold layers follow risk-based selection.
* **Data Engineering Collaboration**: Only tables signed off by Data Engineering are included.
* **Test Data**: One-day delta load is assumed; results are stored in Delta tables and surfaced via dashboards.

Would you like this matrix in Excel or integrated into a Word/PDF format? Also, let me know if you’d like to add test case IDs with descriptions in a separate section.

### ****4. Out of Scope (QA)****

This section defines the boundaries of the test automation framework by clearly listing the tasks and validations **excluded** from the current scope of automated system integration testing. These exclusions have been agreed upon with the project and QA stakeholders based on tool limitations, access constraints, and the nature of the data migration effort.

#### ****4.1 Data and Functional Exclusions****

* **Historical Data Validation**  
  The automation testing framework will **not validate historical data** already present in the Bronze/Silver/Gold layers.  
  The scope of automation testing is **limited to delta (incremental) loads only**, typically covering one-day data load for comparison.
* **Legacy Defects or Known Data Issues**  
  Any known or ongoing data issues/defects in the legacy system (Oracle/PC layer) or existing IDMC pipelines are **excluded** from automated validation.
* **Business Logic Validation (Silver, Gold, Outbound)**  
  The automation framework does **not cover business logic validations** for transformation rules applied within Silver and Gold layers.  
  Only **risk-based validation for key transformation tables** and **full validation for reference tables** will be performed.

#### ****4.2 Technical Exclusions****

* **Cost Analysis**  
  Execution cost comparison between ADF and Databricks is **not in scope**, due to lack of access to cost analytics tools and billing reports.
* **End-to-End Performance Tuning**  
  Performance testing is limited to **ingestion timing benchmarks** for 1-day loads. Deep performance profiling or optimization is **not included**.
* **Outbound File Content Validation**  
  The framework will only confirm **whether files are generated** in outbound folders/mounts.  
  File **content-level comparisons** or downstream integration validations are **out of scope**.
* **Source-to-Target Business Mapping Validation**  
  The framework does **not validate mapping documents** or transformation intent, as this is a **technical migration** and **no new logic is introduced**.
* **UI-Based or Functional Workflow Testing**  
  Front-end validation, workflow UI behavior, or UI automation activities are **not covered** under this test plan.

#### ****4.3 Environment and System Constraints****

* **Integration/Downstream Systems Testing**  
  The test team will not perform any validation of data usage in **downstream applications**, dashboards, or third-party systems.
* **Firewall and Cross-Platform Access Restrictions**  
  Due to **network restrictions**, integration between **Databricks (cloud)** and **JIRA (self-hosted)** is not feasible.  
  Hence, **JIRA test metrics will be manually updated** based on exported reports from automation.
* **Regression Testing of Existing Code**  
  As the ingestion is being re-pointed to new data sources (Read Replica / CDA) without introducing transformation changes, **regression testing of existing transformation code is out of scope**.